

Capítulo 3

Sistemas que aprenden

1. Introducción.

Un sistema de aprendizaje es un programa de ordenador que toma decisiones basadas en experiencias acumuladas de casos resueltos con éxito. A diferencia de un sistema experto que resuelve problemas usando un ordenador pero con un modelo de razonamiento humano, un sistema que aprende puede usar diferentes técnicas para aprovechar el poder computacional de un ordenador, sin relación con el proceso de pensamiento humano.

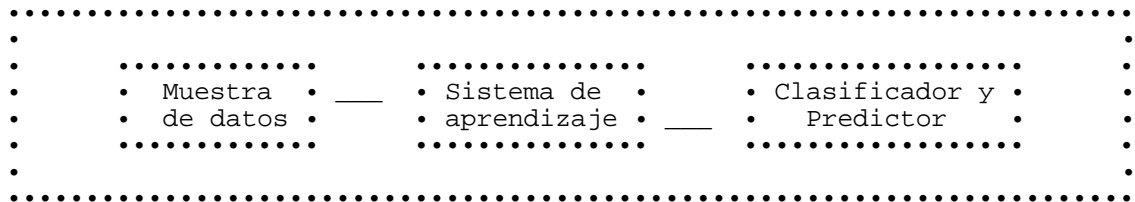


Fig. 3.1: Sistema que aprende

Una de las tareas más destacadas y básicas del aprendizaje es la de clasificación o predicción. En este caso el sistema tiene disponible un conjunto finito o muestra de ejemplos de casos resueltos. Los datos de cada caso estriban en un conjunto de observaciones o características de ese caso (patrón) y la correspondiente clasificación correcta. El aprendizaje consiste (Fig. 1) en elegir una estructura modelo y adaptar correctamente unos parámetros de ésta para una óptima clasificación de los datos conocidos. El resultado es un clasificador, sistema que proporciona una decisión para cada patrón admisible que se le proporcione (Fig. 2).

Técnicamente, esta forma de aprendizaje es denominada aprendizaje supervisado, en el sentido, de que el sistema aprende de un conjunto conocido de casos correctamente clasificados, que han sido producidos por un experto humano que supervisa la elección de estos casos. El aprendizaje no supervisado o clustering es un problema más difícil; en él los casos no están clasificados y el objetivo es identificar clusters de patrones que son similares, identificando, de esta manera, posibles clases. En esta tesis el aprendizaje siempre se entenderá supervisado. En los siguientes apartados se detallan algunos de los sistemas de aprendizaje más extendidos que se han dividido en tres grandes campos: métodos estadísticos, redes neuronales y reglas de decisión. Previamente a esta exposición se presentan algunas formas de estimar la bondad de un sistema de aprendizaje.

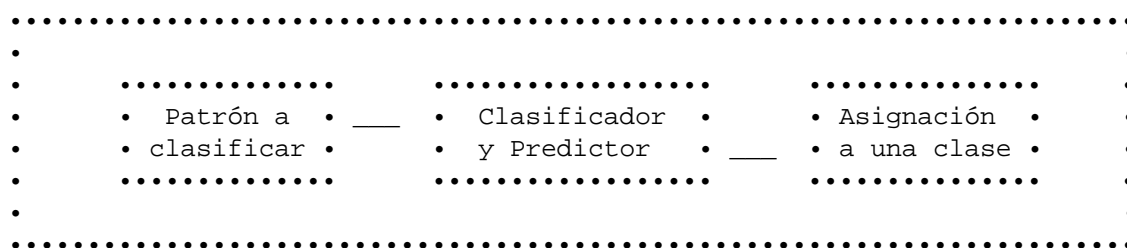


Fig. 3.2: Utilización de un clasificador

2. Rendimiento de un sistema que aprende.

2.1 Medida del error.

La medida más usada para evaluar el éxito de un clasificador es la tasa de error, que de forma empírica puede definirse como

$$\text{tasa de error} = \text{número de errores} / \text{número de casos}$$

utilizándose como estimador de la tasa de error verdadera o real la tasa de error de un clasificador sobre un número asintóticamente grande de nuevos casos seleccionados de manera independiente de los casos usados en el diseño del clasificador. Si no es posible generar estos nuevos casos de manera fácil, se puede recurrir a la tasa de error aparente, definida como la tasa de error de la muestra usada en el diseño del clasificador. Lógicamente la tasa de error aparente casi siempre es un valor menor que la tasa de error real y a menudo un estimador demasiado optimista de ésta.

Para solucionar este problema se divide la muestra de aprendizaje en casos de entrenamiento y casos de test, y se calcula la tasa de error empírica sobre los casos de test.

Existen diversas técnicas para obtener esa división de manera que la tasa de error calculada se aproxime a la real. Destacan tres de ellas:

Muestreo aleatorio.

Para ello se reparte de manera aleatoria la población de muestra entre casos de entrenamiento y casos de test, obteniéndose la tasa de error sobre estos últimos. Se repite un número de veces el procedimiento, estimándose la tasa de error real como la media de las tasas de error de cada prueba.

Validación cruzada.

Se fracciona la población en k conjuntos de aproximadamente igual tamaño. Se toma como casos de test un conjunto dado y se entrena el sistema con los $k-1$ restantes, obteniéndose para cada conjunto una tasa de error. La tasa de error real se aproxima por la media de las k tasas resultantes. Una de las variantes más utilizadas de esta técnica es la conocida como dejando uno fuera ("leaving one out" [Lachenbruch68]), que consiste en utilizar cada caso como conjunto test de un solo individuo, y los restantes como conjunto de entrenamiento. La tasa de error es ahora el número de errores cometidos en cada test dividido por el número de elementos en la muestra. Realmente la validación cruzada es una generalización del método de dejando uno fuera atribuida a [Stone74].

Bootstrapping.

Este método, originalmente desarrollado en [Efron82], consiste en realizar un muestreo con reemplazamiento en la muestra inicial siendo copiados en el conjunto de entrenamiento sucesivos casos de la muestra escogidos aleatoriamente y con reemplazamiento. Este proceso se repite hasta que el conjunto de entrenamiento tenga los mismos individuos que la muestra inicial. El conjunto de test son aquellos casos que no estén en el conjunto de entrenamiento. La tasa de error de este grupo es un estimador de la tasa real, aunque lo habitual es repetir el proceso un número determinado de veces y hallar la media.

En [Weiss91] se realiza una comparación entre estos métodos recomendándose validación cruzada para muestras mayores de 50 casos validación cruzada o su variante dejando uno fuera, teniendo en cuenta que esta última tiene un coste computacional mucho mayor, por lo que no se recomienda con más de 100 casos. Para muestras de menos de 50 casos se recomienda bootstrapping.

2.2 Complejidad del clasificador.

Dos propiedades caracterizan un clasificador además de su tasa de error. Una es el

número de características u observaciones necesarias de cada dato para obtener una correcta clasificación. Esto es debido a que no siempre más información produce un mejor sistema y sí de mayor complejidad. Esto dará lugar a una necesaria selección de características previa a la creación del clasificador. Este tema se volverá a tratar en el capítulo 5 bajo el nombre de sensibilidad. La segunda propiedad a tener en cuenta es la propia complejidad del clasificador obtenido, es decir, número de reglas, legibilidad de éstas, esfuerzo computacional necesario, memoria usada, etc... Estos atributos deberán ser tenidos en cuenta a la hora de diseñar un sistema de aprendizaje.

3. Sistemas estadísticos.

El problema de clasificar a un individuo a partir de una muestra de la población a la que pertenece, ha sido ampliamente estudiado por la Estadística, existiendo una amplia literatura al respecto. Estos métodos tienen unos fundamentos teóricos muy fuertes y conforman una base natural para nuevos procedimientos que utilicen las capacidades de los modernos ordenadores. En los siguientes apartados se comentan los análisis estadísticos clásicos: Bayesiano y discriminante lineal, y la técnica del vecino más próximo, relativamente más reciente en el tiempo y ampliamente utilizada en esta memoria.

3.1 Clasificador Bayesiano.

Un clasificador óptimo se puede obtener basándose en la teoría de probabilidad del análisis de Bayes. Esta teoría supone tamaño de muestra asintóticamente infinito e independencia de las probabilidades de cada clase. Con estas condiciones se puede calcular las distribuciones de probabilidad de cada clase en la población. Así si C_i son las clases existentes en la población y e es un conjunto de características de un individuo a clasificar, éste se asignaría a la clase C_i que cumpla:

$$\forall j \neq i. P(C_i|e) > P(C_j|e)$$

donde $P(C|e)$ es calculada por la regla de Bayes como

$$P(C|e) = \frac{P(e|C)P(C)}{P(e)}$$

luego la inecuación anterior quedaría:

$$\forall j \neq i. P(e/C_i)P(C_i) > P(e/C_j)P(C_j)$$

por último, si se supone que e es el conjunto formado por d observaciones $\{e_1, e_2, \dots, e_d\}$ y se asume la independencia de las probabilidades condicionadas, la desigualdad anterior resulta

$$\forall j \neq i. \prod_{k=1}^d P(e_k/C_i)P(C_i) > \prod_{k=1}^d P(e_k/C_j)P(C_j)$$

Con esta fórmula, dado un nuevo caso se obtienen las probabilidades condicionadas para cada clase, y se escoge aquella con mayor probabilidad. Las probabilidades se obtienen de la muestra o test de aprendizaje; así la probabilidad $P(C_i)$ es la proporción de la clase i en la muestra. Para las características discretas, el valor de $P(e_j/C_i)$ se estima igualmente a partir de la proporción de casos con esa característica j en la clase i . Por último, para características con distribución continua, se discretizan tomando intervalos de clase de longitud fija, estimándose la probabilidad con la frecuencia de casos para cada intervalo.

3.2 Análisis Discriminante Lineal.

El nombre de lineal [Fisher36] proviene de que la separación entre las clases será indicada por una combinación lineal de las d características. Geométricamente, esto significa que un hiperplano de dimensión $d-1$ intentará separar las clases. En la figura 3 se observa una separación idealizada de dos clases en el plano formado por dos características.

Aunque, lógicamente, en la mayoría de las situaciones reales esta separación no será posible sin error, este sistema tiene la ventaja de tener una estructura muy simple y de ofrecer la posibilidad de obtener otro tipo de superficie de separación por linealización a trozos mediante estos hiperplanos.

La forma general de cualquier clasificador lineal viene dada por la expresión

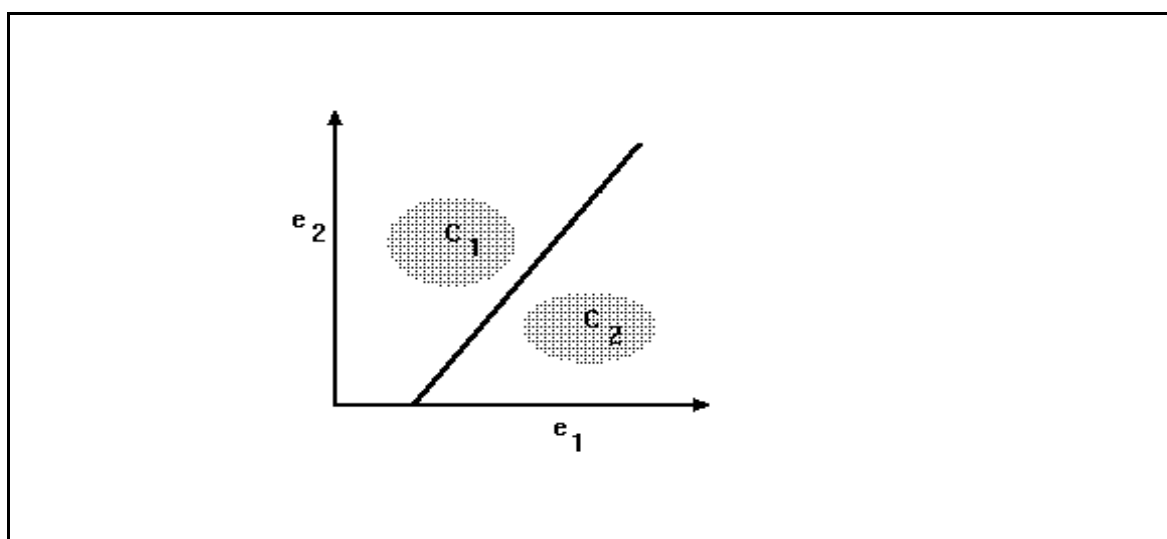


Fig. 3.3: Clasificador lineal

$$\omega_1 e_1 + \omega_2 e_2 + \dots + \omega_d e_d + \omega_0$$

donde los pesos ω_i son constantes a ser estimadas. En esta situación, para separar más de dos clases se necesita un hiperplano que divida las clases de dos en dos, o un hiperplano que separe cada clase del resto. Una modificación consiste en encontrar una combinación lineal como la anteriormente expresada para cada clase, de forma que un nuevo caso es clasificado en la clase cuya expresión correspondiente da un mayor valor.

Para encontrar el valor de los pesos, la estadística clásica supone que la función de densidad de cada clase C_i se distribuye según una normal multivariante $N(\mu_i, \Sigma_i)$ donde μ_i y Σ_i son respectivamente el vector de medias y la matriz de covarianzas de las características de la clase i . Con esta consideración, el vector de pesos $\{\omega_1, \omega_2, \dots, \omega_d\}$ puede ser calculado para cada clase i como $\mu_i^t \Sigma_i^{-1}$, y ω_0 como el vector anterior por $-1/2\mu_i$.

Aunque la suposición realizada exige características con distribución continua y según una Normal, diversos estudios (pp.152-160 de [Weiss91]) demuestran con resultados empíricos la posible aplicación con buenos resultados de este método a poblaciones sin estas restricciones.

3.3 Aproximación mediante la técnica de vecinos.

Las bases para la técnica de clasificación de los vecinos más próximos fueron establecidas en dos estudios de Fix y Hodges. En el primero de ellos, se analizan las propiedades de

consistencia de procedimientos de clasificación no paramétricos y su asintótica bondad en muestras de gran tamaño. El segundo trabajo considera el caso más real de muestras de tamaño pequeño e incluye el estudio de varios ejemplos de clasificación con técnicas de vecinos y su comparación con procedimientos paramétricos. Estos dos estudios abrieron una nueva área de investigación en las técnicas de clasificación de patrones, que anteriormente habían estado limitadas a procedimientos paramétricos como el análisis discriminante Bayesiano.

Algunos años después, Cover y Hart enunciaron formalmente la regla del vecino más próximo y la desarrollaron como una herramienta para la clasificación de patrones. Su enunciado es básicamente el siguiente:

Si se tienen n pares de valores (x_i, θ_i) $1 \leq i \leq n$, donde las x_i toman valores en un espacio métrico X con una distancia definida d , y las θ_i toman valores en el conjunto $\{1, 2, \dots, M\}$, es decir, cada θ_i es el índice de una clase o categoría a la que pertenece x_i , dado un punto x en X , se pretende estimar su clase correspondiente. Para ello se define x' :

$$x' \in \{x_1, x_2, \dots, x_n\} \min_i d(x_i, x) = d(x', x)$$

es decir, x' es el vecino más próximo. Una vez calculado se asigna a x la clase θ' correspondiente a x' .

Cover y Hart demostraron la convergencia de la regla y que la probabilidad de error R estaba acotada inferiormente por la probabilidad de error de Bayes R^* y superiormente por $2R^*$. Concretamente

$$R^* \leq R \leq R^* \left(2 - \frac{M}{M-1} R^* \right)$$

Este resultado puede interpretarse como que la mitad de toda la información que facilita una muestra infinita está contenida en la regla del vecino más próximo.

Estos resultados pueden ser generalizados al obtener los k vecinos más cercanos a x , y asignarle la clase mayoritaria entre esos k vecinos. En la figura 4 se puede observar su aplicación suponiendo que el candidato está señalado por x y $k=5$, entonces la clase asignada sería $+$. Para la regla generalizada de los k -vecinos se pueden encontrar unas cotas del error con una fórmula similar a la anterior, en la que además se pone de manifiesto que la cota superior es monótonamente decreciente en k .

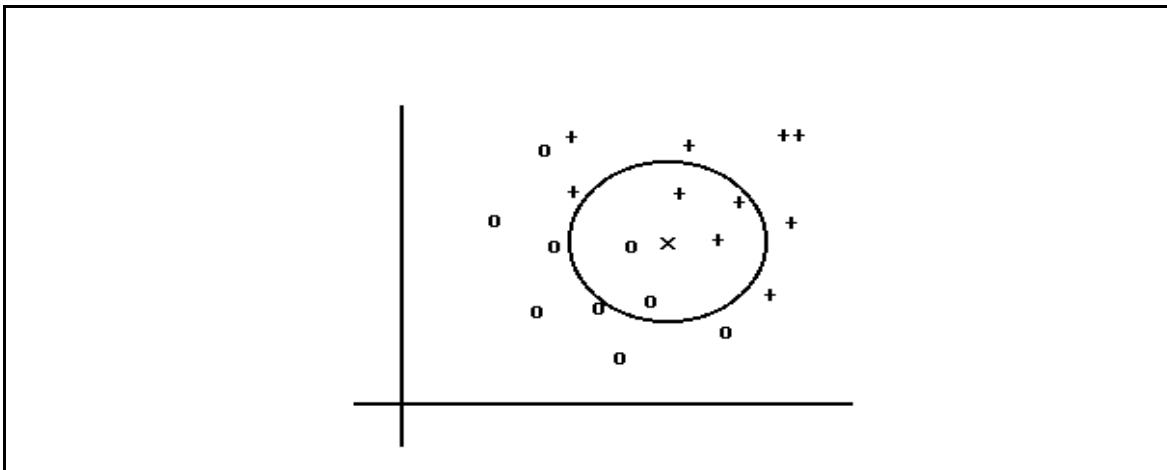


Fig. 3.4: Clasificador mediante k-vecinos

Por último, dentro de las contribuciones iniciales a esta técnica, Patrick y Fischer generalizan la regla para k vecinos en un problema de dos clases, basándose en la teoría de las regiones de tolerancia estadística. Este estudio tiene una importancia práctica pues permite la posibilidad de preprocesar o editar los datos de entrenamiento obteniendo reducciones computacionales significativas, sin apenas penalización de la probabilidad de error.

A partir de estas aportaciones la técnica ha tenido un gran desarrollo [Dasarathy91], con generalizaciones, avances algorítmicos, edición del fichero de entrenamiento, conceptos de clustering, técnicas borrosas, mejoras computacionales, etc.

Una de estas mejoras merece ser comentada por su amplia utilización en esta memoria. En [Dudani76] se introduce la idea intuitiva de ponderar las observaciones de forma que el vecino más cercano tenga más peso o influencia que un vecino más lejano. Concretamente propone la siguiente regla:

Sean n pares de valores $\Omega = \{(x_i, \theta_i) / 1 \leq i \leq n\}$, donde las x_i toman valores en un espacio métrico X con una distancia definida d , y las θ_i toman valores en el conjunto $\{1, 2, \dots, M\}$, es decir, cada θ_i es el índice de una clase o categoría a la que pertenece x_i . Dado un punto x en X , se pretende estimar su clase correspondiente. Para ello se obtienen los puntos $x_{j'} \in \Omega$ $1 \leq j' \leq k$, que son los puntos más cercanos a x en Ω . Supóngase que están ordenados de forma que $x_{1'}$ es el más cercano y que $x_{k'}$ es el más lejano, defínase $d_j = d(x_{j'}, x)$ y un atributo ω_j para el j -ésimo vecino:

$$\omega_j = \begin{cases} \frac{d_k - d_j}{d_k - d_1}, & d_k \neq d_1 \\ 1, & d_k = d_1 \end{cases}$$

Una vez calculado los pesos ω_i , la clase de x será la categoría cuya suma de pesos para todos los k vecinos sea mayor.

Dudani propone otras posibles definiciones para los pesos tales como $\omega_i = 1/d_i$ ó $\omega_i = k - j + 1$. Aunque la utilización de esta regla en el caso de asumir conjuntos de entrenamientos infinitos conlleva una tasa de error asintóticamente mayor que la regla simple, según se demuestra en [Bailey78], estudios posteriores como [Macleod87] demuestran que para el caso de tamaño de muestras finito, esa aseveración no es cierta, demostrándose en algunos casos justo lo contrario.

4. Redes Neuronales.

4.1 Introducción.

La teoría de Redes Neuronales está inspirada en los conocimientos existentes del funcionamiento del sistema nervioso de los seres vivos [McCulloch43]. A partir de éste, surgen modelos matemáticos relativamente simples, que intentan simular el proceso de aprendizaje de las neuronas interconectadas en el cerebro. Para ello se definen una serie de capas (llamadas también leyes), donde cada capa tiene un número determinado de nodos que se relacionan con la capa anterior, recibiendo unos valores numéricos o impulsos ponderados. El nodo suma estos impulsos y mediante una función da a los nodos de la siguiente capa un impulso propio. La primera capa es llamada vector de entrada que toma los datos del fichero de aprendizaje y la última capa es denominada de salida, cuyos nodos adquieren (mediante un proceso de aprendizaje) distintos valores para patrones distintos. Muchas redes neuronales tienden a ser variaciones de clasificadores lineales como el descrito en el apartado anterior, aunque sin la necesaria suposición sobre la distribución de la población.

A grandes rasgos se pueden distinguir dos modelos de redes neuronales: estáticas y dinámicas, según las ecuaciones que relacionan los nodos no dependan o sí, respectivamente, del tiempo. Las redes estáticas son las más extendidas y un gran número de paradigmas están descritos en la abundante literatura existente. Estas redes se pueden dividir en tres grupos dependiendo del tipo de aprendizaje: con aprendizaje supervisado, redes auto-organizadas con aprendizaje no supervisado y redes híbridas que tratan de

explotar las ventajas de los dos modelos anteriores. Los paradigmas más extendidos dentro de cada uno de estos tipos de aprendizaje son respectivamente el Perceptrón Multicapa propuesto en [Rosenblatt58], las redes auto-organizadas en [Kohonen82] y la red de función de base radial (Radial Basis Function) en [Moody88].

Dadas las características de la investigación de esta Tesis sólo se van a comentar las redes estáticas con aprendizaje supervisado, y concretamente el modelo perceptrón multicapa (PMC) y más ligeramente el modelo Radial Basis Function (RBF).

4.2 El modelo perceptrón multicapa.

4.2.1 Ecuaciones.

El modelo más simple de red neuronal es el perceptrón de salida única, estrictamente equivalente al análisis discriminante lineal. Así la expresión

$$\omega_1 e_1 + \omega_2 e_2 + \dots + \omega_d e_d + \omega_0$$

puede ser reescrita

$$\sum_i \omega_i I_i + \theta$$

donde I_i describe las entradas y θ es una constante llamada umbral, que geoméricamente indica el punto donde la línea cruza con el eje de abscisas. Este perceptrón clasifica una entrada como perteneciente a una entre dos clases. Así si la suma de las entradas por los pesos da mayor de $-\theta$, la salida es 1, en otro caso la salida es 0. Esta función de activación recibe el nombre de función no lineal en escalón.

Al ser la separación lineal poco eficiente en la mayoría de las situaciones reales, el perceptrón de salida simple puede ser extendido a redes neuronales más complicadas. De esta manera situando los perceptrones en cascada mediante una estructura de múltiples capas podemos implementar fronteras de decisión complejas. Como se puede apreciar en la figura 5 el vector de entrada alimenta directamente a las neuronas o nodos de la primera capa, las salidas de la primera capa directamente a las neuronas de la segunda capa, y así sucesivamente. La red representada en la figura tiene un único nodo en la capa de salida, necesario para el reconocimiento de poblaciones con dos clases; para casos de múltiples clases son necesarios más nodos en la capa de salida.



Fig. 3.5: Estructura de una red neuronal

Existe una gran variedad de modelos de redes neuronales, siendo uno de los más extendidos el Perceptrón Multicapa (PMC) que sigue el siguiente modelo matemático

$$y_j^p = \sum_{i=1}^{N_p} \omega_{ij}^p u_i^{p-1} + \theta_j^p = f(y_j^p)$$

donde y_j^p representa el estado interno de la neurona j de la capa p , u_i^p la salida de esa neurona, ω_{ij}^p los pesos que conectan la neurona i de la capa $p-1$ con la neurona j de la capa p , los pesos positivos indican excitación y los negativos inhibición, el término θ_j^p representa un estímulo exterior. La salida de las neuronas de la primera capa o capa de entrada proporciona los valores de las características de la muestra de entrenamiento. La función f suele implementarse mediante una sigmoide no lineal que permite, en comparación con la función escalón, una transición gradual de una respuesta a la otra.

Para el aprendizaje de los pesos existen numerosos algoritmos propuestos. Uno de los más extendidos y el escogido en esta investigación es el conocido como retropropagación propuesto en [Rumelhart86].

4.2.2 Algoritmo.

Para describir el algoritmo se simplifica la notación anterior quedando la siguiente ecuación que es representada esquemáticamente en la figura 6:

$$y_j^p = f\left(\sum_i \omega_{ij}^p y_i^{p-1}\right)$$

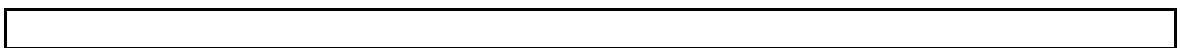


Fig. 3.6: Estructura del perceptrón multicapa

donde θ_j ha sido incluido en la suma de entradas a la neurona j -ésima y f es la función sigmoide no lineal con derivada:

$$f'(\alpha) = f(\alpha)(1 - f(\alpha))$$

y cuya representación gráfica se puede ver en la figura 7.



Fig. 3.7: Función sigmoide no lineal

El algoritmo de aprendizaje usado utiliza una técnica de búsqueda por gradiente. La función a minimizar es la suma de los errores al cuadrado sobre los elementos de la muestra. Es decir, de las diferencias entre la salida de la red neuronal en la última capa y_j^L y la salida esperada en la muestra de entrenamiento d_j para cada característica j . Para cada caso en el conjunto de elementos de entrenamiento el error es:

$$E = \frac{\sum_j (d_j - y_j^L)^2}{2}$$

Las componentes del vector gradiente en un punto son las derivadas parciales de E respecto a los pesos, por tanto los pesos pueden ser calculados mediante un proceso iterativo como el siguiente:

$$\omega_{ij}^p(k+1) = \omega_{ij}^p(k) - \mu \frac{\partial E}{\partial \omega_{ij}^p} \quad (15)$$

donde la derivada se puede calcular mediante la regla de la cadena

$$\frac{\partial E}{\partial \omega_{ij}^p} = \frac{\partial E}{\partial y_j^p} \frac{\partial y_j^p}{\partial \omega_{ij}^p}$$

y teniendo en cuenta la derivada de f queda

$$\frac{\partial E}{\partial \omega_{ij}^p} = \frac{\partial E}{\partial y_j^p} y_j^p (1 - y_j^p) y_j^{p-1}$$

El primer término representa la sensibilidad del error a la salida de la neurona j -ésima de la capa p , la cual muestra su influencia en el error a través de todos los nodos en las siguientes capas. Así el primer término de la derecha de la ecuación se puede expresar como función de la sensibilidad de los nodos de las capas sucesivas:

$$\frac{\partial E}{\partial y_j^p} = \sum_k \frac{\partial E}{\partial y_k^{p+1}} y_k^{p+1} (1 - y_k^{p+1}) \omega_{jk}^{p+1}$$

Este proceso continúa para las sucesivas capas hasta que se alcanza la capa de salida donde la sensibilidad de los neuronas de esta capa viene dada por:

$$\frac{\partial E}{\partial y_j^L} = y_j^L - d_j$$

que es el error para cada elemento de la muestra. Al ser la sensibilidad o error en cada capa calculada desde la última capa hacia atrás, es por lo que recibe este algoritmo el nombre de retropropagación.

En la búsqueda por gradiente el algoritmo visto es muy sensible a la elección del parámetro μ . Si es muy pequeño (del orden de centésimas o milésimas) la búsqueda puede ser muy lenta. Pero si es demasiado grande el gradiente puede descender describiendo grandes oscilaciones. Esto se puede evitar añadiendo un término llamado momentum en la ecuación (15). La idea de [Plaut86] es dar a cada peso una inercia de forma que tienda a cambiar en la dirección de la fuerza descendente media que sienta, en lugar de oscilar ampliamente con cada pequeño "tirón". Con ello se puede aumentar el valor del parámetro de convergencia μ sin que ocurran oscilaciones divergentes.

Para implementarlo basta con añadir una contribución del paso anterior en la actualización de los pesos:

$$\Delta\omega(k+1) = -\mu \frac{\partial E}{\partial \omega} + \alpha \Delta\omega(k)$$

donde α es el parámetro momentum, con un valor entre 0 y 1.

4.2.3 Capacidad y limitaciones del modelo.

El Perceptrón Multicapa es capaz de implementar cualquier aplicación no lineal, y dado un conjunto de ejemplos, el algoritmo anterior aprende la aplicación en los puntos ejemplos. Sin embargo, hay otras cuestiones prácticas que deben ser tenidas en cuenta: una es la complejidad en el tiempo del proceso de aprendizaje, y otra la habilidad para obtener buenos resultados con ejemplos que no estén en el conjunto de entrenamiento; como es lógico ambas cuestiones están muy relacionadas.

Los resultados teóricos demuestran que el PMC es capaz de implementar cualquier función no lineal, pero esto es verdad si el tamaño de la red es arbitrariamente grande [Funahashi89]. En general no se conoce el tamaño óptimo para un problema determinado, aunque usualmente no suelen ser empleadas más de tres capas. Respecto al número de neuronas en la capa intermedia, las sugerencias encontradas en la literatura proponen una proporción lineal al número de características de la población [Hush89].

Otra cuestión es la elección de la tasa de aprendizaje μ . Alguna regla propuesta sugiere hacer la tasa de aprendizaje para cada neurona inversamente proporcional a la media de los pesos que llegan a esa neurona. La elección inicial de los pesos es importante porque el método del gradiente busca mínimos locales, siendo pues necesario ejecutar el algoritmo con diferentes inicializaciones hasta conseguir una buena aproximación al óptimo. El problema de hallar los pesos se ha demostrado como NP [Blum88], lo que hace que el algoritmo de aprendizaje sea extraordinariamente lento. Por último, queda reflejar que el PMC necesita un gran número de casos de entrenamiento. Algunas heurísticas recomiendan alrededor de diez veces el número de pesos [Baum89].

En general, la investigación con redes neuronales conlleva un fuerte factor de prueba y error hasta encontrar un modelo suficientemente acurado.

4.3 La red de función de base radial.

4.3.1 Ecuaciones.

El modelo RBF es una red con dos capas. La primera capa sigue un modelo auto-organizado similar al propuesto por Kohonen, y calcula la similaridad entre la entrada y los

patrones representativos de cada nodo. Estas medidas de similitud son pasadas a la segunda capa, que sigue un modelo perceptrón, de forma que da una suma de pesos por similitudes como salida.

La ecuación de la primera capa viene dada por

$$u_j = e^{-\frac{\|x-w_{1j}\|^2}{2\sigma_j^2}} \quad j = 1, 2, \dots, N_1$$

donde u_j es la salida del nodo j , x es la muestra de entrada, w_{1j} es el vector de pesos (es decir, el patrón representativo del nodo j), σ_j es el parámetro de normalización del nodo j , y N_1 es el número de nodos de la capa. Es de tener en cuenta que la salida de cada nodo está en el rango $[0,1]$ y que dada una entrada, cuanto mayor es la salida del nodo, más se aproximan los pesos de ese nodo (w_{1j}) a la entrada.

Los pesos se pueden calcular por un algoritmo de autoaprendizaje como el propuesto por Kohonen o por el popular algoritmo de clustering "K-means". Los parámetros σ_j , que miden la dispersión de los datos asociados a cada nodo, se pueden calcular por una heurística como la del vecino más cercano. La implementación de estos algoritmos se puede ver en [Neural93].

Las ecuaciones de la segunda capa vienen dadas por

$$y_j = w_{2j}^T u_1 \quad j = 1, 2, \dots, N_2$$

donde y_j es la salida del nodo j -ésimo de la segunda capa, w_{2j} es el vector de pesos para este nodo, u_1 es el vector de medidas de similaridad resultado de la primera capa (con una componente unidad añadida como en el caso PMC), y N_2 es el número de nodos de la segunda capa. La salida de esta capa es una combinación lineal ponderada de las medidas de similaridad resultado de la primera capa. Es decir, la red al completo realiza una transformación no lineal de $\mathbb{R}^2 \rightarrow \mathbb{R}^{N_2}$ formando una combinación lineal de funciones básicas, en este caso, funciones de distribución normal o gaussianas de distinta media y varianza. El nombre del paradigma proviene de la simetría radial de las funciones que se emplean.

4.3.2 Capacidad y limitaciones del modelo.

Las ventajas que presenta RBF con respecto al PMC son varias:

- La fase de entrenamiento es más rápida, debido a la división del proceso, en dos etapas relativamente eficientes. En el proceso de clustering de la primera capa el tiempo es lineal, y aunque la solución no es óptima, tampoco es necesario, pues durante la segunda etapa el resultado puede ser excelente a partir de una solución suficientemente buena.
- Obtiene mejores resultados en problemas de decisión y clasificación [Leonard91].
- La representación de la primera capa mediante funciones de densidad del espacio de entrada es más natural que la representación del PMC. Además puede ser usada para medir la probabilidad de que un nuevo vector de entrada sea parte de la misma distribución de los valores de entrenamiento [Leonard92].

Entre sus desventajas se pueden señalar:

- La primera fase es no supervisada, lo que puede producir pérdida de importante información.
- El PMC tiene una representación más compacta.

5. Sistemas de reglas.

5.1 Introducción.

Los sistemas que aprenden en base a reglas, examinan una muestra de casos resueltos y proponen un conjunto de reglas de decisión en términos de un modelo subyacente. Para los métodos revisados hasta ahora, no hay duda de la necesidad de usar el ordenador para procesar la información; igualmente una vez que el entrenamiento se ha completado, clasificar un nuevo caso también es muy sencillo para el computador. Desgraciadamente para muchos expertos no informáticos, este proceso puede producirle cierta desconfianza o recelo, por su nula participación en la predicción. Además, si el usuario quiere una explicación del por qué de una determinada clasificación, la mayoría de las técnicas matemáticas sobre inferencia dan unas respuestas no fácilmente entendibles por usuarios no matemáticos.

Estas razones dieron lugar a la búsqueda de nuevos métodos de aprendizaje, de forma que la representación de la solución se hiciera en un formato que fuera fácilmente entendible y compatible con el razonamiento humano. Dejando aparte el debate filosófico sobre la representación del conocimiento y razonamiento humano, es fácil de ver que reglas de la forma

Si X es verdad e Y es falso entonces A

son sencillas de entender y de aceptar por los humanos. La condición X puede ser evaluada de forma directa para características discretas, mientras que para parámetros continuos es necesario dividir en intervalos su dominio y usar operadores relacionales del tipo "mayor que" o "menor que".

El objetivo de un sistema de este tipo es encontrar un conjunto de reglas que cubra todos los casos de una clase particular sin cubrir los casos de otras clases, o al menos, minimizando el número de estos casos o errores cometidos.

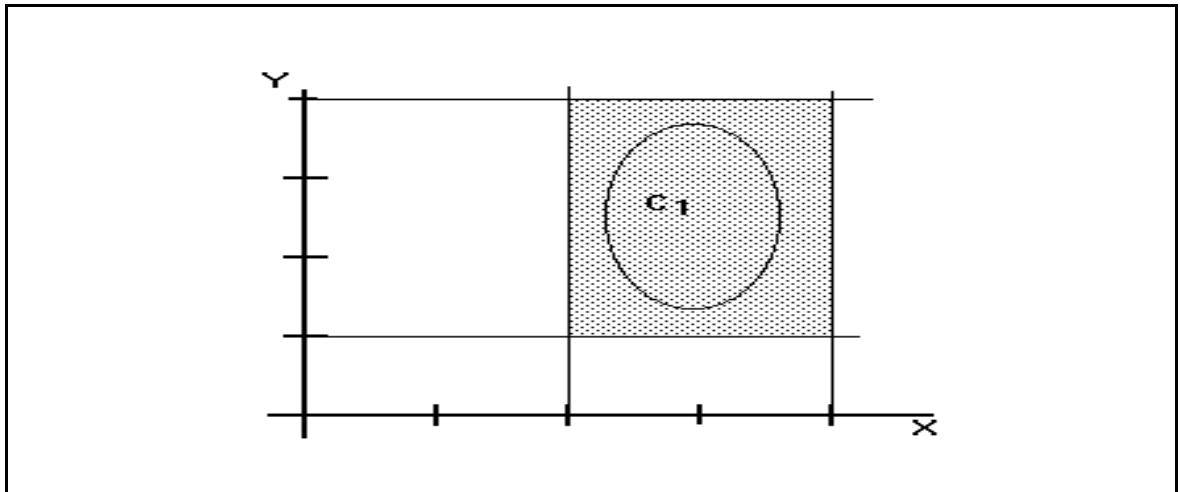


Fig. 3.8: Clasificador mediante reglas

Geoméricamente un sistema que aprende y que encuentra reglas como las anteriormente definidas, describe regiones de decisión cuyas fronteras son líneas paralelas a los ejes, formando zonas rectangulares. Por ejemplo con dos variables continuas X e Y y las condición $X > 2$ AND $Y > 1$ y $X < 4$ AND $Y < 4$, la región sería como en la Fig. 8. Ajustando apropiadamente el tamaño de los rectángulos, se puede aproximar cualquier superficie y cubrir cualquier clase. Por tanto, la efectividad y eficiencia con la que un sistema de aprendizaje pueda cubrir los datos mediante regiones de forma rectangular determinará la bondad de estos sistemas.

Dentro de las técnicas desarrolladas para encontrar un conjunto de reglas que represente el conocimiento de una base de datos, destacan dos metodologías: la construcción previa de árboles de decisión o la construcción directa de las reglas mediante otros métodos. En los siguientes apartados se describen las principales características y variantes de éstos.

5.2 Aproximación mediante árboles de decisión.

5.2.1 Introducción.

Dentro de esta metodología se encuentran dos sistemas que partiendo de orígenes diferentes y por caminos independientes han llegado a métodos similares: uno es el sistema CART (Classification and regresion tree) [Breiman84] basado en los estudios de [Friedman77], el otro más extendido es el sistema ID3 [Quinlan86] y su sucesor C4.5 [Quinlan93].

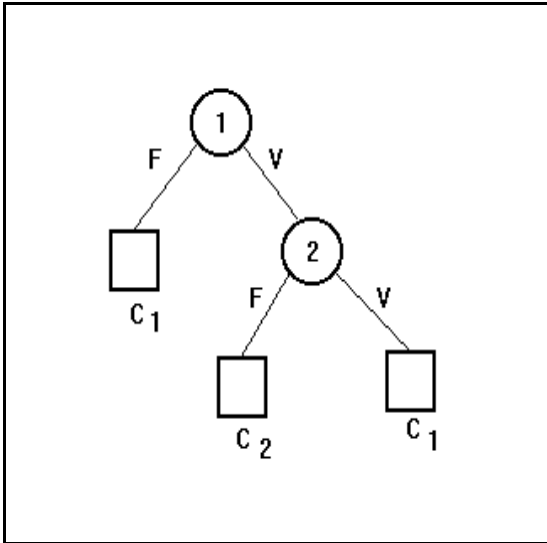


Fig. 3.9: Arbol de decisión binario

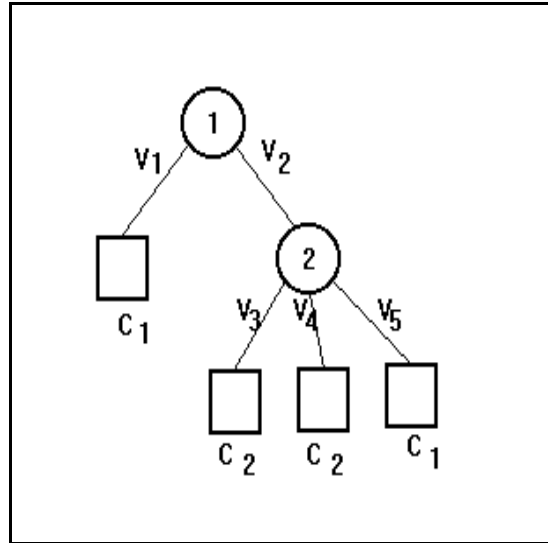


Fig. 3.10: Arbol de decisión no binario

Un árbol de decisión consiste en una estructura árbol con nodos y ramas. Cada nodo interno representa una condición simple y cada nodo terminal u hoja, una clase a asignar. Si el árbol es binario (Fig. 9) las condiciones se evalúan con dos posibles resultados: verdadero o falso; si el árbol no es binario (Fig. 10) las condiciones pueden cubrir distintos valores, por ejemplo, si una característica es alta, media o baja.

El cómo un árbol de decisión produce un conjunto de reglas lógicas, es inmediato; así el ejemplo de la figura 9 daría lugar a dos reglas de la forma:

Si Condición 1 es falsa o si Condición 1 es verdadera y Condición 2 es verdadera entonces Clase 1.

Si Condición 1 es verdadera y Condición 2 es falsa entonces Clase 2.

Por el contrario, el árbol no binario de la figura 10 daría este otro conjunto de reglas:

Si Condición 1 es V_1 o Condición 2 es V_5 entonces Clase 1.

Si Condición 1 es V_2 y Condición 2 es V_3 o V_4 entonces Clase 2.

5.2.2 Construcción de árboles de decisión.

La metodología empleada por Quinlan para la construcción de árboles de decisión está basada en las ideas de [Hunt66] sobre un algoritmo recursivo con técnica de divide y vencerás para la construcción del árbol, pero donde se encuentra la base del problema es en la búsqueda del árbol óptimo entre la infinidad de árboles posibles. Quinlan en el ID3 propuso un criterio que se basa en la siguiente norma de la Teoría de Información: *la información transmitida por un mensaje depende de su probabilidad y puede ser medida en bits como el menos logaritmo en base 2 de esta probabilidad*. El criterio entonces es:

Supóngase un conjunto T de casos clasificados en k clases C_i , que existe un atributo o característica en los casos de T que toma los n posibles valores O_j . Se somete T a un test X de manera que T puede ser partido en n subconjuntos T_j , donde cada T_j contiene todos los casos de T que toman el valor O_j en la característica señalada.

Por otro lado supóngase que S es cualquier conjunto de casos y sea $freq(C_i, S)$ el número de casos de S que pertenecen a la clase C_i y $|S|$ el número de casos en S . Así pues, si se selecciona aleatoriamente un caso de S y se enuncia que pertenece a alguna clase C_i , la probabilidad de este mensaje es

$$\frac{freq(C_i, S)}{|S|}$$

y la información que transmite es

$$-\log_2\left(\frac{freq(C_i, S)}{|S|}\right) \text{ bits}$$

La información esperada para S es la suma sobre las clases en proporción a sus frecuencias en S , dando

$$info(S) = - \sum_{i=1}^k \frac{freq(C_i, S)}{|S|} \times \log_2\left(\frac{freq(C_i, S)}{|S|}\right) \text{ bits}$$

Cuando se aplica al conjunto T , $info(T)$ mide la cantidad media de información necesaria para identificar la clase de un caso en T (también llamada entropía de T). Si se aplica esta medida al conjunto T después de haber sido dividido en los n

subconjuntos T_j y si se denomina $info_X$ a la información esperada mediante este test X , se puede calcular como una suma ponderada sobre todos los subconjuntos:

$$info_X(T) = \sum_{j=1}^n \frac{|T_j|}{|T|} \times info(T_j)$$

La cantidad

$$ganancia(X) = info(T) - info_X(T)$$

mide la información que se gana al dividir T de acuerdo con el test X . El criterio de ganancia entonces, selecciona aquel test que maximiza esta ganancia de información.

Este fue el criterio aplicado para la herramienta ID3, con el que se obtienen buenos resultados, pero presenta una seria deficiencia: Tiene una fuerte predisposición en favor de test sobre características con muchos valores posibles. De esta manera si existiera un atributo con un valor distinto para cada caso, su $info_X$ sería 0, así que la ganancia de información al usar este atributo para particionar el conjunto sería máxima. Sin embargo, desde el punto de vista de la predicción, esta división es completamente inoperante.

En la herramienta C4.5 Quinlan corrigió esta tendencia en el criterio de optimización mediante lo que él denomina información de la división o "split", que es una normalización del criterio anterior, en la que la ganancia atribuible a un test con una característica con muchos valores es ajustada convenientemente. Para ello define un nuevo criterio:

Considera la información potencial que representa dividir T en subconjuntos y la define de manera análoga a $info(S)$:

$$split\ info(X) = - \sum_{j=1}^n \frac{|T_j|}{|T|} \times \log_2 \left(\frac{|T_j|}{|T|} \right)$$

Entonces el cociente entre la ganancia definida anteriormente y la información "split", expresa la proporción de información generada por la división que es útil:

$$ganancia\ proporcional(X) = \frac{ganancia(X)}{split\ info(X)}$$

Este es el criterio que maximiza el programa C4.5, aunque si la división incluye pocos subconjuntos, el divisor puede ser cercano a cero, y por tanto, el cociente sería inestable. Para evitarlo, este criterio selecciona un test que maximice el cociente anterior pero sujeto a la restricción de que el numerador debe ser suficientemente grande, al menos como la ganancia media de todos los test examinados.

Esta regla es fácil de aplicar para el caso de atributos discretos, pero para características continuas existe la dificultad de encontrar los umbrales o puntos de corte para clasificar los subconjuntos T_i . Para ello, Quinlan propone el algoritmo propuesto en [Paterson82]. Básicamente consiste en ordenar todos los valores que para un determinado atributo A presentan los casos del fichero T . Si hay m valores distintos, sólo hay $m-1$ umbrales que dividan T según A; al estar los valores ordenados se puede realizar en una sola pasada un recorrido por todos ellos, escogiéndose aquél que da una ganancia proporcional mayor.

En cuanto a la salida, el C4.5 proporciona una representación gráfica del árbol encontrado, además de una posible simplificación de éste, con un aumento mínimo del error. Asimismo facilita los tamaños en número de nodos de estos árboles y unas tasas de errores aparentes (en el sentido de que son errores sobre el fichero de entrenamiento) para ambos y una tasa de error estimado para el árbol simplificado. Esta última, la calcula como la probabilidad de clasificar mal un caso no visto, suponiendo que en cada hoja se sigue una distribución de probabilidad binomial con parámetros el número de puntos mal clasificados y el número total de puntos, y un determinado nivel de confianza que por defecto se sitúa en el 25%. Esta tasa de error estimada es calificada por Quinlan como muy pesimista, y comenta que es superior a la obtenida mediante validación cruzada.

Los sistemas de aprendizaje basados en árboles de decisión son muy efectivos en la práctica, además de ofrecer una representación simple y clara de los resultados. Sin embargo tienen dos limitaciones importantes:

1. Un árbol con un tamaño fijo no tiene por qué ser el mejor árbol de ese tamaño. La razón está en que cada nodo es dividido una vez, sin posibilidades de "vuelta atrás".
2. Como ya se ha comentado, las regiones de decisión tienen formas rectangulares (en dos dimensiones, en general n -ortodros), lo que dificulta sobremanera encontrar otras fronteras de decisión tan simples como $X > Y$.

5.3 Producción directa de reglas.

5.3.1 Introducción

La producción de reglas directamente es una herramienta potencialmente más poderosa que derivándolas a partir de un árbol de decisión. Las aproximaciones más extendidas para esta estrategia son tres: la herramienta AQ, los algoritmos genéticos y la lógica borrosa. De las tres, las dos últimas son las más novedosas y, por tanto, las más extensamente comentadas en los siguientes apartados.

La técnica AQ [Michalski83] y [Michalski86] básicamente consiste en un algoritmo iterativo que construye inicialmente una regla para cubrir sólo un caso positivo y uno negativo, para continuar mediante heurísticas especiales, que se pueden adaptar a criterios de aprendizaje específicos, hasta construir una regla completa y consistente. Esta aproximación sigue conceptualmente las ideas de una metodología inductiva, dado que el conocimiento alcanzado puede ser general o especializado según convenga. El algoritmo en sí mismo sólo usa los operadores lógicos de negación, unión e intersección para procesar las descripciones.

5.3.2 Aproximación mediante algoritmos genéticos.

Desde principios de los ochenta, el interés por aplicar la metodología de optimización de los algoritmos genéticos en aprendizaje ha ido creciendo. El principal problema de tal aplicación es encontrar una representación adecuada, para captar las características del problema y representar una solución potencial. Además la representación tradicional de los algoritmos genéticos mediante una longitud fija de las soluciones, es problemática en este caso, por desconocerse a priori el número de reglas óptimo.

Dos enfoques diferentes se han propuesto:

- La Escuela de Michigan utiliza una población de individuos de longitud fija, y donde la solución viene representada por un conjunto de cromosomas o individuos. Esta metodología desarrollada originalmente en [Holland86], consiste básicamente en representar en un cromosoma, una regla que compite con otros individuos. La mejor regla reconoce una serie de casos y activa otras reglas para los casos sobrantes, y así sucesivamente. Este método, por su formulación, es más indicado para planificación que para clasificación.
- La escuela de Pittsburgh representa mediante un individuo o cromosoma de dimensión variable, un conjunto de reglas que compite con otro conjunto de reglas por cubrir todos los casos. Este método fue originalmente propuesto en [Smith80], y es una representación más adecuada para un aprendizaje supervisado, dado que

cada cromosoma proporciona una solución independiente.

Desde estos enfoques iniciales, existen diferentes especificaciones y metodologías; por ejemplo, referidas al enfoque de Pittsburgh estarían, entre otros, la herramienta SAMUEL [Grefenstette91], el sistema GABIL [DeJong93] o el programa GIL [Janikow93]. Estos estudios difieren en la representación de las reglas y en distintos detalles de la función a minimizar e investigan nuevos operadores específicos para el proceso de aprendizaje, pero mantienen en común unas ideas básicas que se exponen a continuación.

Representación interna.

Un conjunto de reglas lógicas se van a representar mediante un cromosoma con una codificación binaria de la siguiente forma:

- Si el atributo es continuo, se divide éste en dominios de valores o intervalos de clase, de forma que si un atributo se divide en cinco intervalos, la representación 10100 indica que el valor se encuentra o en el primer intervalo o en el tercero.
- Si el atributo es discreto, por ejemplo, con cuatro valores posibles, una cadena de cuatro elementos como 1001 indica que el atributo toma el primer o cuarto valor en un determinado orden fijo.
- Una regla es una sucesión de cadenas como las anteriores, una para cada atributo más una codificación en binario para la característica de cada clase.
- El conjunto de reglas que forma el individuo, es una cadena binaria formada por cadenas que representan reglas, donde o se limita el número máximo de reglas o se penaliza en la función objetivo el número de éstas.

Población inicial.

La población inicial puede ser seleccionada, sólo por mecanismos aleatorios (criterio habitual en los algoritmos genéticos), inicializarla a partir de los datos, es decir cada individuo forma una regla, o mediante una inicialización con reglas conocidas a priori como válidas, si esto es posible. En general, los experimentos conocidos proponen combinaciones de todas.

Mecanismo de evaluación.

La función a evaluar debe reflejar un criterio de aprendizaje. En aprendizaje supervisado los criterios normalmente incluyen completitud, consistencia y posiblemente complejidad. La completitud y consistencia de una regla o un conjunto de reglas miden su calidad con respecto a los casos de entrenamiento; así una completitud alta indicaría que una regla cubre de forma positiva bastantes casos del fichero de entrenamiento, mientras que una

consistencia alta indicaría que el número de casos en los que esta regla falla es bajo. En cuanto a la complejidad puede venir medida en función del número de reglas y de la cantidad de condiciones de éstas. Lo habitual es una función a minimizar que sea alguna combinación de estas medidas.

Operadores genéticos.

Además de los operadores clásicos en algoritmos genéticos, es decir, cruces y mutaciones, existe un gran número de posibles operadores específicos para sistemas de reglas. Algunos de los más extendidos serían los siguientes:

- Intercambio de reglas: dos conjuntos de reglas (dos individuos) intercambian dos reglas entre sí (cruzan su información genética).
- Copia de reglas: dos conjuntos de reglas copian una regla entre sí, es decir, se añaden información de otro individuo.
- Añadir regla: si un caso no es cubierto por un conjunto de reglas, se añade una regla que lo contenga.
- Extender una regla: generalizar las condiciones de una regla.
- Omitir una condición o una regla: se simplifica una regla o un conjunto de reglas mediante la eliminación de alguna información genética.
- Generalización de reglas: dadas dos reglas, producir otra que sea una generalización de ellas. Así, si una regla es $[A=0, B=0, C=1]$ y otra $[A=1, C=0, D=0]$ la generalización de ambas es $[A=0, 1, C=0, 1]$.
- Especialización de reglas: dadas dos reglas, producir una nueva que sea la "intersección" de ambas. Así, si una regla es $[A=0, 1, C=0, 1]$ y otra $[A=1, 2, C=0, 2]$, la especialización sería $[A=1, C=0]$.

Parámetros del sistema.

Además de los parámetros habituales, tales como tamaño de la población, número de generaciones, método de selección de los individuos, estrategia de reproducción, etc., es necesario añadir probabilidades a los operadores específicos, peso de la completitud, consistencia y complejidad en la función de evaluación, proporción de población inicial aleatoria, etc. Uno de los principales inconvenientes de este método es precisamente su alta parametrización y la probable dependencia entre ellos.

5.3.3 Aproximación mediante lógica borrosa.

La teoría de los conjuntos borrosos (también llamados difusos) es un área de activa investigación, con un origen fuertemente matemático. Los defensores de esta metodología creen que la lógica borrosa puede proporcionar fundamentos robustos y consistentes para el proceso de información incluyendo el reconocimiento de patrones y la construcción de sistemas que aprenden. Desde este punto de vista, la lógica borrosa juega al menos dos importantes papeles en el reconocimiento de patrones. El primero, sirve de interface entre construcciones realizadas en base a variables lingüísticas (aparentemente preferidas por los humanos) y caracterizaciones cuantitativas apropiadas para las máquinas.

El segundo papel es que permite una correcta interpretación de los conceptos imprecisos o inciertos. Es decir, los conjuntos borrosos proporcionan una herramienta legítima para interpretar algunas distribuciones que nos parecen útiles y que difícilmente se justificarían con la teoría de probabilidad clásica. Estos dos papeles no son distintos, pero existen diferencias importantes.

5.3.3.1 El papel de interface.

Con respecto al papel de interface, una característica o elemento de clasificación será definida como un conjunto borroso. Es decir, es determinada en términos de una función de pertenencia. Esta es definida en el dominio de alguna variable medible y el valor de la función de pertenencia indica hasta que punto un valor de la variable es compatible con el concepto de esa característica. Los escépticos arguyen que, en la práctica, esto no es muy diferente de asignar una probabilidad subjetiva a los valores de una característica, a lo que los partidarios de la lógica borrosa responden diciendo que los valores de la función de pertenencia deben ser interpretados más en términos de distribución de posibilidad que de probabilidad.

Así, el concepto de que un parámetro sea "razonablemente alto", es algo impreciso y debería poder utilizarse como tal, no sólo porque es una forma de representación del conocimiento humano, sino también porque es con tales tipos de conceptos con los que los humanos razonan. En la teoría de los conjuntos borrosos se utilizan conceptos como "razonablemente alto" como una cantidad imprecisa y el parámetro correspondiente es definido como una variable en un dominio de medida determinista. Para ello se proporciona una interface entre las dos vistas definiendo "razonablemente alto" como un conjunto borroso en el dominio de un parámetro con el uso de una función de pertenencia.

De esta forma si X es una colección de objetos x , un conjunto borroso A en X se define un conjunto de pares ordenados:

$$A = \{(x, \mu_A(x)) / x \in X\}$$

La entidad $\mu^A(x)$ es llamada la función de pertenencia, el valor que representa el grado de

pertenencia de x en A . Usualmente la función de pertenencia se representa normalizada, es decir,

$$\mu_A: X \rightarrow [0,1]$$

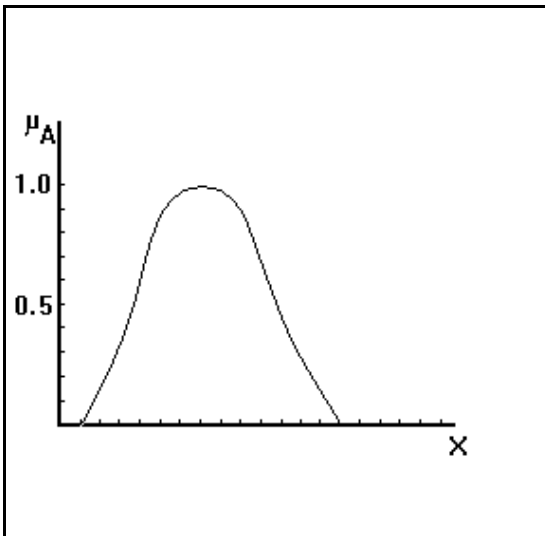


Fig. 3.11: Función de pertenencia

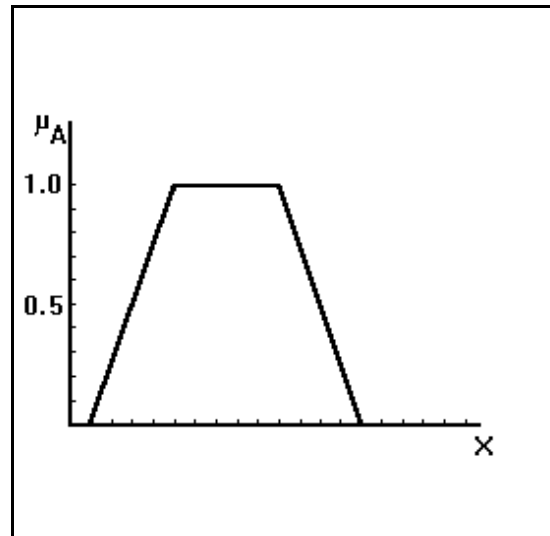


Fig. 3.12: Conjunto borroso mediante trapecio

A partir de estas definiciones existen diversas notaciones posibles para un conjunto borroso, aunque la más utilizada es la que lo representa en términos de una función de valor continuo, como se puede ver en la figura 11. Para simplificar la función de pertenencia a menudo se dibuja mediante trazos lineales, bien como un triángulo o como en la figura 12, donde el conjunto borroso se representa por un trapecioide definido por cuatro puntos y las correspondientes rectas entre éstos que caracterizan la función de pertenencia.

5.3.3.2 El papel de la distribución de posibilidad.

En los sistemas tradicionales de clasificación o de reconocimiento de patrones, se debe decidir sobre si un elemento estaba en la clase i o en la clase j . Según se ha visto, esto comúnmente consiste en usar una función de decisión $g_i(x)$ para cada clase i , decidiendo que x es de la clase i si y solo si $g_i(x) > g_j(x) \forall j \neq i$.

La lógica borrosa introduce la noción de posibilidad para modificar el concepto de pertenencia a una clase. La idea no es preguntar si x pertenece a la clase i sino preguntar cuál es el grado de pertenencia que x tiene a cada clase i , para todo i . La pertenencia a una clase se convierte en un conjunto borroso sobre el dominio de todos los posibles patrones.

El valor de la función de pertenencia es numéricamente igual al valor de una función de distribución de posibilidad que expresa la *posibilidad* de que el caso x pueda ser incluido en la clase i .

Zadeh en [Zadeh78] recomienda el punto de vista de la distribución de posibilidad y discute las diferencias con el concepto de probabilidad. En algunos aspectos, la posibilidad constituye una cota superior de la probabilidad: lo que es imposible es también improbable, pero lo que es posible no siempre es probable.

Hay también circunstancias en las que los dos papeles son esencialmente indistinguibles. Por ejemplo, si una clasificación es primordialmente el problema de estimar la posibilidad de pertenecer a varias clases, entonces el énfasis está en el segundo papel. Sin embargo, si cada una de las clases es definida como un conjunto borroso, entonces el primer papel es también implicado.

La extensión de las operaciones y conceptos matemáticos a los conjuntos borrosos se debe, principalmente, a Zadeh en [Zadeh75] y a Dubois y Prade [Dubois80]. Sin embargo, existe una abundante literatura donde se definen conceptos y operaciones con conjuntos borrosos, aunque según [Pao89] no esté claro que los resultados obtenidos sean siempre útiles.

5.3.3.3 Uso en reconocimiento de patrones.

Existen al menos tres circunstancias en las que los conceptos y técnicas de los conjuntos borrosos son útiles en el reconocimiento de patrones:

- 1) Si algunas de las características de la población a clasificar son simbólicas o no numéricas, un conjunto borroso puede servir de interface entre esta característica lingüística y medidas cuantitativas.
- 2) La utilización de la función de pertenencia como discriminador de si un elemento pertenece o no a una clase (definida como conjunto borroso), es algo más ambigua que los clasificadores clásicos, pero la clasificación borrosa, al asignar a cada caso un valor de pertenencia (posibilidad) para cada clase, proporciona más información que una simple asignación a una determinada clase.
- 3) Si se tienen casos con información incompleta, la función de pertenencia puede proporcionar un estimador de este conocimiento incompleto.

En esta memoria se ha hecho uso de las dos primeras situaciones, pues dadas las características de la investigación no existe la posibilidad de información perdida.

5.3.3.4 Clustering Borroso.

Los algoritmos clásicos de clustering generan una partición de la población de forma que

cada caso es asignado a exactamente un cluster. El algoritmo ISODATA [Ball66] es uno de los más usados, y fue modificado en [Dunn74] y generalizado en [Bezdek81] hasta convertirlo en un algoritmo de clustering borroso de propósito general, denominado algoritmo de las C-Medias Borrosas (conocido por sus siglas en inglés: FCM).

Básicamente, este algoritmo intenta conseguir clasificar n elementos $x^k \in X$ con $1 \leq k \leq n$, con p características cada uno, es decir, $X \subset \mathbb{R}^p$, en c cluster borrosos, asignando una función de pertenencia u_{ik} :

$$u_{ik} : X \rightarrow [0,1] \forall i, 1 \leq i \leq c \forall k, 1 \leq k \leq n$$

para ello, trata de minimizar la función J_m definida:

$$J_m(U, P : X) = \sum_{k=1}^n \sum_{i=1}^c (u_{ik})^m D_{ik,A}^2$$

$m > 1$ es un exponente de peso sobre cada pertenencia borrosa.

U es la matriz (u_{ik}) .

$P = (v_1, v_2, \dots, v_c)$ son c vectores de \mathbb{R}^p , que son los centroides de los clusters.

A es una matriz $p \times p$ definida positiva.

$D_{ik,A}$ es la distancia asociada a la norma A de las x^k a v_i :

$$D_{ik,A} = \|x_k - v_i\|_A = \sqrt{(x_k - v_i)^t A (x_k - v_i)}$$

Las c -particiones del conjunto X obtenidas por la aplicación de un algoritmo de clustering se caracterizan como conjuntos de $c \times n$ valores $\{u_{ik}\}$ que satisfacen las restricciones siguientes:

$$1) 0 \leq u_{ik} \leq 1 \quad 1 \leq i \leq c, 1 \leq k \leq n$$

$$2) 0 < \sum_{k=1}^n u_{ik} < n \quad 1 \leq i \leq c$$

$$3) \sum_{i=1}^c u_{ik} = 1 \quad 1 \leq k \leq n$$

Las condiciones necesarias para minimizar aproximadamente J_m son conocidas:

$$u_{ik} = \left[\sum_{j=1}^c \left(\frac{\|x_k - v_j\|_A}{\|x_k - v_i\|_A} \right)^{\frac{2}{m-1}} \right]^{-1} \quad \forall i, k$$

$$v_i = \frac{\sum_{k=1}^n (u_{ik})^m x_k}{\sum_{k=1}^n (u_{ik})^m} \quad \forall i$$

5.3.3.5 Modelo Borroso.

El algoritmo FCM anterior fue utilizado en [Sugeno93] para construir un modelo borroso en base a reglas de la forma:

$$R^i : \text{si } x \in A^i \text{ entonces } y \in B^i \quad (38)$$

donde $x = (x_1, x_2, \dots, x_n) \in \mathbb{R}^n$ son las características de entrada, $A = (A_1, A_2, \dots, A_n)$ son n conjuntos borrosos, $y \in \mathbb{R}$ es la variable de salida y B es un conjunto borroso para esta variable.

La metodología propuesta consiste básicamente en aplicar la partición borrosa FCM a la variable de salida. Una vez obtenida una partición del espacio de salida en clusters borrosos se realiza una proyección de estos clusters sobre el espacio de entrada obteniendo un conjunto borroso en \mathbb{R}^n , que proyectado sobre cada eje asigna a cada característica un conjunto borroso según la ecuación anterior (39).

A partir de este modelo inicial, se realiza un refinamiento intentando minimizar la suma del cuadrado de los errores que se cometen al intentar estimar el valor de la variable de salida con el valor propuesto por el modelo. Este refinamiento se produce por un ajuste de los parámetros que definen los conjuntos borrosos trapezoidales para cada parámetro según un procedimiento secuencial que actúa repetidamente sobre cada parámetro de cada cluster borroso de cada variable de entrada, modificándolo en una cantidad constante y estudiando si el error se minimiza, en cuyo caso el parámetro queda modificado, obteniéndose así un modelo borroso a partir de la base de datos.

Este modelo borroso, una vez optimizado, puede proporcionar un modelo cualitativo mediante una aproximación lingüística. Para ello ajusta a los conjuntos borrosos obtenidos, unos conjuntos borrosos predefinidos en base a términos, modificadores (adjetivos) y conectivos (conjunciones) lingüísticos.

6. Referencias.

Ball, G.H. y D.J. Hall, "ISODATA, an iterative method of multivariate data analysis and pattern classification", Proceedings of the IEEE International Communications Conference, Philadelphia, 1966.

Baum E.B. y D. Haussler, "What size net gives valid generalization ?", Neural Computation, vol. 1, pp. 151-160, MIT Press, 1989.

Bayley, T. y A.K. Jain, "A note on distance-weighted k-nearest neighbour rules", IEEE Transactions on Systems, Man and Cybernetics, Vol. SMC-8, pp.311-313, 1978.

Bezdek, J.C., "Pattern Recognition with Fuzzy Objective Function Algorithms", Plenum Press, Nueva York, 1981.

Blum A. y R.L. Rivest, "Training a 3-node neural network is NP-complete", Proceedings of Computational Learning Theory, pp. 9-18, Morgan Kaufman, 1988.

Breiman, L., J. Friedman, R. Olshen, C. Stone, "Classification and Regression Trees", Ed. Wadsworth, 1984.

Cover, T.M. y P.E. Hart, "Nearest Neighbor Pattern Classification", IEEE Transactions on Information Theory, Vol. IT-13 pp. 21-27, 1967.

Dasarathy, B.V. (Ed.), "Nearest Neighbor(NN) Norms: NN pattern classification techniques", IEEE Computer Society Press, 1991.

De Jong, K.A., W.M. Spears y D.F. Gordon, "Using Genetic Algorithms for Concept Learning", Machine Learning, 13, pp. 161-188, 1993.

Dubois, D. y H. Prade, "Fuzzy Sets and Systems: Theory and Applications", Academic Press, New York, 1980.

Dudani, S.A. "The Distance-Weighted k-Nearest-Neighbor Rule", IEEE Transactions on Systems, Man and Cybernetics, Vol. SMC-6, n° 4, pp. 325-327, 1976.

- Dunn, J., "A Fuzzy Relative of the Isodata Process and its use in Detecting Compact Well Separated Clusters", *Journal of Cybernetics*, Vol. 3, pp. 32-57, 1974.
- Efron, B., "Estimating the Error Rate of a Prediction Rule", *Journal of the American Statistical Association*, n° 78, pp. 316-333, 1983.
- Fisher, R., "The use of multiple measurements in Taxonomic problems", *Annals of Eugenics*, n° 7, pp. 179-188, 1936.
- Fix, E. y J.L. Hodges, "Discriminatory Analysis: Nonparametric Discrimination: Consistency Properties", Project 21-49-004, Report n° 4, pp. 261-279, USAF School of Aviation Medicine, Texas, 1951.
- Fix, E. y J.L. Hodges, "Discriminatory Analysis: Nonparametric Discrimination: Small Sample Performance" Project 21-49-004, Report n° 11, pp. 280-322, USAF School of Aviation Medicine, Texas, 1952.
- Friedman, J., J. Bentley, R. Finkel, "An Algorithm for Finding Best Matches in Logarithmic Expected Time", *ACM Transactions on Mathematical Software*, Vol. 3, n° 3, pp. 209-226, 1977.
- Funahashi K., "On the approximate realization of continuous mappings by neural networks", *Neural Networks*, Vol. 2, n° 3, pp. 183-192, Pergamon, 1989.
- 1991.
- Grefenstette, J., "Lamarckian learning in multi-agent environments", *Proceedings of the Fourth International Conference on Genetic Algorithms*, pp 303-310, Morgan Kaufmann,
- Holland, J. "Escaping brittleness: The possibilities of general-purpose learning algorithms applied to parallel rule-based systems", *Machine Learning: An Artificial intelligence approach*, Vol. 2, Eds. Michalski, Carbonell y Mitchel, San Mateo, Morgan Kaufmann, 1986.
- Hunt, E.B., J. Marin y P.J. Stone, "Experiments in induction", Academic Press, 1966.
- Hush, D.R. y B. Horne "An overview of neural networks. Part I: static networks" *Informática y Automática*, Vol. 25, N° 1, pp 19-36, 1992.
- Hush, D. R., "Classification with neural networks: a performance analysis", *Proceedings of IEEE International Conference on Systems Engineering*, pp. 277-280, 1989.
- Janikow, C.Z., "A knowledge-Intensive Genetic Algorithm for Supervised Learning", *Machine Learning*, n° 13, pp. 189-228, 1993.

Kohonen, T. "Self-organized formation of topologically correct feature maps", *Biological Cybernetics*, vol. 43, pp. 59-69, 1982.

Lachenbruch, P. y M. Mickey, "Estimation of Error Rates in Discriminant Analysis", *Technometrics*, n° 10, pp. 1-11, 1968.

Leonard, J.A. y M.A. Kramer, "Radial Basis Functions for classifying process faults", *IEEE Control Systems*, Abril 1991.

Leonard, J.A., M.A. Kramer y L.H. Ungar, "Using Radial Basis Functions to approximate a Function and its Error Bounds", *IEEE Transactions on Neural Networks*, Vol. 3, n° 4, pp 624-627, 1992.

Macleod, J.E., A. Luk y D.M. Titterington, "A Re-Examination of the Distance-Weighted k-Nearest Neighbor Classification Rule", *IEEE Transactions on Systems, Man and Cybernetics*, Vol. SMC-17, n° 4, pp. 689-696, 1987.

McCulloch, W.S. y W. Pitts, "A logical calculus of ideas immanent in nervous activity", *Bulletin of Mathematical Biophysics* n° 5, pp. 115-133, 1943.

Michalski, R.S., "Theory and methodology of inductive learning", *Machine Learning: An Artificial intelligence approach*, Vol. 1, Eds. Michalski, Carbonell y Mitchel, San Mateo, Morgan Kaufmann, 1983.

Michalski, R.S., I. Mozetic, J. Hong y N. Lavrac, "The AQ15 inductive learning system: An overview and experiments", *Technical Report UIUCDCS-R-86-1260*, Department of Computer Science, University of Illinois, 1986.

Moody, J. y C. Darken, "Learning with localized receptive fields", *Proceedings of the 1988 Connectionist Models Summer School of Pittsburg*, eds. D. Touretzky, G. Hinton, y T. Sejnowski, pp. 133-143, Morgan Kaufmann, 1988.

"Neural Computing, a technology handbook for NeuralWorks Explorer", NeuralWare, Inc. 1993.

Pao, Y.H. "Adaptative Pattern Recognition and Neural Networks", Addison-Wesley, 1989.

Patrick, E.A. y F.P. Fischer III, "A Generalized k-Nearest Neighbor Rule", *Information and Control*, Vol. 16, n° 2, pp. 128-152, Academic Press, 1970.

Plaut, D., S. Nowlan y G. Hinton, "Experiments on Learning by Back Propagation". *Technical Report CMU-CS-86-126*, Department of Computer Science, Carnegie Mellon

University, Pittsburgh, 1986.

Quinlan, J., "Induction of Decision Trees", *Machine Learning*, n° 1, pp.81-106, 1986.

Quinlan, J., "C4.5: Programs for Machine Learning", Morgan Kaufmann Publisher, Inc., 1993.

Rosenblatt, F. "The perceptron: A probabilistic model for information storage and organization in the brain", *Psychological Review*, vol. 65, pp. 386-408, 1958.

Rumelhart, D.E., G.E. Hinton y R.J. Williams, "Learning internal representations by error propagation", in *Parallel Distributed Processing: Exploration in the Microstructure of Cognition*, ed. D.E. Rumelhart y J.L. McClelland, vol. 1, pp 318-362. MIT Press, 1986.

Smith, S.F., " A learning system based on genetic algorithms", Tesis Doctoral, Department of Computer Science. Universidad de Pittsburgh, 1980.

Sugeno, M. y T. Yasukawa, "A Fuzzy-Logic Based Approach to Qualitative Modeling", *IEEE Transactions on Fuzzy Systems*, Vol 1, N° 1, pp 7-31, 1993.

Weiss, S.M. y C.A. Kulikowski, "Computer Systems That Learn", Morgan Kaufmann Publishers, Inc., San Francisco, 1991.

Zadeh, L.A. "Fuzzy Sets as a Basis for a Theory of Possibility", *Fuzzy Sets and Systems*, Vol. 1, pp3-28, 1978.

Zadeh, L.A. "The concept of a linguistic variable and its application to approximate reasoning". Parts 1, 2, y 3, *Information Sciences*, Vol. 8, pp. 199-249, pp. 310-357; Vol. 9, pp. 43-80, 1975.