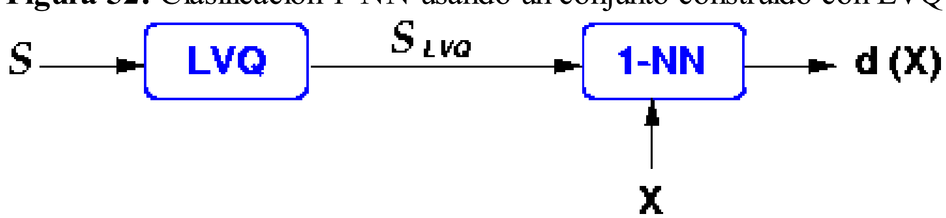
Subsections

6.2 Aprendizaje por cuantificación vectorial (LVQ)

El aprendizaje por cuantificación vectorial (LVQ) supone una extensión del aprendizaje competitivo donde los prototipos están **etiquetados**. Ahora, además de considerar la cercanía de un prototipo se puede evaluar la clase de éste e imponer, por lo tanto, correcciones de premio (acercamiento) o castigo (alejamiento).

El objetivo final puede resumirse como sigue. Dado S, el conjunto de entrenamiento original, se trata de construir un conjunto de referencia, S_{LVQ}, mediante la técnica LVQ. Una vez construido, los patrones a clasificar se etiquetarán usando la regla 1-NN tomando como referencia al conjunto S_{LVQ} (figura 52).

Figura 52: Clasificación 1-NN usando un conjunto construido con LVQ



Así, el objetivo del aprendizaje adaptativo con la técnica LVQ es la construcción del conjunto de referencia S_{LVQ}. Este proceso puede descomponerse en dos pasos:

1. Inicialización del conjunto de prototipos.
2. Aprendizaje (corrección de los prototipos).

que deberán concretarse para cada una de las técnicas particulares de aprendizaje adaptativo que vamos a estudiar. Para concretar más, podemos plantear la expresión de un algoritmo genérico de aprendizaje adaptativo que sirva de marco general a las técnicas cocretas que estudiaremos posteriormente. Este algoritmo genérico producirá como resultado un conjunto de prototipos, S_{LVQ} y recibirá como entradas:

- $S \in A$. Conjunto de entrenamiento a partir del cual se seleccionarán N_{p} prototipos y se corregirán, tras r pasos, dando como resultado S_{LVQ}. En general, $S_{\mathrm{LVQ}} \not\subseteq S$
- $N_{\mathrm{p}} \in N$. Número de prototipos que tendrá S_{LVQ} ($|S_{\mathrm{LVQ}}| = N_{\mathrm{p}}$)
- $r \in N$. Número de pasos de aprendizaje. En cada paso, t, se procesa una única muestra de aprendizaje, $X(t)$, que modifica el conjunto de prototipos en ese paso, $S_{\mathrm{LVQ}}(t)$ para $t = 0, 1,..., r - 1$

Aprendizaje adaptativo (S, N_p, r)

<u>Entradas</u>

$S \in A$	Conjunto de entrenamiento original.
$N_p \in \mathbb{N}$	Numero de prot. del cto. de referencia.
$r \in \mathbb{N}$	Numero de pasos de aprendizaje.

<u>Salidas</u>

$S_{LVQ} \in A$	Conjunto de referencia resultante.

<u>Auxiliares</u>

$Z(t) \in S$	Un prototipo del conjunto S.

<u>Algoritmo</u>

$S_{LVQ}(0) \leftarrow$ **Inicializar** (S, N_p) $\{$*Inicializacion*$\}$
$t \leftarrow 0$
Repetir $\{$ *Aprendizaje* $\}$
    $Z(t) \leftarrow$ **Extraer** (S)
    $S_{LVQ}(t+1) \leftarrow$ **Corregir** $(Z(t), S_{LVQ}(t))$
    $t \leftarrow t + 1$
Hasta $((t = r)$ O $($*convergencia*$))$
$S_{LVQ} \leftarrow S_{LVQ}(t-1)$

El proceso se estabiliza con un número suficiente (elevado) de muestras: cuando $|S|$ es pequeño, éste se vuelve a visitar de forma cíclica o seleccionando aleatoriamente nuevas muestras. Vamos a profundizar algo más sobre este algoritmo comentando **Inicializar**(), **Extraer**() y **Corregir**().

- $S_{LVQ}(0) \leftarrow$ **Inicializar** (S, N_p)

  Esta función selecciona *apropiadamente* un subconjunto de N_p prototipos de S. Tras la inicialización, $S_{LVQ}(0) \subseteq S$ y $|S_{LVQ}(0)| = N_p$.

  La manera en que se inicializa en conjunto $S_{LVQ}(0)$ depende del método de corrección que se va a aplicar posteriormente. Puede ser meramente alatorio o imponiendo ciertas restricciones para asegurar que los prototipos seleccionados para una clase estén ``dentro'' del agrupamiento.

- $Z(t) \leftarrow$ **Extraer** (S)

  Selecciona un prototipo en el paso t, $Z(t)$, del conjunto de aprendizaje S. Su patrón asociado es $X(t)$, que se calcula evaluando la función **Patron** $(Z(t))$.

- $S_{LVQ}(t+1) \leftarrow$ **Corregir** $(Z(t), S_{LVQ}(t))$

  Modifica el conjunto actual de prototipos, $S_{LVQ}(t)$, a partir del prototipo $Z(t)$, $t = 0, 1,..., r - 1$

  Al utilizar prototipos etiquetados para el aprendizaje, las correcciones aplicables son de **premio** o **castigo**.

  La forma particular de esta función es la que va a diferenciar los métodos concretos de aprendizaje adaptativo por cuantificación vectorial.

A continuación estudiaremos con más detalle los pasos de inicialización y aprendizaje, que dan lugar al conjunto de referencia S_{LVQ}.

6.2.1 Inicialización

Consiste en la construcción de $S_{LVQ}(0)$, o sea, se trata de determinar el valor inicial de los N_p prototipos que constituirán $S_{LVQ}(0)$ a partir de S. Sobre este punto debemos aclarar dos cuestiones:

1. Aunque conocemos N_p, el número total de prototipos de referencia de $S_{LVQ}(0)$, ¿Cual será el número de prototipos por clase, N_{pi}?

   Hay dos alternativas que pueden adoptarse:

   (a) Que N_{pi} sea proporcional a N_i. Esto implica que se consideran diferentes probabilidades a priori, problema que se ha discutido anteriormente.

(b)

Que N_{pi} sea el mismo para todas las clases. Implica considerar que todas las clases tienen iguales probabilidades a priori.

2.

Una vez establecidos los valores de N_{pi}, ¿Cómo se seleccionan los prototipos de $S_{LVQ}(0)$?

El procedimiento recomendado es la selección de prototipos con ciertas restricciones que aseguran que éstos se encuentran dentro del agrupamiento correspondiente a su clase. Se puede asegurar que un prototipo está dentro de un agrupamiento si su clasificación mediante la regla k-NN, (con k = 5 ó k = 7, por ejemplo) es correcta. En otro caso, ese prototipo estará cerca de la frontera de decisión o inmerso en otro agrupamiento (en principio, los conjuntos no tienen porqué estar editados). Esta estrategia de selección acelera la convergencia del aprendizaje. Así, el procedimiento empleado es el siguiente:

(a)

Para cada clase, se procesan *secuencialmente* sus prototipos.

(b)

Se añaden a $S_{LVQ}(0)$ si la clasificación k-NN es correcta.

En cualquier caso, una conclusión importante y que debe tenerse en cuenta es que las clases se caracterizan a partir de un número fijo y relativamente bajo de prototipos.

6.2.2 Aprendizaje

El aprendizaje, propiamente dicho, consiste en actualizar de forma iterativa los prototipos del conjunto de referencia actual, $S_{LVQ}(t)$ $t = 0, 1,..., r - 1$, en r pasos, usando un total de r prototipos, $Z(t)$ $t = 0, 1,..., r - 1$ de S.

Obviamente, se establece una sucesión de conjuntos de aprendizaje:

$$S_{LVQ}(0), S_{LVQ}(1), ..., S_{LVQ}(r - 1) = S_{LVQ}$$

y aunque $S_{LVQ}(0) \subseteq S$, usualmente (en la práctica, nunca) $S_{LVQ} \not\subseteq S$. Además, $|S_{LVQ}| = |S_{LVQ}(0)| = N_p$ ya que tan solo se actúa sobre los valores de los prototipos de $S_{LVQ}(t)$ mediante las correcciones oportunas, no sobre el número de éstos.

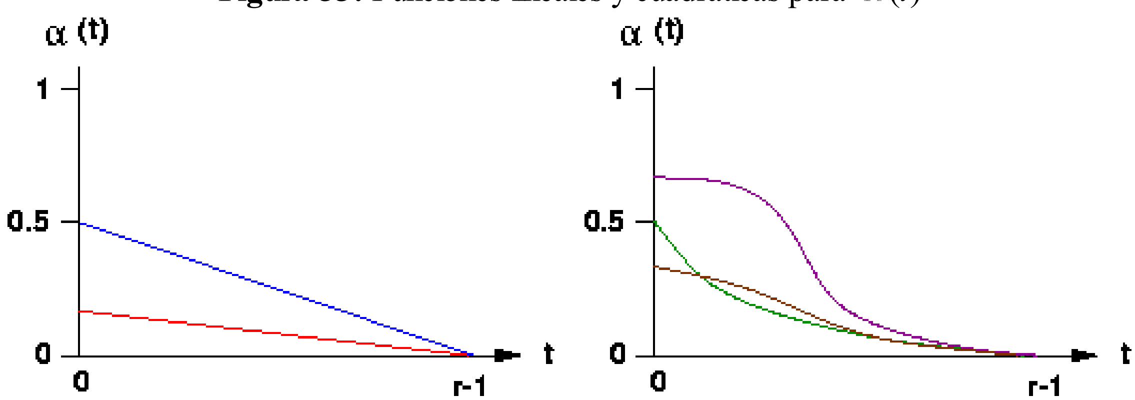
El valor del parámetro de entrada r debe ser *suficientemente grande* aunque el valor adecuado es desconocido a priori. No obstante, el proceso de aprendizaje puede estabilizarse antes por lo que es posible establecer una condición de convergencia adicional, como hemos indicado en el algoritmo. En nuestro estudio, por simplicidad, no utilizaremos ningún criterio de convergencia adicional.

La expresión concreta de la función **Corregir**() determina el método concreto de aprendizaje. Así se habla de los métodos LVQ-1, OLVQ-1, LVQ-2, DSM, etc. cuando en realidad son expresiones concretas de esta función de corrección.

Cualquier función de corrección utiliza la llamada ***función de ganancia*** o ***razón de aprendizaje***, definida como $\alpha : N \rightarrow R$ (una interpretación de esta función se adelantó en este ejemplo). Una vez fijada la forma funcional de $\alpha()$ tan solo hay que especificar el valor inicial, $\alpha(0)$, valor que debe proporcionarse como un parámetro adicional.

La única restricción sobre esta función es que debe proporcionar una secuencia monótona *decreciente* de coeficientes escalares $\alpha(t)$ que cumpla $0 < \alpha(t) < 1$. En la figura 53 mostramos dos funciones válidas de este tipo. El primer caso es una función lineal y el segundo una función cuadrática de t. Por motivos computacionales suele utilizarse la función lineal. En ambos casos, la secuencia de valores $\alpha(t)$ está determinada por el valor inicial, $\alpha(0)$, ya que para $t = r - 1$ el valor de $\alpha(t)$ debe ser muy cercano a cero. Para funciones lineales, el decrecimiento será más rápido cuanto mayor sea el valor de $\alpha(0)$.

Figura 53: Funciones lineales y cuadráticas para $\alpha(t)$



Finalmente, y a modo de adelanto, la *eficacia* conseguida en la clasificación y el *tiempo* requerido por el aprendizaje dependerán de dos factores:

1. El número de prototipos asignados a cada clase y sus valores iniciales.
2. El algoritmo de aprendizaje seleccionado y en particular, los valores de los parámetros utilizados.

A continuación estudiaremos con detalle los métodos de corrección LVQ.

6.2.3 Método LVQ-1

1. **Inicialización**.

   El único parámetro requerido es N_p y la inicialización se realiza con el procedimiento habitual descrito anteriormente: Dado N_p, calcular N_{pi} según el tipo de a priori y para cada clase, seleccionar N_{pi}

prototipos de los N_i disponibles usando la regla k-NN.

2.

Aprendizaje:

Los parámetros del aprendizaje son: r y $\alpha(0)$. Modifica únicamente el prototipo más cercano al patrón de aprendizaje del paso t, $X(t)$, asociado al prototipo de aprendizaje $Z(t)$. La corrección empleada es de premio o castigo y se realiza de acuerdo con las reglas:

$$m_c(t+1) \leftarrow m_c(t) + \alpha(t)[X(t) - m_c(t)] \quad \{Premio\} \tag{37}$$
$$m_c(t+1) \leftarrow m_c(t) - \alpha(t)[X(t) - m_c(t)] \quad \{Castigo\} \tag{38}$$

que se interpretan:

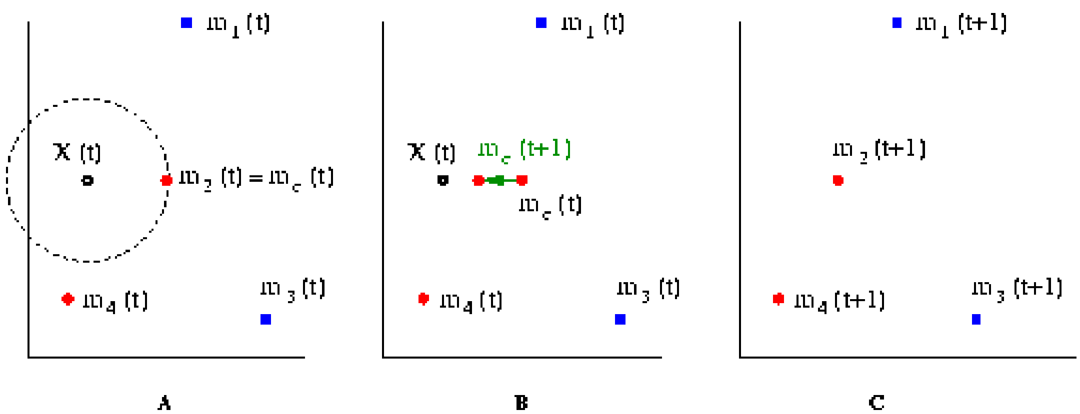
- **Premio**: Si la clase del prototipo de referencia más cercano, $m_c(t)$, coincide con la del patrón de aprendizaje, $X(t)$, entonces $m_c(t)$ se acerca a $X(t)$.
- **Castigo**: En otro caso, $m_c(t)$ se aleja de $X(t)$.

En cualquier caso, la dirección de la corrección está determinada por el vector $X(t) - m_c(t)$ y el valor del desplazamiento depende de $\alpha(t)$.

Ejemplo

En la figura 54 mostramos gráficamente una corrección de premio.

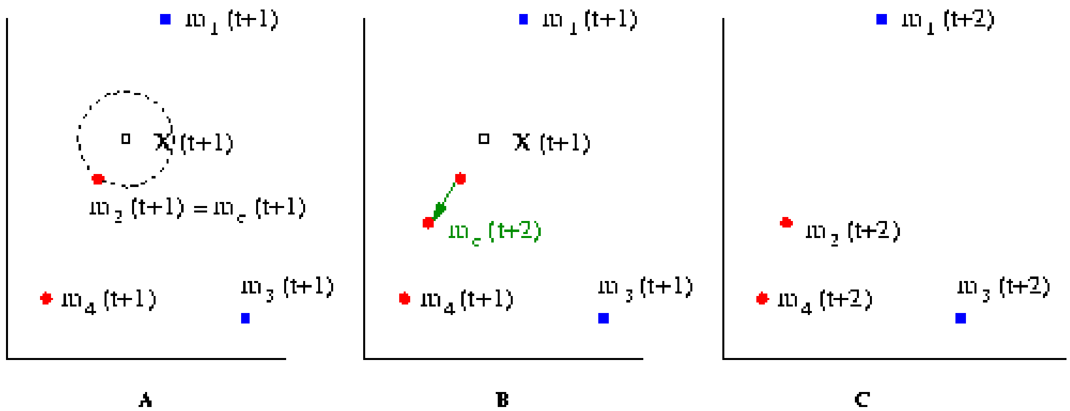
Figura 54: Corrección tipo *premio* al prototipo $m_2(t)$. A) Situación inicial. B) Corrección sobre $m_2(t)$. C) Situación final.



En A mostramos la situación inicial en el instante t: $m_2(t)$ es el prototipo más cercano a $X(t)$ y es de su misma clase. En B indicamos la corrección efectuada sobre $m_2(t)$, acercándolo a $X(t)$. En C mostramos la situación final, esperando el prototipo del paso $t + 1$.

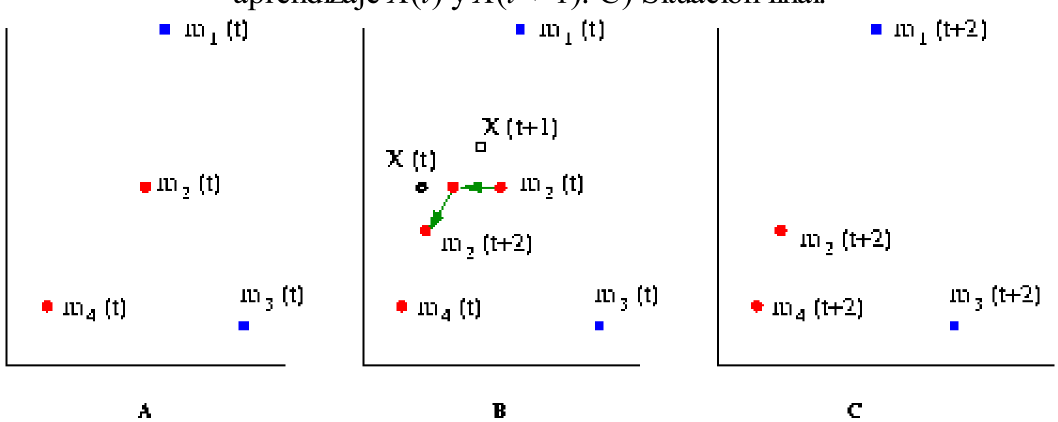
En la figura 55 mostramos gráficamente una corrección de castigo. En A mostramos la situación inicial en el instante $t + 1$: $m_2(t)$ es el prototipo más cercano a $X(t + 1)$, de diferente clase. En B indicamos la corrección efectuada sobre $m_2(t)$, alejándolo de $X(t + 1)$. En C mostramos la situación final, esperando el prototipo del paso $t + 2$.

Figura 55: Corrección tipo *castigo* al prototipo $m_2(t + 1)$. A) Situación inicial. B) Corrección sobre $m_2(t)$. C) Situación final.



En la figura 56 resumimos los pasos de premio y castigo sobre el prototipo m_2 ilustrados en las figuras 54 y 55.

Figura 56: Resumen de las correciones efectuadas sobre el prototipo $m_2(t)$. A) Situación inicial en el instante t. B) Correcciones forzadas por los patrones de aprendizaje $X(t)$ y $X(t + 1)$. C) Situación final.



Con estas consideraciones, la función **Corregir**() empleada en el algoritmo genérico de aprendizaje adaptativo puede particularizarse en la función **Corrección LVQ-1**(). Esta función puede expresarse como sigue:

Corrección LVQ-1 $(Z(t), S_{LVQ-1}(t))$

Entradas

$Z(t) \in S$	Un prototipo de aprendizaje de S.
$S_{LVQ-1}(t) \in A$	Conjunto de prototipos en el paso t.

Salidas

$S_{LVQ-1}(t+1) \in A$	Cto. de prototipos corregido para el paso $t+1$.

Auxiliares

$m_c(t) \in S_{LVQ-1}(t)$	Un prototipo de $S_{LVQ-1}(t)$
$X(t) \in P$	Patron de aprendizaje asociado a $Z(t)$.

Algoritmo

$X(t) \leftarrow$ **Patron** $(Z(t))$

$m_c(t) \leftarrow$ **1-NN** $(X(t), S_{LVQ-1}(t))$ { *El prototipo mas cercano* }

Si (**Clase** $(m_c(t))$ = **Clase** $(Z(t))$) **entonces**

$m_c(t+1) \leftarrow m_c(t) + \alpha(t)[X(t) - m_c(t)]$ { *premio* }

Si-no

$m_c(t+1) \leftarrow m_c(t) - \alpha(t)[X(t) - m_c(t)]$ { *castigo* }

Fin-si

Devolver $(S_{LVQ-1}(t+1))$

donde **1-NN** (*X*, *S*) devuelve el prototipo más cercano al patrón *X* tomando como referencia el conjunto *S*.

La conclusión más importante que debemos tener en cuenta sobre el aprendizaje por LVQ-1 es que este algoritmo tiende a mover los prototipos hacia prototipos de aprendizaje de su misma clase y a alejarlos de los de otra clase. De esta manera, *aproxima las funciones de densidad de probabilidad de las clases*, o, reciprocamente, reduce la densidad de los prototipos alrededor de las fronteras de decisión entre clases.

Sobre los valores adecuados de los parámetros que gobiernan el aprendizaje debemos hacer unas consideraciones:

1. La función $\alpha()$ puede ser constante o decrecer monótonamente con t (una función lineal es la mejor opción). Es recomendable fijar un valor peqeño para $\alpha(0)$, bastante menor que 0.1 (0.02 ó 0.03).

2. En cuanto al número de pasos de aprendizaje es suficiente con presentar un número de prototipos $50 \times N_p < r < 200 \times N_p$.

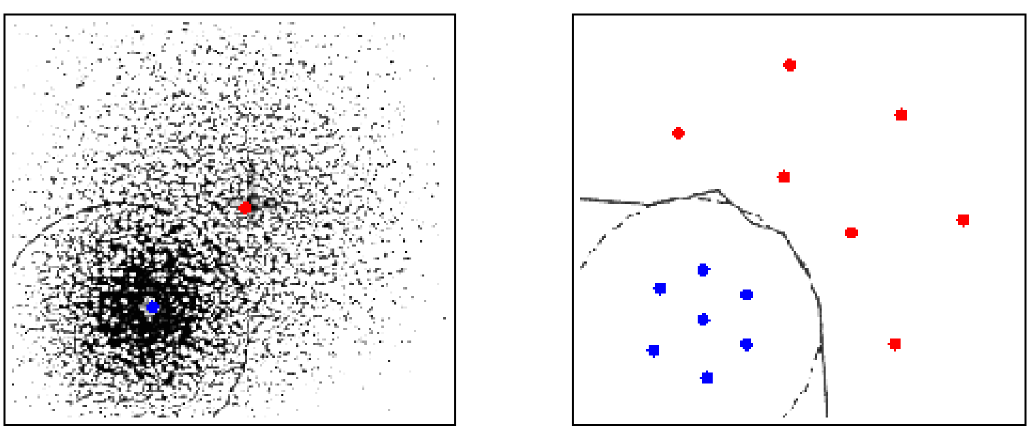
   Si se han utilizado todos los prototipos de una clase y no se ha terminado el aprendizaje *se vuelve a visitar de nuevo*, presentando de forma aleatoria o cíclica los prototipos de esa clase.

   Es importante destacar que *no es tan importante el valor de r si el conjunto de prototipos inicial es de buena calidad (por ejemplo, si el conjunto está previamente editado).*

Ejemplo

En este ejemplo mostraremos como LVQ-1 aproxima las funciones de densidad de probabilidad de dos clases utilizando un conjunto reducido de prototipos. En la figura 57.A mostramos un conjunto de prototipos pertenecientes a dos clases solapadas que siguen una distribución normal. En la figura 57.B mostramos el resultado del aprendizaje por LVQ-1 usando 7 prototipos por clase.

Figura 57: A) Conjunto original de prototipos, S. B) Conjunto de prototipos S_{LVQ-1} construido a partir del anterior (7 prototipos por clase).



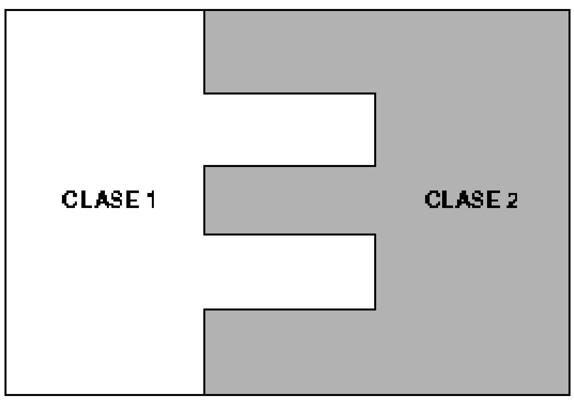
A **B**

En la figura 57.B la línea discontinua indica la frontera de Bayes y la línea continua la frontera inducida por el clasificador 1-NN usando como conjunto de referencia el obtenido por LVQ-1. La aplicación de la regla 1-NN usando $S_{\mathrm{LVQ-1}}$ aproxima las funciones de densidad de probabilidad de las dos clases y de manera *indirecta*, las fronteras de decisión entre las clases.

Ejemplo

En este ejemplo utilizamos un conjunto de prototipos asociado a dos clases (sin solapamiento) que siguen una distribución uniforme. En la figura 58 mostramos las regiones de decisión asocoadas a estas clases.

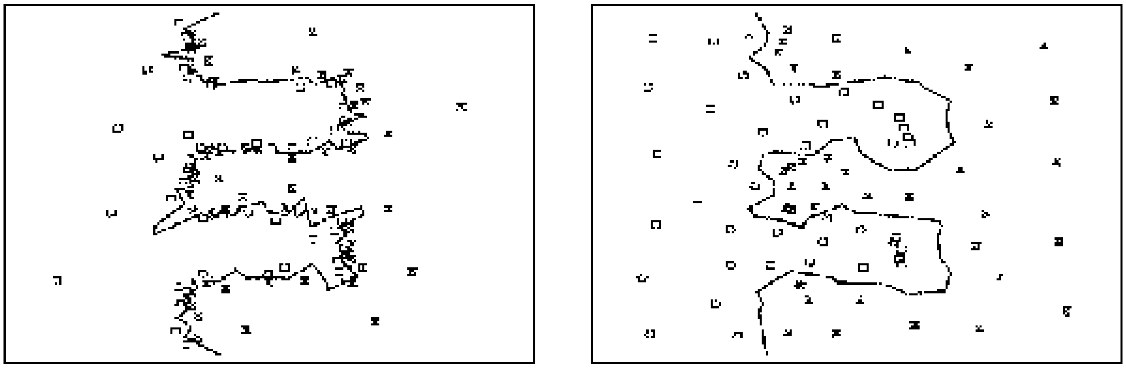
Figura 58: Dos clases con distribución uniforme por clase



Una vez obtenido un conjunto de prototipos de ambas clases, S, se procedió a construir S_{MC} (multiedición + Hart) por una parte, y a construir $S_{\mathrm{LVQ-1}}$, por otra parte.

En la figura 59.A mostramos los prototipos de S_{MC}y en la figura 59.B mostramos los prototipos de $S_{\mathrm{LVQ-1}}$. En ambos casos se muestra la frontera de decisión inducida por el clasificador 1-NN.

Figura 59: a) Condensado de Hart. b) LVQ-1



A **B**

6.2.4 Método LVQ-1 Optimizado (OLVQ-1)

Como su nombre indica, se trata de una mejora sobre el aprendizaje con LVQ-1. La diferencia radica en que cada prototipo del conjunto de referencia tiene su propia razón de aprendizaje, por lo que la función común $\alpha(t)$ de LVQ-1 se sustituye por N_p funciones $\alpha_c(t)$ para $c = 1, 2,..., N_p$.

1. **Inicialización**.

   Se utiliza el método estándar, como en LVQ-1.

2. **Aprendizaje**:

   El procedimiento es similar al del LVQ-1, en cada paso se modifica el valor del *prototipo más cercano* mediante una correción de premio o castigo. Como hemos adelantado, cada prototipo del conjunto de referencia tiene su propia razón de aprendizaje:

$$\alpha_c(t) \leftarrow \frac{\alpha_c(t-1)}{1 \pm \alpha_c(t-1)} \tag{39}$$

   donde el signo del denominador es:

   - Positivo si (**Clase** $(m_c(t))$ = **Clase** $(Z(t))$).

     Este signo hace que en la expresión 39, $\alpha_c(t)$ *decrece* cuando $Z(t)$ se clasifica correctamente. En los prototipos situados en los *centros* de los agrupamientos su razón de apendizaje *decrece rápidamente*. En posteriores correcciones no se modifican perceptiblemente.

- Negativo si (**Clase** ($m_c(t)$) $\neq$ **Clase** ($Z(t)$)).

  Este signo hace que en la expresión 39, $\alpha_c(t)$ *crezca* cuando $Z(t)$ se clasifica incorrectamente. En los prototipos cercanos a las *fronteras* su razón de apendizaje *crece rápidamente*, lo que significa que se alejan muy pronto hacia el centro del agrupamiento.

Para evitar un crecimiento ilimitado se establece un máximo (0.3) para los $\alpha_c()$.

Este método de actualización produce una rápida convergencia, basta con fijar $30N_p < r < 50N_p$ (usualmente, $r = 40N_p$).

En este tipo de aprendizaje puede establecerse la siguiente condición de convergencia adicional: si no hay clasificaciones incorrectas en un número fijado de iteraciones, terminar el aprendizaje.

Con estas consideraciones, la función de corrección por OLVQ-1 puede escribirse como sigue:

Corrección OLVQ-1 $(Z(t), S_{OLVQ-1}(t))$

Entradas

$Z(t) \in S$	Un prototipo de aprendizaje de S.
$S_{OLVQ-1}(t) \in A$	Conjunto de prototipos en el paso t.

Salidas

$S_{OLVQ-1}(t+1) \in A$	Cto. de prototipos corregido para $t+1$.

Auxiliares

$m_c(t) \in S_{OLVQ-1}(t)$	Un prototipo de $S_{OLVQ-1}(t)$
$X(t) \in P$	Patron de aprendizaje asociado a $Z(t)$.
$s(t) \in \{-1, +1\}$	Un valor escalar

Algoritmo

$X(t) \leftarrow$ **Patron** $(Z(t))$
$m_c(t) \leftarrow$ **1-NN** $(X(t), S_{OLVQ-1}(t))$ { *Prototipo mas cercano* }

$s(t) \leftarrow (+1 : -1)$? (**Clase** $(m_c(t)) ==$ **Clase** $(Z(t))$)
$\alpha_c(t) \leftarrow \frac{\alpha_c(t-1)}{1+s(t)\alpha_c(t-1)}$

Si (**Clase** $(m_c(t)) =$ **Clase** $(Z(t))$) entonces
$m_c(t+1) \leftarrow m_c(t) + \alpha_c(t)[X(t) - m_c(t)]$ { *premio* }
Si-no
$m_c(t+1) \leftarrow m_c(t) - \alpha_c(t)[X(t) - m_c(t)]$ { *castigo* }
Fin-si
Devolver $(S_{OLVQ-1}(t+1))$

6.2.5 Métodos LVQ-2.1 y LVQ-3

Otras versiones propuestas del aprendizaje LVQ son los llamados métodos LVQ-2.1 y LVQ-3. La principal diferencia respecto a LVQ-1 es que modifican más de un prototipo de referencia en cada paso de aprendizaje. Sin embargo, requieren más parámetros que especificar. A continuación describiremos brevemente la fase de aprendizaje de estos métodos, ya que la inicialización es la usual.

LVQ-2.1

Para cada prototipo de aprendizaje, $Z(t)$, se modifican **dos** prototipos, $m_i(t)$ y $m_j(t)$, si verifican:

1.

Son los dos prototipos más cercanos tales que uno de ellos es de la misma clase que $Z(t)$ y el otro no.

2.
$Z(t)$ está situado en una ``ventana'' situada alrededor del punto medio entre $m_i(t)$ y $m_j(t)$. $X(t)$, el patrón asociado a $Z(t)$, está dentro de esa ventana si

$$\min\left(\frac{\delta(m_i(t), X(t))}{\delta(m_j(t), X(t))} , \frac{\delta(m_j(t), X(t))}{\delta(m_i(t), X(t))} \right) > \frac{1-w}{1+w} \tag{40}$$

donde w es el ``ancho'' relativo de la ventana (usualmente 0.2 ó 0.3).

Dado un patrón de aprendizaje, $Z(t)$, si se cumplen las condiciones:

1.
$m_i(t)$ y $m_j(t)$ son los dos prototipos más cercanos a $Z(t)$,
2.
$Z(t)$ está dentro de la ventana definida por $m_i(t)$ y $m_j(t)$,
3.
$m_j(t)$ es de su misma clase, y,
4.
$m_i(t)$ es de diferente clase,

la corrección LVQ-2.1 se aplica de la siguiente manera:

$$m_j(t+1) \leftarrow m_j(t) + \alpha(t)[X(t) - m_j(t)] \quad \textit{\{Premio a } m_j(t)\textit{\}} \tag{41}$$
$$m_i(t+1) \leftarrow m_i(t) - \alpha(t)[X(t) - m_i(t)] \quad \textit{\{Castigo a } m_i(t)\textit{\}} \tag{42}$$

Esta estrategia de corrección tiene un serio problema: se ``desestabiliza'' para valores altos de r ya que aleja los prototipos y no aproxima las funciones de densidad de probabilidad. Se soluciona usando valores bajos para $\alpha(0)$ (0.02, por ejemplo) y para r ($30N_p < r < 200N_p$).

LVQ-3

LVQ-3 es un híbrido entre LVQ-1 y LVQ-2.1. La estrategia de corrección es la siguiente: Dado un prototipo de aprendizaje, $Z(t)$, sean $m_i(t)$ y $m_j(t)$ los dos prototipos más cercanos a $Z(t)$,

1.
Si uno de ellos es de la misma clase que $Z(t)$ y el otro no, y $Z(t)$ está en la ventana, entonces se aplica LVQ-2.1.

2.

Si los dos son de la misma clase que $Z(t)$, se premia a ambos:

$$m_i(t+1) \leftarrow m_i(t) + \epsilon\ \alpha(t)[X(t) - m_i(t)] \quad (43)$$

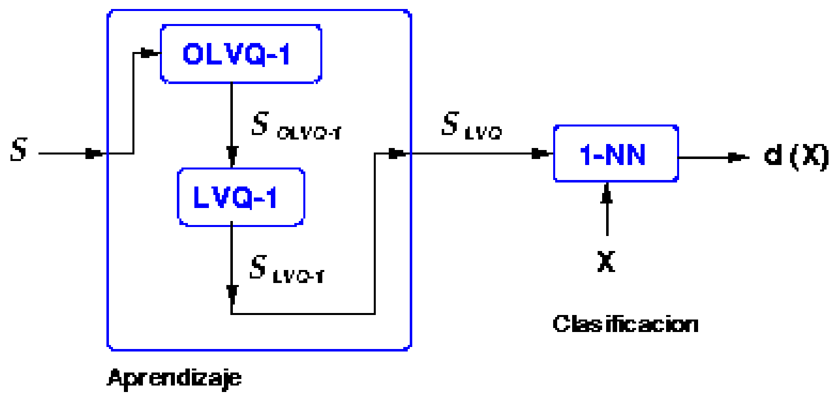
$$m_j(t+1) \leftarrow m_j(t) + \epsilon\ \alpha(t)[X(t) - m_j(t)] \quad (44)$$

donde ϵ toma valores entre 0.1 y 0.5.

6.2.6 Conclusiones

Parece difícil establecer a priori qué estrategia de correción LVQ es la más apropiada para un problema dado. Diversos autores, no obstante, apuntan que los resultados obtenidos por las diversas estrategias LVQ son muy similares. Así, lo más sensato es emplear, como regla general, la estrategia que requiere el ajuste de menos parámetros: LVQ-1.

Figura 60: Esquema propuesto para el aprendizaje por LVQ



A modo de conclusión, considerando que el objetivo final es la clasificación 1-NN que usa como referencia el conjunto resultante del aprendizaje por LVQ, se propone la siguiente metodología (ver [E.3]) esquematizada en la figura 60: *Aplicar inicialmente OLVQ-1 hasta su convergencia y LVQ-1 (preferiblemente) ó LVQ-3 sobre el conjunto de prototipos resultante, con un valor de r moderado.*