Indice

Introducción

Planteamiento

Los procesos de clasificación están presentes en la mayoría de las actividades humanas. En su definición más amplia, el término *clasificación* cubre cualquier contexto en el que se toma una decisión o se realiza un pronóstico en base a información disponible. Desde este punto de vista, un *procedimiento de clasificación* es un método formal que nos permite tomar estas decisiones en nuevas situaciones.

Es evidente la utilidad del desarrollo de sistemas que realicen de forma autónoma estos procesos de clasificación o bien se puedan utilizar como herramientas de ayuda al usuario en procesos de decisión. El diseño de un sistema de clasificación que alcance ambos objetivos implica la obtención no sólo de un alto nivel de rendimiento, sino también de otras dimensiones como robustez, versalitidad, facilidad de interpretación y modificación, y coherencia con el conocimiento previo. Además, tal sistema debe ser capaz de modelar y manejar toda la información a su alcance, incluyendo información incompleta, imprecisa o con incertidumbre, presente con frecuencia en problemas reales de clasificación.

La Teoría de Subconjuntos Difusos [Zad65] nos permite manejar imprecisión y tratar el conocimiento humano de una forma sistemática. En el campo de la clasificación, el papel fundamental de los conjuntos difusos es hacer transparentes los esquemas opacos de clasificación que utilizan normalmente los seres humanos a través del desarrollo de un marco formal implementable en un ordenador [Zad77].

Si a la utilización de la Lógica Difusa unimos el diseño de un sistema basado en reglas, obtendremos un Sistema de Clasificación Basado en Reglas Difusas. Este tipo de sistemas está formado por un conjunto de modelos simples locales, verbalmente interpretables y con un rango de aplicación muy amplio. Se han aplicado con éxito a problemas de clasificación y representan, como el resto de los Sistemas Basados en Reglas Difusas, un esfuerzo de reconciliación entre la precisión de las técnicas tradicionales de la ingeniería y la interpretabilidad de la Inteligencia Artificial.

Los Sistemas Basados en Reglas Difusas permiten la incorporación de toda la información disponible en el modelado de sistemas, tanto de la que proviene de expertos humanos que expresan su conocimiento sobre el sistema en lenguaje natural, como de la que tiene su origen en medidas empíricas y modelos matemáticos. Éste es un aspecto fundamental en la era de la información en la que nos movemos, en la que el conocimiento humano y su

representación e interpretación en los sistemas a desarrollar es cada vez más importante.

En definitiva, el uso de la Lógica Difusa hace posible el tratamiento de información con incerdumbre, muy común en problemas reales de clasificación, y permite el procesamiento de forma eficaz de la información experta disponible. Por otra parte, las reglas difusas permiten representar el conocimiento de una forma comprensible para los usuarios del sistema.

Estos motivos justifican la elección de las reglas difusas como herramienta de representación del conocimiento en el proceso de extracción del conocimiento que presentamos. Uno de los aspectos más importantes en este proceso es la *generación de las reglas difusas*, que implica la determinación de la partición difusa más adecuada para cada variable considerada y la generación de un conjunto de reglas a partir de estas particiones que contenga el mínimo número de reglas con el nivel más alto de cooperación entre ellas.

Una vez obtenida la Base de Conocimiento del sistema, es decir, el conjunto de reglas difusas y la Base de Datos que contiene la definición de las particiones difusas utilizadas, la clasificación de nuevos ejemplos se realizará mediante el uso de un *Método de Razonamiento Difuso* que combine la información representada en las reglas para tomar una decisión. Parece claro que la precisión alcanzable por el Sistema de Clasificación Basado en Reglas Difusas dependerá tanto de la calidad de las reglas inducidas como del buen comportamiento del Método de Razonamiento Difuso.

En muchos de los problemas de clasificación, la cantidad de variables o atributos considerados puede ser grande por alguno de los siguientes motivos:

- Porque los datos no se hayan obtenido para un fin específico, por lo que características que son necesarias para una tarea serán redundantes o irrelevantes para otra.
- Porque los datos se hayan recogido sin conocimiento claro de las variables relevantes para el problema a resolver.
- Por la necesidad, en determinados problemas, de datos de distintas fuentes de información, que hace que si la cantidad de información de cada una de ellas es moderadamente grande, la unión sea enorme.

En estas situaciones es necesario desarrollar un proceso de *Selección de Características* que nos permita determinar las características relevantes en nuestra aplicación para alcanzar un Sistema de Clasificación Basado en Reglas Difusas con máxima precisión y mínimo esfuerzo de medida y de aprendizaje.

Resumiendo todo lo expuesto en los párrafos anteriores, el diseño de un Sistema de Clasificación Basado en Reglas Difusas a través de un proceso de aprendizaje supervisado implica fundamentalmente tres tareas complementarias entre sí:

- La selección de las variables más informativas para el problema de clasificación a resolver.
- La determinación del Método de Razonamiento Difuso a utilizar en la clasificación de nuevos ejemplos.
- El aprendizaje de un conjunto de reglas difusas.

Estos aspectos de diseño se pueden definir de forma automática afrontándolos como problemas de optimización y búsqueda. Los Algoritmos Evolutivos [Bac96] y particularmente los Algoritmos Genéticos [Hol75, Gol89, Mic96] son técnicas generales de resolución de problemas que han mostrado de forma teórica y práctica tener capacidad para proporcionar una solución válida para problemas que requieren una búsqueda eficiente y eficaz en espacios complejos. Este es el motivo por el cual, aunque los Algoritmos Genéticos no se pueden considerar algoritmos de aprendizaje, constituyen una herramienta adecuada para las diferentes tareas implicadas en el diseño de un Sistema de Clasificación Basado en Reglas Difusas.

Objetivos

El objetivo de esta memoria es presentar un proceso de diseño de Sistemas de Clasificación Basados en Reglas Difusas. Este proceso debe ser lo suficientemente general como para abordar los distintos problemas que se plantean en el proceso de aprendizaje. El resultado final debe ser un Sistema de Clasificación Basado en Reglas Difusas, formado por una base de reglas difusas que represente con la mayor precisión posible la información extraíble de los datos, con un Método de Razonamiento Difuso que aproveche toda la información presente en las mismas, y en el que intervengan las variables relevantes para el problema a resolver.

Este objetivo general se puede descomponer en estos objetivos particulares:

1. *Realizar un análisis de la estructura de un Sistema de Clasificación Basado en Reglas Difusas y sus componentes.* Puesto que el objetivo de esta memoria es el desarrollo de un método de aprendizaje completo que nos permita resolver todos los problemas que se plantean en el diseño de un Sistema de Clasificación Basado en Reglas Difusas, el estudio de sus componentes y su modo de funcionamiento nos permitirá determinar las tareas de diseño a realizar y los aspectos determinantes de la calidad del Sistema de Clasificación Basado en Reglas Difusas finalmente obtenido. Para ello se pondrá especial atención en los siguientes aspectos:
   - Estudiar los componentes de un Sistema de Clasificación Basado en Reglas Difusas y la relación entre ellos.
   - Estudiar la influencia del Método de Razonamiento Difuso utilizado en el proceso de clasificación en el comportamiento del sistema.

- Determinar cuáles son las tareas de diseño que es necesario desarrollar para construir un Sistema de Clasificación Basado en Reglas Difusas y cuál es la mejor manera de llevarlas a cabo.

2. *Desarrollar un modelo general de razonamiento difuso para Sistemas de Clasificación Basados en Reglas Difusas.* Este modelo debe ser lo suficientemente general como para incluir como casos particulares los Métodos de Razonamiento Difusos utilizados hasta el momento en Sistemas de Clasificación Basados en Reglas Difusas. A partir de este modelo, *presentar nuevos Métodos de Razonamiento Difusos* para Sistemas de Clasificación Basados en Reglas Difusas que nos permitan mejorar el nivel de rendimiento de los mismos manteniendo su interpretabilidad.

3. *Diseñar un proceso evolutivo multietapa de aprendizaje de Sistemas de Clasificación Basados en Reglas Difusas.* Este proceso debe permitir la obtención de un conjunto de reglas difusas descriptivas, que tengan un alto nivel de cooperación entre sí y con el Método de Razonamiento Difuso utilizado. Además, el proceso debe resolver uno de los aspectos más determinantes de la calidad final del Sistema de Clasificación Basado en Reglas Difusas como es el ajuste adecuado del conjunto de particiones difusas empleadas para cada una de las variables.

4. *Desarrollar un proceso evolutivo de Selección de Características.* Este proceso debe determinar, para un tamaño dado, el conjunto de variables que aportan mayor precisión al Sistema de Clasificación Basado en Reglas Difusas.

Resumen

Esta memoria está organizada en cuatro capítulos y una sección de conclusiones. A continuación presentamos un pequeño resumen de los mismos.

En el Capítulo 1 introducimos los conceptos básicos de los Sistemas de Clasificación. Definimos los Sistemas de Clasificación Basados en Reglas Difusas, describiendo sus componentes, los distintos tipos de reglas existentes, y analizando el proceso de aprendizaje inductivo de este tipo de sistemas. Se estudian los problemas a resolver en el proceso de diseño de Sistemas de Clasificación Basados en Reglas Difusas y se describen algunos métodos propuestos en la bibliografía especializada. Finalmente, se revisan algunos conceptos de la Computación Evolutiva utilizados en el desarrollo de la memoria.

En el Capítulo 2 nos centramos en el proceso de inferencia en un Sistema de Clasificación Basado en Reglas Difusas. Describimos las formas de desarrollar el proceso de clasificación propuestas en la bibliografía y definimos un modelo de inferencia general donde estas propuestas se incluyen como casos particulares. Dentro de este modelo de inferencia definimos dos tipos de procesos de inferencia y proponemos distintos Métodos de Razonamiento Difusos alternativos incluidos en cada uno de los tipos. Mostramos el comportamiento del metodo de inferencia clásico y las nuevas propuestas sobre tres bases de ejemplos (Iris, Wine y Pima), considerando para ello bases de reglas generadas con

seis métodos de aprendizaje distintos. El aprendizaje de los parámetros del método de inferencia constituye una mejora en el comportamiento del sistema. En este capítulo proponemos dos procesos evolutivos que nos permiten adaptar el Método de Razonamiento Difuso al problema y proceso de aprendizaje utilizado.

En el Capítulo 3 describimos un proceso evolutivo multietapa de aprendizaje para la obtención de Sistemas de Clasificación Basados en Reglas Difusas lingüísticos. El proceso integra el Método de Razonamiento Difuso en la obtención de una Base de Conocimiento que coopere con el método de inferencia utilizado en la clasificación, determina el mejor conjunto de modificadores lingüísticos para los términos de las variables lingüísticas, y ajusta las particiones difusas utilizadas para cada una de las variables consideradas. Al final del capítulo se muestran los resultados obtenidos por el mismo con las bases de ejemplos Iris y Pima, y se comparan con los alcanzados con otros procesos de aprendizaje inductivo.

En el Capítulo 4 se analiza el problema de Selección de Características. Una vez definido el problema y los componentes principales de un algoritmo que lo resuelva, estudiamos distintas propuestas realizadas bajo un enfoque evolutivo y presentamos dos nuevas propuestas evolutivas para el problema de Selección de Características. Los procesos de Selección de Características propuestos son independientes al proceso de extracción de reglas difusas y, por tanto, aplicables como etapa previa a cualquiera de ellos. Mostramos sus resultados para la base de ejemplos Sonar, comparándolos con los alcanzados por otros métodos de Selección de Características conocidos.

Por último, incluimos una Sección de Conclusiones que resume los resultados obtenidos en esta memoria, presenta algunas conclusiones sobre éstos y plantea algunos estudios futuros que se derivan a partir de la misma.

La memoria finaliza con una recopilación bibliográfica que recoge las contribuciones más destacadas en la materia estudiada.

Capítulo 1

Sistemas de Clasificación Basados en Reglas Difusas

En este capítulo haremos una breve revisión de conceptos utilizados a lo largo de la memoria. Comenzaremos repasando brevemente la definición de Sistema de Clasificación y su diseño a través de un proceso de aprendizaje supervisado. Posteriormente estudiaremos los Sistemas de Clasificación Basados en Reglas Difusas y abordaremos en las Secciones 3, 4 y 5 los tres aspectos básicos de su diseño:

- la determinación del Método de Razonamiento Difuso,
- la obtención de la Base de Conocimiento, y
- la Selección de Características.

Finalmente revisaremos brevemente algunos conceptos de la Computación Evolutiva y particularmente de los Algoritmos Genéticos que van a constituir una herramienta básica en nuestras propuestas de diseño de Sistemas de Clasificación Basados en Reglas Difusas.

1.1 Sistemas de Clasificación

1.1.1 Definición

El problema de clasificación, desde el punto de vista del aprendizaje supervisado, consiste establecimiento de una regla de decisión que permita determinar la clase de una nuevo objeto en una de las clases existentes y conocidas, $C_j \in C = \{1, \ldots, M\}$. Cada uno de los objetos $e \in E$, a los que tambien se denomina ejemplos, se describe mediante

un conjunto de observaciones $X(e) = (e_1, \ldots, e_N)$ y a cada una de estas observaciones se le denomina variable, atributo o característica.

El diseño de un *Clasificador* o *Sistema de Clasificación* se puede ver como la búsqueda de una correspondencia

$$D : X(E) \longrightarrow C$$

optimal en el sentido de un cierto criterio $\delta(D)$ que nos determine la bondad del Clasificador. Normalmente el objetivo final es el diseño de Sistema de Clasificación que asigne clases a cualquier punto del espacio de atributos con el mínimo error posible.

En un problema de clasificación se pueden distinguir tres componentes principales [MST94]:

1. La frecuencia relativa con la que ocurren las clases en el conjunto de ejemplos, expresada formalmente como una distribución de probabilidad a priori. Es importante que la técnica que se utilice para diseñar un Sistema de Clasificación, no se derive incondicionalmente de una distribución de probabilidad particular, para que el sistema pueda generalizar adecuadamente ante nuevos ejemplos y cambios en la frecuencia de clases.

2. Un criterio implícito o explícito para separar clases, es decir, una relación entrada-salida subyacente que utilice las características observadas para distinguir los objetos de cada clase.

3. El costo asociado a una clasificación errónea. Supongamos que el costo asociado al error en la clasificación de un objeto de la clase C_i en la clase C_j es c_{ij}. La obtención de un Sistema de Clasificación con mínimo error implica la definición de una forma de combinación del coste de error sobre todos los ejemplos. En la práctica es muy difícil obtener los costos de clasificación errónea. Incluso en situaciones donde hay una diferencia evidente en la penalización o recompensa que debería darse por cometer un error o no, es difícil cuantificar ésta. En esta memoria trabajaremos con problemas en los cuales el costo de clasificación errónea es el mismo para todas las clases y es igual a 0 en el caso en que no se cometa error. De cualquier forma, el desarrollo de la memoria puede extenderse para otros casos, simplemente aportando la forma de combinación de errores.

1.1.2 Diseño de un Sistema de Clasificación a través de un Proceso de Aprendizaje Inductivo Supervisado

Como se ha mencionado, el diseño de un Sistema de Clasificación a través de un *proceso de aprendizaje supervisado* parte de un conjunto de ejemplos descritos mediante un conjunto de variables o características, de los cuales se conoce su clasificación correcta.

De este conjunto de ejemplos de entrenamiento, el proceso de aprendizaje inductivo extrae la información necesaria, descrita en la estructura de representación del conocimiento elegida, para la clasificación de los nuevos ejemplos en alguna de las clases determinadas previamente. Este proceso se describe en la figura 1.1.

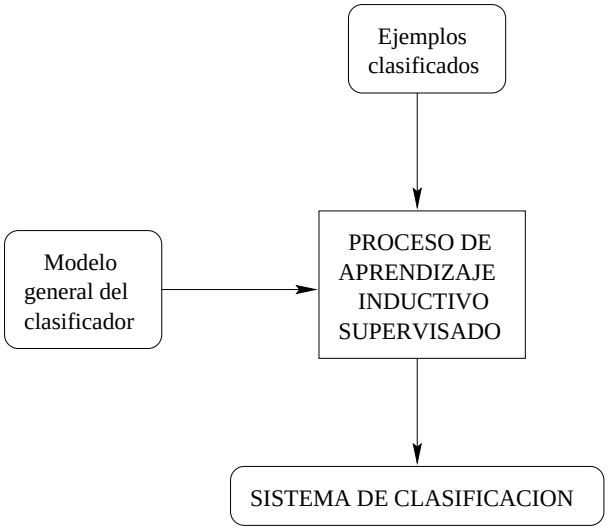

Figura 1.1. Fase de aprendizaje de un Sistema de Clasificación

El *proceso de clasificación* propiamente dicho, se realiza cuando el Sistema de Clasificación recibe un patrón de datos admisible cualquiera con clase desconocida y toma la decisión sobre la clase a la cual pertenece. Este proceso puede verse en la figura 1.2. Habitualmente se realiza una estimación de la capacidad de predicción del Sistema de Clasificación con alguna técnica de estimación de error [WK91] basada en el porcentaje de acierto alcanzado por el Clasificador sobre ejemplos de prueba no utilizados en el proceso de aprendizaje.

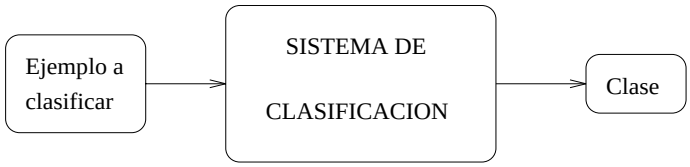

Figura 1.2. Funcionamiento de un Sistema de Clasificación

El proceso de diseño de un Sistema de Clasificación se ha abordado desde distintos puntos de vista que se pueden agrupar en torno a tres enfoques [WK91, MST94]:

- **Estadístico**. Las propuestas estadísticas para el diseño de un Sistema de Clasificación se caracterizan por la utilización de un modelo probabilístico subyacente, que proporciona ante un objeto, más que una clasificación, la probabilidad de pertenecer a cada clase. La mayoría de las técnicas que se utilizan son estadísticas y, por tanto,

hay intervención humana en la selección y transformación de variables, así como en la estructuración global del problema.

Este campo está muy desarrollado y en él se incluyen, entre otras, técnicas como el discriminante lineal, cuadrático, el vecino más cercano, el k-vecinos más cercanos, la regla de Bayes o técnicas basadas en redes causales.

- **Aprendizaje Automático**. Todas las técnicas de aprendizaje automático aplicadas a problemas de clasificación intentan generar expresiones de clasificación a partir de conjuntos de ejemplos, suficientemente simples para ser comprendidas con facilidad por los seres humanos.

  En este enfoque se ha prestado un interés especial al desarrollo de técnicas basadas en árboles de decisión o en reglas como modelo de representación para el conocimiento extraído de los ejemplos. Dentro de los algoritmos que utilizan árboles de decisión, destacan por su importancia la familia de técnicas desarrollada a partir del programa ID3 de Quinlan [Qui86]. ID3 construye un árbol de decisión de forma descendente guiado por una medida de información establecida sobre los ejemplos. Este algoritmo presenta algunos problemas como vulnerabilidad al ruido y a valores desconocidos en los ejemplos, poca capacidad de generalización o incapacidad para trabajar con atributos con dominio contínuo. A partir de él y para solucionar parte de sus limitaciones se desarrolló C4.5 [Qui93], que trabaja con variables contínuas e incrementa tanto la comprensibilidad del árbol obtenido como su precisión. Utilizaremos este algoritmo en algunas de las etapas experimentales de esta memoria para comparar sus resultados con los obtenidos por los distintos procesos propuestos.

- **Redes Neuronales**. En este campo se incluye un grupo amplio de técnicas que, de forma general, obtienen Redes Neuronales consistentes en capas de nodos interconectados, en las que cada nodo produce una función no lineal de su entrada (procedente de otros nodos o directamente de la entrada de datos). Algunos de los nodos constituyen la salida de la red. De esta forma, la red al completo representa un conjunto complejo de interdependencias que puede incorporar cualquier grado de no linealidad y permite el modelado de funciones muy generales.

  La Red Neuronal más sencilla es el perceptrón. En la bibliografía especializada existen distintas propuestas para su diseño, así como el de redes multicapa más complejas [WK91, MST94].

  Los enfoques de las Redes Neuronales combinan la complejidad de algunas técnicas estadísticas con el objetivo del Aprendizaje Automatizado de imitar la inteligencia humana. Su principal inconveniente es su incapacidad para hacer transparentes al usuario los conceptos aprendidos.

Hay que tener en cuenta que los tres enfoques mencionados agrupan técnicas que pueden combinar características de los otros grupos por lo que, en ocasiones, la clasificación en una u otra categoría puede ser difícil.

La característica común a estos enfoques es que las técnicas incluidas en ellos intentan derivar procedimientos de clasificación capaces de:

- Igualar o superar el comportamiento de un experto humano en el tema, pero con la ventaja de una mayor consistencia en los resultados.
- Tener capacidad para manejar una amplia variedad de problemas.
- Poder ser utilizados en entornos prácticos con posibilidad de éxito.

En determinados problemas de clasificación (como problemas de diagnóstico de enfermedades, de reconocimiento de rasgos faciales, o la clasificación de un objeto de una pintura, entre otros) interviene información compleja que los expertos humanos procesan de forma robusta, pero que es difícil representar y procesar en un Sistema de Clasificación. Para diseñar un esquema de clasificación robusto y con un rendimiento e interpretabilidad altos, es necesario el uso de una herramienta adecuada para tratar con este tipo de información presente en la mayoría de los procesos de clasificación.

Los conjuntos difusos se han utilizado ampliamente en el campo de Reconocimiento de Patrones [Ped90, Ped97a], fundamentalmente porque, desde el punto de vista metodológico, la teoría de conjuntos difusos es una teoría adecuada para desarrollar herramientas que modelen procesos cognitivos humanos relativos a aspectos de reconocimiento. En relación con esto, Zadeh señala en [Zad77], que los conjuntos difusos hacen transparentes los esquemas opacos de clasificación utilizados normalmente por los humanos a través del desarrollo de un marco formal implementable en un ordenador. Si a la utilización de la Lógica Difusa unimos el diseño del sistema basado en reglas, obtendremos un Sistema de Clasificación Basado en Reglas Difusas.

El enfoque basado en reglas difusas fue aplicado inicialmente a problemas de control por Mamdani y utilizado posteriormente como una herramienta de modelado por Sugeno entre otros. Este enfoque constituye, en cualquier área, un intento de reconciliación del rigor empírico de las técnicas tradicionales de la ingeniería y la interpretabilidad de las descripciones de la Inteligencia Artificial ([BD95]).

1.2 Sistemas de Clasificación Basados en Reglas Difusas

1.2.1 Definición

Un Sistema de Clasificación Basado en Reglas Difusas (al que denominaremos en lo sucesivo SCBRD) está compuesto por una *Base de Conocimiento* (BC) y un *Método de*

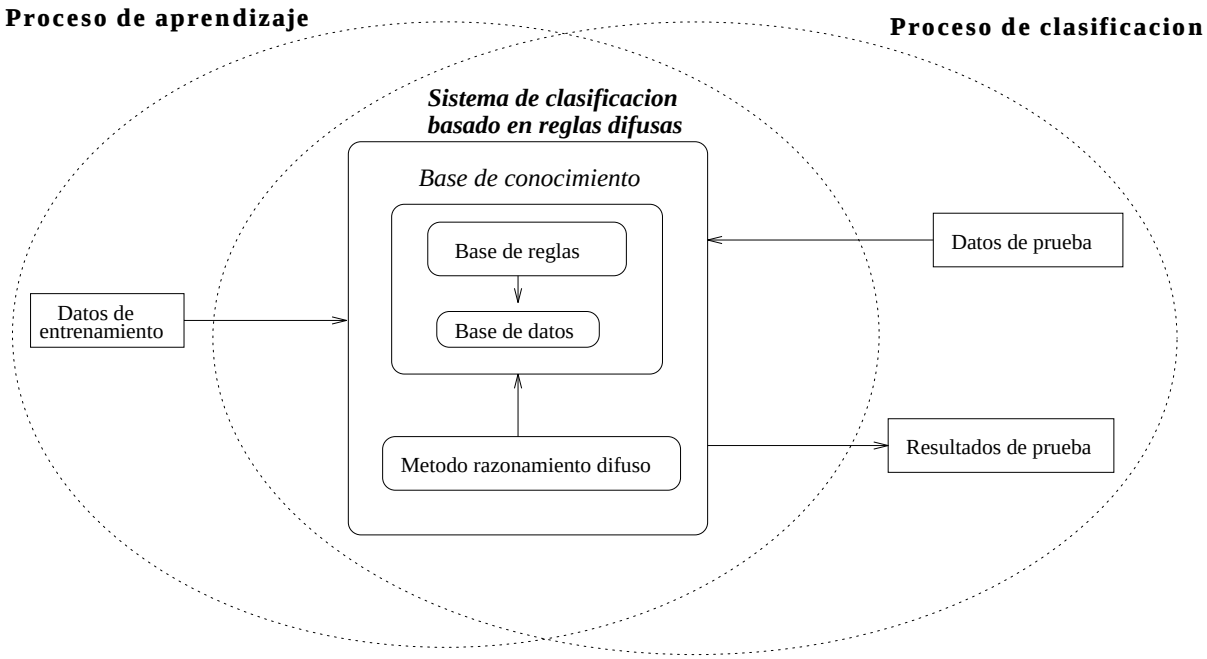

Figura 1.3. Diseño de un SCBRD (aprendizaje y clasificación)

Razonamiento Difuso (MRD) que, utilizando la información de la BC, determina una clase para cualquier patrón de datos admisible que llegue al sistema. Esta estructura se refleja en la figura 1.3.

La BC está formada por dos componentes:

- La *Base de Datos* (BD), que contiene la definición de los conjuntos difusos asociados a los términos lingüísticos utilizados en la Base de Reglas.
- La *Base de Reglas* (BR), formada por un conjunto de reglas de clasificación

$$R = \{R_1, \ldots, R_L\}$$

de uno de los tipos siguientes utilizados en la literatura especializada para SCBRDs:

(a) **Reglas difusas con una clase en el consecuente**

$$R_k: \text{ Si } X_1 \text{ es } A_1^k \text{ y } \ldots \text{ y } X_N \text{ es } A_N^k \text{ entonces } Y \text{ es } C_j$$

donde $X_1, \ldots, X_N$ son las variables asociadas a los diferentes atributos del sistema de clasificación, $A_1^k, \ldots, A_N^k$ son las etiquetas lingüísticas utilizadas para discretizar los dominios contínuos de las variables, e Y es la variable que indica la clase C_j a la cual pertenece el patrón. González y Pérez en [GP98], Kuncheva en [Kun96] y Nauck y Kruse en [NK97] desarrollan SCBRDs con este tipo de reglas.

(b) **Reglas difusas con una clase y un grado de certeza asociado a la clasificación para esa clase en el consecuente**

$$R_k: \text{ Si } X_1 \text{ es } A_1^k \text{ y } \ldots \text{ y } X_N \text{ es } A_N^k \text{ entonces } Y \text{ es } C_j \text{ con grado } r^k$$

donde r^k es el grado de certeza asociado a la clasificación en la clase C_j para los ejemplos pertenecientes al subespacio difuso delimitado por el antecedente de la regla.

Este tipo de reglas son empleadas por Ishibuchi y otros en [INT92, NIT96].

(c) **Reglas difusas con grados de certeza asociados a cada una de las clases en el consecuente**

$$R_k: \text{ Si } X_1 \text{ es } A_1^k \text{ y } \ldots \text{ y } X_N \text{ es } A_N^k \text{ entonces } (r_1^k, \ldots, r_M^k)$$

donde r_j^k es el grado de fuerza de la regla R_k para predecir la clase C_j para un ejemplo perteneciente a la región difusa representada por el antecedente de la regla.

Este tipo de reglas se utilizan en los SCBRDs propuestos por Mandal y otros en [MMP92, PM92].

El último modelo de regla extiende a los dos anteriores empleando distintos valores para los parámetros del vector $(r_1^k, \ldots, r_M^k)$. Así, con

$$r_v^k = 1, \quad r_j^k = 0, \quad j \neq v, \quad j = 1, \ldots, M$$

tenemos una regla difusa tipo (a), y con

$$r_v^k = r^k, \quad r_j^k = 0, \quad j \neq v, \quad j = 1, \ldots, M$$

una regla tipo (b).

En algunas propuestas de SCBRDs se utilizan reglas con consecuente difuso, es decir, reglas difusas en las que el consecuente es una etiqueta lingüística definida a través de un conjunto difuso. Este tipo de reglas, en un Sistema de Clasificación, requieren un proceso de defuzzificación adicional que permita la obtención de un valor nítido de clasificación, y no serán consideradas en esta memoria.

El MRD es un procedimiento de inferencia que utiliza la información de la BC para predecir una clase ante un ejemplo no clasificado. Tradicionalmente, en la literatura especializada [MMP92, INT92, UAL95, CYP96, RG96, AT97, GP98], se ha utilizado el MRD del máximo, también denominado MRD clásico o de la regla ganadora, que considera la clase indicada por una sóla regla, aquella con la que el ejemplo tiene mayor grado de compatibilidad (calculando esté a través de alguna t-norma aplicada al grado de pertenencia del ejemplo a los distintos conjuntos difusos presentes en el antecedente).

La necesidad de interpretabilidad de los resultados exigida a un Sistema de Clasificación en la mayoría de los casos, hace que el enfoque adoptado para su diseño sea con frecuencia el de un *SCBRD descriptivo* como el explicado en esta sección y utilizado en esta memoria. En él, las variables consideradas para describir las observaciones del sistema se consideran variables lingüísticas y se asocia un conjunto de etiquetas a cada una de ellas, almacenándose la semántica de estas últimas en la BD. En un *SCBRD aproximativo* las variables se consideran variables difusas y, en cada regla, cada una de las variables tomará como valor un conjunto difuso distinto por lo que no es necesaria una BD en la BC del sistema, que queda reducida de esta forma a una BR formada por un conjunto de reglas difusas aproximativas.

1.2.2 Aspectos de Diseño de un Sistema de Clasificación Basado en Reglas Difusas

El diseño de un SCBRD mediante un proceso de aprendizaje inductivo supervisado comienza con un conjunto de ejemplos correctamente clasificados y tiene como objetivo final la obtención de un SCBRD que determine, con el mínimo error posible, la etiqueta de clase para ejemplos no clasificados. Una vez diseñado el SCBRD, se determina su rendimiento sobre datos de prueba para tener una estimación del error real del SCBRD. Este proceso se describe en la figura 1.3.

Las principales tareas en el proceso de clasificación propiamente dicho, una vez diseñado el SCBRD, son: extracción del conjunto de características seleccionado en el ejemplo, aplicación del MRD y decisión basada en los resultados aportados por el método de inferencia.

El análisis de los componentes y del funcionamiento de un SCBRD nos permite determinar que el proceso de aprendizaje puede implicar cinco tareas complementarias entre sí:

- La selección de las variables más informativas para el problema de clasificación a resolver,
- la determinación del MRD a utilizar en el proceso de clasificación de nuevos ejemplos,
- la generación de un conjunto de reglas,
- la selección de un subconjunto de reglas con la mejor cooperación y la menor redundancia, y
- el establecimiento y ajuste de las particiones difusas para los dominios de las variables del problema.

Hay que notar en este punto que los tres últimos aspectos de diseño se suelen agrupar en el proceso de obtención de la BC, y que la selección de variables, la simplificación de reglas y el ajuste de particiones difusas son tareas opcionales no necesariamente incluidas en todos los métodos de aprendizaje de SCBRDs.

La definición automática de cualquiera de estas tareas se puede tratar como un problema de optimización o búsqueda. Sabemos que los Algoritmos Evolutivos y, particularmente los Algoritmos Genéticos, son técnicas de búsqueda que han mostrado de forma teórica y empírica capacidad robusta de búsqueda en espacios complejos. Esta es la razón por la que, aunque los Algoritmos Genéticos no se pueden considerar algoritmos de aprendizaje, constituyen una herramienta de optimización robusta, independiente del dominio y aplicable a las distintas tareas que componen el aprendizaje de SCBRDs.

En esta memoria se presentan procesos evolutivos para la resolución de los problemas planteados. En las siguientes subsecciones definiremos de forma más precisa estos problemas y veremos algunas propuestas no evolutivas para los mismos.

1.3 Determinación del Método de Razonamiento Difuso

El MRD en un SCBRD determina la forma de utilización de la información contenida en un conjunto de reglas que describen la relación entre las variables del sistema, en el proceso de predicción de la clase a la que pertenece un nuevo patrón. Es, por tanto, uno de los aspectos más importantes en un SCBRD y más determinantes de su rendimiento e interpretabilidad.

La formulación de un MRD debe permitir aprovechar al máximo la potencia del Razonamiento Aproximado propia de los Sistemas Difusos, que permite obtener un resultado incluso cuando no se tenga compatibilidad exacta (con grado 1) entre el patrón y los antecedentes de las reglas.

Para Sistemas de Control Difuso en los que el consecuente de las reglas es un conjunto difuso, se han formulado distintas versiones de la regla composicional de inferencia de Zadeh [Zad73] que permite obtener la salida de una regla difusa [MZ82, DP92] y distintos operadores de defuzzificación que combinan las salidas de todas las reglas disparadas para obtener una respuesta global del sistema [TT93, CS94, CHP97].

En SCBRDs como los considerados en esta memoria, en los que el consecuente no es un conjunto difuso, el MRD más utilizado es el denominado de compatibilidad máxima o de la regla ganadora. Kuncheva define en [Kun96] de forma general el funcionamiento de este MRD, que también han utilizado otros autores [MMP92, INT92, UAL95, CYP96,

RG96, AT97, GP98]: Para cada regla difusa y para cada clase, se combina el grado de compatibilidad del ejemplo con su antecedente con el grado de certeza de la clasificación en esa clase. La salida del SCBRD es, en base al principio de la regla ganadora, la clase para la que se alcanza valor máximo de combinación entre compatibilidad y certeza para una de las reglas. En caso de empate, se puede seleccionar aleatoriamente una de las clases para las cuales se obtiene ese valor máximo o determinar como salida la clase para la cual se han disparado más reglas (considerando que una regla se dispara en la clasificación de un ejemplo, si éste tiene grado de compatibilidad con el antecedente de la regla mayor que cero).

A diferencia de los empleados habitualmente en control difuso, este MRD utiliza la información que proporciona una sóla regla. Se han desarrollado otras propuestas que tratan de combinar la información aportada por las reglas disparadas:

- El MRD denominado *de combinación aditiva* en [BD95], *de compatibilidad máxima acumulada* en [CYP96] o *basado en la votación máxima* en [IMN96], que acumula para cada clase la combinación del grado de compatibilidad con el grado de certeza aportado por cada regla, y clasifica con la clase para la que se obtiene el valor máximo.

- El MRD *basado en la defuzzificación del centroide* [CWY95, CYP96] que, para una BR con L reglas difusas tipo a), determina la salida S para un ejemplo e de acuerdo a la siguiente expresión

$$S = \frac{\sum_{k=1}^{L} R^k(e) \cdot C_k}{\sum_{k=1}^{L} R^k(e)}$$

  donde C_k es la clase indicada en el consecuente de la regla k y $R^k(e)$ es el grado de compatibilidad del ejemplo con el antecedente de la regla k (o lo que es lo mismo, el grado de pertenencia del ejemplo a la región difusa determinada por el antecedente de la regla).

  Una vez obtenida esta salida S se clasifica con la clase C_j tal que

$$C_j = (int)(S + 0.5)$$

  donde (int) denota el valor entero más pequeño.

  Este tipo de MRD sólo trabaja bien con problemas de clasificación en dos clases y necesita un gran número de reglas difusas para dar resultados satisfactorios en cualquier otro caso, por lo que no es muy utilizado.

Como se ha mencionado, el MRD incluido en un SCBRD determina su rendimiento e interpretabilidad, por lo que nuevas propuestas de MRDs que integren la información de todas las reglas o seleccionen parte de las mismas en un proceso de clasificación pueden llevar a la obtención de mejores SCBRDs. Este problema se estudia en el Capítulo 2 de esta memoria.

1.4 Obtención de la Base de Conocimiento

Un proceso de generación de la BC de un SCBRD descriptivo debe obtener un conjunto de reglas difusas que representen, de la forma más precisa posible, las relaciones entre las variables del problema de clasificación y permitan la clasificación correcta de nuevos ejemplos que se presenten al sistema.

En este problema uno de los primeros aspectos a determinar es la partición difusa utilizada para cada una de las variables. La obtención del conjunto de etiquetas lingüísticas (y conjuntos difusos asociados) óptimo para cada variable es un problema difícil que se ha afrontado de diferentes formas en distintos métodos:

- Estableciendo una partición difusa fija para cada variable en base a conocimiento experto o a una división equidistante del dominio con un tipo de función de pertenencia preestablecida o determinada tras una experimentación. Este enfoque se sigue en la extensión a problemas de clasificación del proceso de generación de Wang y Mendel [CWY95, CYP96], en el método de generación desarrollado por Ishibuchi y otros [INT92], en el proceso de extracción propuesto por Pal y Mandal en [PM92] o en el método genético desarrollado por Yuan y Zhuang [YZ96].

- Utilizando técnicas de agrupamiento que determinan grupos de datos con determinadas relaciones entre las características y construyendo los conjuntos difusos mediante proyecciones. Habitualmente esto determina conjuntos difusos para cada una de las reglas, por lo que se obtienen SCBRDs aproximativos. Algunos métodos dentro de esta línea de trabajo se pueden encontrar en [AL95, UAL95, AT97]. Una vez obtenida la BC, algunos autores proponen métodos de ajuste de las funciones de pertenencia utilizadas [AT97].

- Integrando el proceso de búsqueda de la partición difusa más adecuada en el proceso de aprendizaje, como hacen Hong y Lee en el método descrito en [HL96].

- Utilizando distintas particiones difusas para cada variable, que se diferencian no sólo en el número de términos utilizados sino también en la forma de la función de pertenencia y, posteriormente, seleccionando las reglas con mejor comportamiento en el sistema para obtener así reglas difusas con distinta granularidad en la partición, como proponen Ishibuchi y otros en [INT92, INT94].

Los métodos de extracción de BCs difusas se diferencian no sólo en la forma de obtención de la partición difusa sino también en el enfoque utilizado para la generación de reglas. Así tenemos entre otros:

- Métodos iterativos que

  - exploran el conjunto de ejemplos de entrenamiento, como el propuesto por Chi y otros en [CWY95, CYP96], y determinan la mejor regla para cada ejemplo, o
  - exploran el espacio de patrones y determinan la mejor regla para cada subregión del espacio determinada por la partición difusa, como hacen Ishibuchi y otros en [INT92] y Pal y Mandal en [MMP92].

- Métodos basados en Redes Neuronales que permiten obtener BCs descriptivas [MP96, NK97] o aproximativas [UAL95].
- Métodos basados en técnicas de agrupamiento [AL95, HL96, TA96, AT97].
- Métodos basados en la derivación de reglas difusas a partir de ID3 [CY96].
- Métodos basados en Algoritmos Genéticos, que serán analizados en el Capítulo 3.

En el proceso de obtención de una BR suele ser necesaria una etapa de simplificación de la misma por alguno de los siguientes motivos:

- Por la propia heurística del proceso de generación que tiende a obtener BRs con cardinalidad alta. Un ejemplo claro de este problema está representado en el método de generación de reglas distribuidas de Ishibuchi [INT92], donde se utilizan simultáneamente distintas particiones difusas para cada una de las variables por lo que el número de reglas generadas es enorme.
- Porque se disponga de un número de ejemplos elevado.
- Porque cada uno de los ejemplos se describa mediante un gran número de características o variables.

En cualquiera de estos casos es conveniente reducir el número de reglas del SCBRD resultante para hacer que el sistema sea más interpretable, y más rápido en el proceso de clasificación. Esta simplificación del SCBRD se puede realizar antes o después del proceso de generación de reglas. Los procesos de reducción de la dimensionalidad del problema previos a la generación de reglas intentan alcanzar alguno de estos objetivos [CYP96]:

- Reducir el número de ejemplos de entrenamiento mediante alguna técnica de agrupamiento.
- Reducir el número de características mediante algún proceso de Selección de Características.
- Reducir las particiones difusas de las variables generando las funciones de pertenencia basándose en la distribución de datos de entrenamiento.

Para reducir una BR ya generada se han propuesto métodos de eliminación de reglas soportadas por un número bajo de ejemplos de entrenamiento o con bajo grado de certeza y métodos de combinación de reglas, entre otros. Estos procesos, que se pueden plantear como procesos de búsqueda, se han resuelto con distintas técnicas heurísticas, entre ellas los Algoritmos Genéticos como veremos en el Capítulo 3.

Todos estos problemas se resuelven, total o parcialmente, en un proceso en una etapa o en varias, en los distintos métodos propuestos en la bibliografía. En las siguientes subsecciones, describiremos brevemente cuatro procesos no evolutivos de obtención de BCs, que serán utilizados en algunas de las experimentaciones de esta memoria. En el Capítulo 3 se analizarán las propuestas evolutivas.

1.4.1 Extensión del Algoritmo de Wang y Mendel a Problemas de Clasificación

Wang y Mendel proponen en [WM92] un método de generación de reglas difusas para problemas de control, que Chi y otros extienden en [CWY95, CYP96] a problemas de clasificación de la forma que se describe a continuación.

Consideremos un conjunto de datos de entrenamiento con la siguiente estructura

$$\begin{array}{c} e^1 = (e^1_1, \ldots, e^1_N, s^1), \\ e^2 = (e^2_1, \ldots, e^2_N, s^2), \\ \vdots \\ e^p = (e^p_1, \ldots, e^p_N, s^p) \end{array}$$

siendo s^h la etiqueta de la clase a la que pertenece el ejemplo e^h.

Para generar un conjunto de reglas a partir de este conjunto de entrenamiento que describan las relaciones entre las variables y determinen una correspondencia entre el espacio de características y el conjunto de clases $C = \{C_1, \ldots, C_M\}$, se siguen los siguientes pasos:

1. *Fuzzificar el espacio de características.* Una vez determinado el dominio de variación de cada atributo o característica x_i, se divide en Z_i regiones ($i = 1, \ldots, N$) y se define un conjunto difuso para cada región difusa. En nuestras experimentaciones hemos utilizado conjuntos difusos con funciones de pertenencia triangulares.

2. *Generar una regla difusa para cada ejemplo* $e^h = (e^h_1, \ldots, e^h_N, s^h)$

   2.1 Calcular los grados de pertenencia del ejemplo e^h a las distintas regiones difusas.

   2.2 Asignar el ejemplo e^h a la región difusa a la que tiene mayor grado de pertenencia.

2.3 Generar una regla para el ejemplo, cuyo antecedente está determinado por la región difusa seleccionada y con la etiqueta de clase del ejemplo en el consecuente.

2.4 Si la BR es tipo (b) o tipo (c) calcular el grado de certeza. Para ello se determinará el grado de certeza de la clasificación en una clase C_j como el cociente S_j/S, siendo S_j la suma del grado de pertenencia de los ejemplos de entrenamiento de la clase C_j a la región difusa determinada por el antecedente de la regla, y S la suma del grado de pertenencia a la misma región de todos los ejemplos independientemente de la clase a la que pertenezcan. En [CYP96] se describen otras formas de cálculo del grado de certeza.

1.4.2 Método de Generación de Reglas de Ishibuchi y otros

Ishibuchi y otros proponen en [INT92] un proceso de derivación de reglas difusas a partir de datos numéricos para problemas de clasificación, dividido en dos etapas:

1. Realizar una *partición difusa del espacio de patrones*, e
2. identificar una regla difusa para cada subespacio definido.

La partición difusa para cada una de las variables se debe especificar de forma adecuada ya que determina el rendimiento de las reglas difusas generadas. Para los casos en que no se disponga de conocimiento suficiente para determinar la partición difusa los autores proponen distintas soluciones: el uso de particiones múltiples para cada una de las variables en la construcción de lo que denominan *reglas distribuidas* [INT92], o un proceso de generación de reglas con partición secuencial del espacio de patrones [INT93]. Aquí consideraremos una partición difusa con el mismo número de etiquetas lingüísticas para cada variable.

Para una BR tipo (b), el proceso de generación de reglas determina una regla para cada región difusa del espacio de patrones de la siguiente forma:

1. Se calcula, para cada clase, la suma de los grados de pertenencia de los ejemplos de la clase a la región difusa.
2. Si este valor es igual a cero para todas las clases (es decir, no hay ejemplos en esa zona del espacio), no se genera la regla.
3. En caso contrario, se asocia al consecuente de la regla la clase C_j que tiene un valor mayor de suma de grados de pertenencia.

   En caso de empate para dos o más clases, no se genera la regla.

4. Sea $\beta_1, \ldots, \beta_M$, la suma de los grados de pertenencia para ejemplos de la clase $C_1, \ldots, C_M$ respectivamente. El grado de certeza de la clasificación en la clase C_j, $r_j \in (0, 1]$, será igual a

$$r_j = \frac{\beta_j - \beta}{\sum_{v=1}^{M} \beta_v}$$

con $\beta = \sum_{v=1}^{M} \frac{\beta_v - \beta_j}{M-1}$

El cálculo del grado de pertenencia de un ejemplo a una región difusa se realiza mediante la aplicación de una t-norma sobre los grados de pertenencia de los atributos del ejemplo a los conjuntos difusos (con función de pertenencia triangular) correspondientes. En [INT92] se analiza el comportamiento de la t-norma mínimo y producto, y en la experimentación realizada en esta memoria, se utiliza el mínimo.

1.4.3 Método de Generación de Reglas de Pal y Mandal

Pal y Mandal proponen en [PM92] un proceso de aprendizaje de SCBRDs con BRs formadas por reglas tipo (c) que admite que los valores de los atributos de los ejemplos vengan descritos de forma cualitativa o cuantitativa. El proceso, particularizado para información de entrada cuantitiva, en la que se centra esta memoria, se describe en los siguientes pasos:

1. *Partición difusa del espacio de características.* Cada variable se divide en tres regiones solapadas para representar las propiedades primarias *pequeño*, *mediano* y *alto*, cuya semántica se describe mediante un conjunto difuso con función de pertenencia tipo π. De esta forma el espacio de características se divide en 3^N regiones solapadas.

2. Se genera *una regla difusa por cada subespacio determinado por la partición difusa establecida.* Para una regla difusa tipo (c)

$$R_k : \text{ Si } X_1 \text{ es } A_1^k \text{ y } \ldots \text{ y } X_N \text{ es } A_N^k \text{ entonces } (r_1^k, \ldots, r_M^k)$$

el cálculo del grado de certeza para cada clase, r_j^k con $j = 1, \ldots, M$, viene dado por:

$$r_j^k = \frac{1}{n_j} \sum_{h=1}^{n_j} \left[\frac{1}{N} \sum_{i=1}^{N} \mu_{A_i^k}(e_i^h) \right]$$

siendo $e^h = (e_1^h, \ldots, e_N^h)$ un ejemplo de entrenamiento y n_j el número de ejemplos de entrenamiento de la clase C_j.

En [MMP94] se propone otra forma alternativa de cálculo del grado de certeza.

1.4.4 Método de Generación de Reglas de Hong y Lee

Hong y Lee proponen en [HL96] un proceso de aprendizaje para derivar funciones de pertenencia y reglas difusas de un conjunto de ejemplos de entrenamiento. Es un proceso de aprendizaje que no parte de una partición difusa determinada por el experto, como el proceso de aprendizaje de Ishibuchi y otros o el de Pal y Mandal, sino que determina las funciones de pertenencia en el proceso de generación de reglas. El método, adaptado para un problema de clasificación con clases conocidas, se describe en los siguientes pasos:

1. *Construir las funciones de pertenencia iniciales para las variables de clasificación.* Para ello se analizan, para cada variable, los valores presentes en los datos de entrenamiento y se determina la diferencia mínima entre dos valores (h). Se construye una partición difusa inicial para todas las variables con conjuntos difusos con soporte igual a $2h$. Es decir, si para un problema de clasificación dado, se consideran dos variables de entrada X_1, y X_2 y para la variable X_1 los 5 ejemplos de entrenamiento de que se dispone tienen valores $e_1^1 = 15$, $e_1^2 = 10$, $e_1^3 = 20$, $e_1^4 = 5, e_1^5 = 35$, h tendrá el valor 5, y se definirán conjuntos difusos triangulares definidos por tres puntos (a_i, b_i, c_i), que verifiquen que $b_i - a_i = c_i - b_i = h$.

2. *Construir la tabla de decisión inicial.* Se define una tabla de decisión multidimensional (su dimensión vendrá dada por el número de variables que tenga el problema) en la que se indicará la clase a la que pertenece el ejemplo para las celdas que representen zonas en las que exista algún dato de entrenamiento. Las celdas restantes permanecerán vacías.

3. *Simplificar la tabla de decisión*, para eliminar celdas redundantes o innecesarias. Para ello se definen cinco operaciones aplicables a columnas o filas de la matriz de decisión. Por simplicidad de la descripción, estas operaciones se definen para un problema de clasificación con dos variables.

   3.1 Si las celdas en dos columnas son iguales, unir las dos columnas.

   3.2 Si dos columnas son iguales en algunas de sus celdas y las celdas en que se diferencian estan vacías en la otra columna, unir las dos columnas.

   3.3 Si tenemos tres columnas de forma que dos son iguales y la tercera está formada por celdas vacías, unir las tres columnas en una igual a las dos iguales.

   3.4 Si todas las celdas de una columna están vacías y las dos columnas adyacentes son iguales o se pueden unir con las operaciones anteriores, unir las tres columnas.

   3.5 Las zonas (conjuntos de columnas) de la tabla que estén vacías se reparten entre las zonas adyacentes no vacías.

4. *Reconstruir las funciones de pertenencia iniciales* de acuerdo a la nueva composición de la tabla de decisión. Las modificaciones de las funciones de pertenencia correspondientes a las operaciones del paso anterior son:

4.1 Operaciones 3.1 y 3.2. Si (a_i, b_i, c_i) y (a_j, b_j, c_j) son los puntos que determinan las funciones de pertenencia para las etiquetas lingüísticas d_i y d_{i+1} del i-ésimo atributo, la función de pertenencia del conjunto difuso resultado de la unión de columnas viene dada por $(a_i, (b_i + b_j)/2, c_j)$.

4.2 Operaciones 3.3 y 3.4. Si las funciones de pertenencia de las etiquetas d_{i-1}, d_i y d_{i+1} están determinadas por (a_i, b_i, c_i), (a_j, b_j, c_j) y (a_k, b_k, c_k), la nueva función de pertenencia viene dada por $(a_i, (b_i + b_j + b_k)/3, c_k)$.

4.3 Operación 3.5. Si las funciones de pertenencia de las etiquetas d_{i-1}, d_i y d_{i+1} están determinadas por (a_i, b_i, c_i), (a_j, b_j, c_j) y (a_k, b_k, c_k), las nuevas funciones de pertenencia son (a_i, b_i, c_j) y (a_j, b_k, c_k).

5. *Derivación de las reglas de decisión a partir de la tabla de decisión.* En este paso, por cada celda se genera una regla con los conjuntos difusos correspondientes a la misma.

1.5 Selección de Características

Como hemos mencionado en la Sección 1.1.1, en el diseño de un Sistema de Clasificación se parte de un conjunto de datos formado por patrones con clase conocida. Cada una de estas instancias está representada por un conjunto de características que presumiblemente contienen suficiente información para distinguir entre clases. El objetivo final es inducir un Sistema de Clasificación con capacidad para predecir la clase de un ejemplo desconocido, con el mayor grado de precisión y la mayor simplicidad en su estructura posibles.

Para un problema concreto, el número de características candidatas puede ser elevado por alguno de los siguientes motivos [LM98]:

- Los datos no se han obtenido para un fin específico, por lo que características que son necesarias para una tarea, serán redundantes o irrelevantes para otra. Esto ocurre, por ejemplo, cuando se realiza un registro de información para diagnóstico médico. En estos casos se almacena toda la información disponible para un paciente, información útil para la obtención de reglas de diagnóstico de distintas enfermedades pero con partes irrelevantes para cada problema de clasificación (de diagnóstico) específico.

- Los datos se han recogido sin conocimiento claro de las variables relevantes para el problema a resolver. En estas situaciones se suelen introducir todas las características consideradas remotamente relevantes. Como consecuencia, y de forma inevitable, algunas de ellas serán redundantes o irrelevantes.

- Un determinado problema puede necesitar datos de distintas fuentes de información, y si la cantidad de información de cada una de ellas es moderadamente grande, la unión será enorme.

Por lo tanto, es necesario un algoritmo de Selección de Características (SC), previo al proceso de aprendizaje del Sistema de Clasificación, que determine las características relevantes para alcanzar el máximo rendimiento con el mínimo esfuerzo de medida.

Los resultados inmediatos de la SC son:

- Menor cantidad de datos, de forma que el algoritmo de aprendizaje pueda aprender más rápidamente.
- Mayor precisión, que hará que el Sistema de Clasificación obtenido pueda generalizar mejor.
- Resultados más simples y más fáciles de entender.
- Menor número de características, que puede hacer que en nuevas etapas de recogida de datos para el mismo problema se pueda evitar la obtención de características irrelevantes o redundantes.

El proceso de SC es una etapa de simplificación del Sistema de Clasificación previa a su diseño, por lo que en esta memoria desarrollaremos dos métodos de SC evolutivos independientes de la estructura de representación del conocimiento y del proceso de aprendizaje del Sistema de Clasificación. Para ello, en esta sección definiremos el problema de la SC y los componentes de un algoritmo de SC *para un Sistema de Clasificación en general* (y no específicamente para un SCBRD) y en el Capítulo 4 describiremos nuestras propuestas mostrando la validez de los resultados obtenidos para el diseño de un SCBRD y de otros tipos de Sistemas de Clasificación.

En las siguientes subsecciones, describimos los distintos enfoques del proceso de SC utilizados en la literatura especializada y analizamos sus componentes.

1.5.1 El Proceso de Selección de Características

Distintos autores han definido el proceso de SC de diferentes formas en función de su objetivo. Algunas de ellas son [DL97]:

1. Un proceso de SC trata de encontrar el conjunto mínimo de características necesario y suficiente para el concepto a describir [KR92a]. Es una versión teórica e idealizada del proceso.

2. Un algoritmo de SC selecciona para un problema con N características, un subconjunto de tamaño H (siendo $H \leq N$) que optimice el valor de una función criterio sobre todos los subconjuntos de tamaño H [NF77].

3. El objetivo de un algoritmo de SC es la determinación de un subconjunto de características que mejore la precisión del Sistema de Clasificación, o bien que disminuya la complejidad sin decrementar la precisión del Sistema de Clasificación construido sobre esas características.

4. El objetivo de un proceso de SC es la selección de un subconjunto de variables, tal que la distribución de clases resultante aportando valores sólo para las características seleccionadas, esté tan cerca como sea posible de la distribución de clases dadas todas las características.

La definición 1 implica la resolución de un problema de optimización combinatorial de forma exacta, con el inconveniente añadido de la dificultad de especificación del conjunto de "características necesarias y suficientes para el concepto a definir" de forma independiente del método de aprendizaje inductivo a utilizar posteriormente para el diseño del Sistema de Clasificación. La mayoría de los algoritmos de aprendizaje intentan encontrar una hipótesis a través de una aproximación de problemas de optimización NP, por lo que en ocasiones es necesario definir un subconjunto optimal de características respecto a un algoritmo de aprendizaje concreto, considerando su heurística, sesgo y compensaciones. En estas situaciones, el problema de SC "necesarias y suficientes" se puede reducir a un problema de SC optimales definido de la siguiente forma:

Dado un algoritmo de aprendizaje inductivo y un conjunto de datos con características $X_1, \ldots, X_N$, un *conjunto de características optimal* es el conjunto de características para el cual la precisión del Clasificador inducido es maximal.

La definición 4 implica el conocimiento de la distribución de probabilidad subyacente en los datos y en la mayoría de los problemas reales se desconoce a priori y no se dispone de suficientes datos para realizar una estimación adecuada de la misma.

Cualquiera de las definiciones establece el proceso de SC como un proceso de búsqueda de variables *relevantes* para un problema dado. En la literatura existen distintas definiciones de *relevancia* [GLF89, AD91, JKP94, BL97]. En [KJ97] Kohavi y John muestran, a través de ejemplos, que las definiciones propuestas dan respuestas no esperadas ante determinadas situaciones de correlación entre variables y no ofrecen una solución válida para problemas reales en los que no se tiene acceso a la distribución de probabilidad subyacente en los datos.

De este análisis de las distintas definiciones para el proceso de SC se puede extraer la siguiente definición general. Para un problema con un número total de variables N, un método de SC trata de encontrar entre los 2^N subconjuntos de variables posibles el mejor subconjunto de variables de acuerdo a alguna medida de evaluación dependiente del

problema, del objetivo de la SC, y en ocasiones -si el objetivo fundamental es incrementar la precisión del Sistema de Clasificación resultante- del método de aprendizaje a utilizar.

1.5.2 Componentes de un Algoritmo de Selección de Características

El proceso de SC es un proceso de búsqueda en el cual cada estado dentro del espacio de búsqueda especifica un subconjunto de características candidato. En él se distinguen los siguientes componentes:

1. Un *algoritmo de búsqueda*, que explora el espacio de subconjuntos de variables.
2. Una *función de evaluación*, que mide la bondad de un subconjunto de características producido por el proceso de generación de subconjuntos de características integrado en el algoritmo de búsqueda.
3. Una *función de rendimiento*, que determina la validez del conjunto de variables obtenido por el proceso de SC. Para problemas de clasificación será la precisión del Sistema de Clasificación obtenido a partir de esas variables.

   Este procedimiento de validación no es parte del proceso de SC, pero en la práctica está presente ya que cualquier método de SC debe ser validado.

La naturaleza del proceso de búsqueda vendrá determinada, además de por los aspectos anteriores, por el punto de inicio dentro del espacio (que influirá en la dirección de la búsqueda y en los operadores utilizados para generar nuevos subconjuntos de características) y por el criterio de parada del proceso de búsqueda. El proceso completo se muestra en la Figura 1.4. En las siguientes secciones describiremos distintas alternativas en el diseño de los principales componentes, el algoritmo de búsqueda y la función de evaluación.

1.5.2.1 Algoritmo de Búsqueda

Para un problema con un número total de características N, el espacio de búsqueda tiene tamaño 2^N, un número enorme incluso para tamaños medios de N. Existen distintas formas de abordar este problema de búsqueda o, lo que es lo mismo, existen distintos procedimientos de generación de subconjuntos de variables, encuadrados en los siguientes grupos:

- *Algoritmos de búsqueda completa*, que exploran todo el espacio de búsqueda para encontrar el subconjunto de variables optimal respecto a la función de evaluación

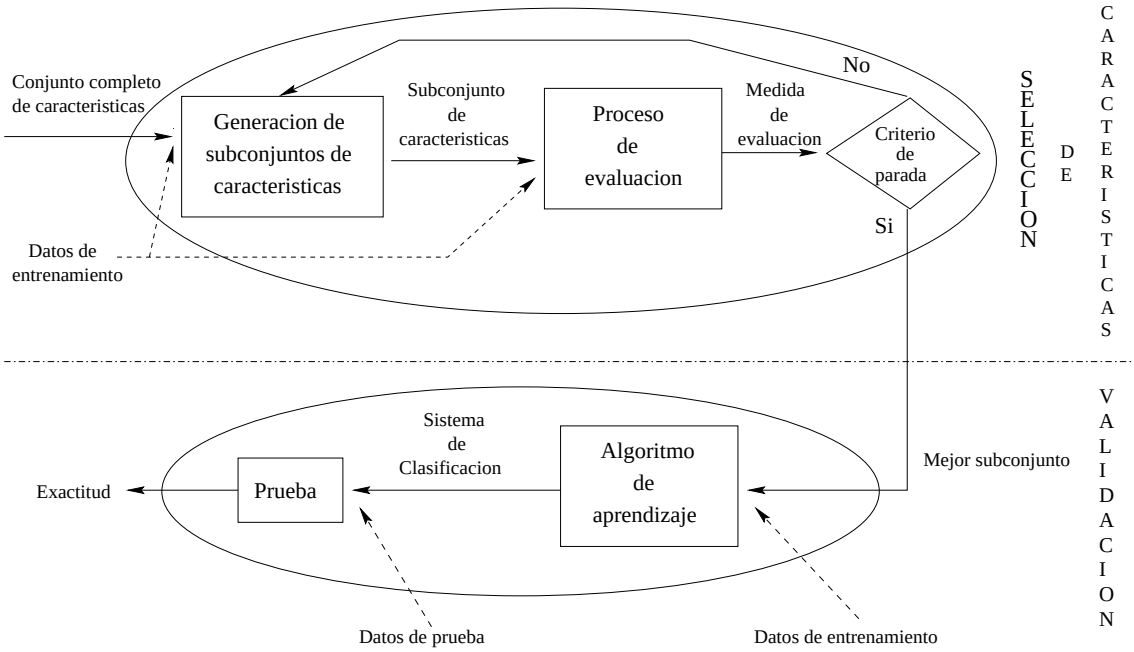

Figura 1.4. Proceso de Selección de Características

utilizada. Como ejemplo tenemos los algoritmos de búsqueda exhaustiva [AD91, Alm94], los algoritmos Branch and Bound [NF77] y sus variantes [SS88, LMD98], o la búsqueda del mejor primero [FN95], entre otros. Este tipo de métodos tienen complejidad $O(2^N)$, por lo que no se pueden utilizar en la mayoría de los problemas reales.

- *Algoritmos de búsqueda heurística*, que para evitar una búsqueda completa, emplean heurísticas para orientar la búsqueda aún a riesgo de perder subconjuntos de variables optimales. Dentro de este grupo se incluyen los algoritmos de búsqueda secuencial [AB96] hacia delante, hacia atrás, o combinados, que utilizan técnicas de ascensión de colinas, quitando o añadiendo una variable a un subconjunto de variables, como el enfoque Greedy [Car93, CF94, JKP94, LS94b, ML94].

- *Algoritmos de búsqueda no determinística* que, a diferencia de las dos estrategias de búsqueda precedentes, obtienen el siguiente subconjunto de forma no determinística, por lo que no se debe esperar la misma solución en diferentes ejecuciones. Con ellos se trata de evitar la caida en mínimos locales que puede ocurrir con los algoritmos de búsqueda heurística. Aquí se encuadran, entre otros, los Algoritmos Genéticos o los basados en Enfriamiento Simulado.

Respecto al *punto de inicio*, que implícitamente determina en algunos casos la dirección de búsqueda, puede ser un conjunto vacío de variables al que el proceso irá añadiendo sucesivamente variables (y por tanto se desarrollará un proceso de búsqueda hacia delante), o un conjunto con todas las variables con un proceso de eliminación de características

a través de una búsqueda hacia atrás. Dos alternativas adicionales, son el inicio en un punto intermedio y el desarrollo del proceso de SC a través de búsqueda bidireccional, o el comienzo en un subconjunto aleatorio y una generación aleatoria que evite que el algoritmo de SC caiga en un óptimo local.

El *criterio de parada* del algoritmo de búsqueda debe determinar en qué punto se detendrá la búsqueda en el espacio de subconjuntos de características, aspecto totalmente influenciado por el objetivo del proceso de SC. Como posibles condiciones de parada se pueden señalar: (1) la selección de un subconjunto de características suficientemente bueno según un criterio de evaluación; (2) la obtención de alguna cota preestablecida como por ejemplo la selección del número de características buscado o la generación del número máximo preestablecido de subconjuntos; (3) el mantenimiento del valor de la función de evaluación independientemente de la incorporación o eliminación de nuevas características; o (4) que se haya completado la búsqueda, en cuyo caso se habría realizado un proceso de búsqueda exhaustiva.

1.5.2.2 Función de Evaluación

Cualquier algoritmo de SC es un proceso de búsqueda de un conjunto de características optimales respecto a una función de evaluación. Esta función debe intentar medir la capacidad de discriminación del conjunto de variables entre las distintas clases del problema. En la literatura especializada se han propuesto distintos criterios de selección, que se pueden agrupar de la siguiente forma [BB82, Lan94, KJ97, LM98]:

1. **Medidas de separabilidad de clases**, dentro de las cuales se incluyen:

   (a) *Medidas de información*, basadas en medidas de incertidumbre respecto a la clase verdadera. Así, dada una función de incertidumbre y las probabilidades de las clases $P(C_j)$ (con $j = 1, \ldots, M$), la ganancia de información de una característica X viene dada por la diferencia entre la incertidumbre a priori y la incertidumbre esperada utilizando la variable X. Según este criterio, una variable será seleccionada si reduce más incertidumbre. Esta idea se utiliza en ID3 [Qui86] y C4.5 [Qui93] para seleccionar una variable en la construcción de un árbol de decisión.

   Una medida de información muy utilizada en procesos de SC es la entropía de Shannon. Ben-Bassat hace en [BB82] un estudio de medidas de este tipo.

   (b) *Medidas de dependencia, asociación o correlación.* Este tipo de medidas cuantifican la fuerza con la que dos variables están correlacionadas, para así determinar de qué forma el valor de una variable puede predecir el de otra. En procesos de SC se busca la característica con mayor grado de asociación con la variable de clase. El estudio de la dependencia de una variable con otras puede además determinar el grado de redundancia de la variable.

(c) *Medidas de distancia, separabilidad, divergencia o discriminación.* Para un problema con dos clases, un proceso de SC basado en este tipo de criterio de selección determina que una característica X es preferible a otra Y si X induce una diferencia mayor que Y entre las dos probabilidades condicionales de las clases. Si la diferencia fuese cero, las variables X e Y se considerarían indistinguibles.

(d) *Medidas de consistencia.* Los tres grupos de medidas definidos anteriormente tienen un aspecto común: intentan encontrar las características que puedan, de forma maximal, predecir una clase frente al resto. Este enfoque no puede distinguir entre dos variables igualmente adecuadas, por lo que no puede detectar variables redundantes en el conjunto seleccionado.

Las medidas de consistencia tratan de encontrar el mínimo número de características que puedan separar las clases de la misma forma en que lo hace el conjunto completo de variables.

2. **Medidas de exactitud**, basadas en el cálculo de la precisión de un Sistema de Clasificación inducido a partir de las variables seleccionadas. De esta forma, el proceso de SC seleccionará, para un algoritmo de aprendizaje dado, el subconjunto de características con el cual el Sistema de Clasificación inducido tenga el mejor comportamiento.

La descripción de las distintas medidas a utilizar para evaluar un subconjunto de características en el proceso de búsqueda permite definir de forma general una *característica relevante* como aquella que provoca el deterioro de la medida de evaluación del conjunto de variables restantes al ser eliminada, siendo dicha medida una medida de exactitud, consistencia, información, distancia o dependencia [LM98]. Esta definición de relevancia no está basada en la asunción de que la optimalidad se define en términos de precisión.

El uso de medidas de evaluación de separabilidad de clases o de exactitud permite distinguir dos tipos de algoritmos de SC, los algoritmos de SC tipo filtro y los envolventes, descritos en las secciones siguientes.

1.5.3 Modelos de Selección de Características Envolventes

Una de las formas más simples e inmediatas de realizar el proceso de SC para un problema de clasificación implica la utilización del algoritmo de aprendizaje para proporcionar una medida de evaluación de cada uno de los subconjuntos de variables implicados en el proceso de búsqueda. Parece evidente que la búsqueda de un subconjunto de variables "adecuado" para diseñar con la mayor precisión posible un Sistema de Clasificación debe incluir en la función de evaluación de todo subconjunto candidato, la precisión alcanzada por un Sistema de Clasificación inducido con esas variables. En este caso, el

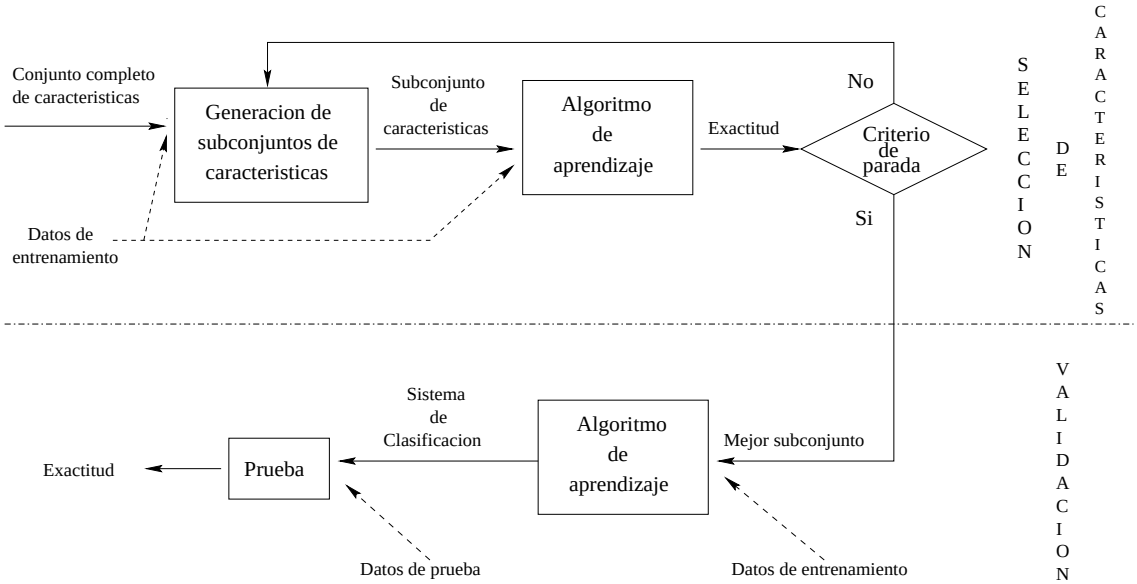

Figura 1.5. Modelo de selección de características envolvente

algoritmo de SC se puede ver como una *envoltura* del algoritmo de aprendizaje inductivo, por lo se denomina *algoritmo de SC envolvente*. El proceso se describe en la figura 1.5.

En este modelo se pueden distinguir cuatro etapas importantes:

1. Generación de un subconjunto de características, de acuerdo a algún algoritmo de búsqueda de los descritos en la sección anterior.

2. Evaluación de cada nuevo subconjunto generado, que implica el aprendizaje del Sistema de Clasificación y el cálculo de su precisión.

3. Evaluación del criterio de parada: mientras no se verifique, se seguirán generando subconjuntos de variables.

4. Validación, posterior al proceso de SC, del conjunto de variables obtenido. Esto implica el aprendizaje del Sistema de Clasificación para la obtención de la capacidad de generalización sobre datos de prueba.

Tal y como Siedlecki y Sklansky argumentan en [SS88], Doak en [Doa92] y Kohavi en [KJ97], el enfoque envolvente es el más apropiado si nuestro objetivo es minimizar el error de clasificación y el costo de medida para todas las características es el mismo. No obstante, presenta algunos problemas como la posibilidad de sobreajuste a los datos de entrenamiento y el costo computacional del cálculo de la función de evaluación.

Además, en su diseño hay que determinar el mecanismo de estimación de error a utilizar en el proceso de SC, ya que para la obtención de la medida de evaluación se debe estimar el error sobre datos de prueba a partir de los datos de entrenamiento. Kohavi y

John aportan en [KJ97] una buena solución a este problema, consistente en la división del conjunto de entrenamiento en distintos subconjuntos entrenamiento-prueba según la filosofía de validación cruzada [WK91] y el cálculo de la estimación del grado de precisión del Sistema de Clasificación a través de la media aritmética de los resultados obtenidos para esos subconjuntos de prueba. En el Capítulo 4 se describe de forma más detallada este mecanismo de estimación utilizado en los métodos envolventes desarrollados.

El enfoque envolvente para la resolución del problema de SC ha tenido bastante uso desde el punto de vista estadístico y de reconocimiento de patrones [DK82] donde el problema de SC ha sido un área de investigación activa durante mucho tiempo. En el campo del aprendizaje automatico, la investigación es más reciente. Doak en [Doa92], Kohavi en [JKP94], y Caruana y Freitag en [CF94] muestran estudios comparativos de tipo experimental de distintos métodos de SC envolventes propuestos en la literatura especializada.

Respecto a la función de evaluación empleada, nos encontramos distintas líneas de trabajo. Se han desarrollado algoritmos de SC que utilizan como función de evaluación el rendimiento de Sistema de Clasificación basado en árboles de decisión [CF94, LS96a, KJ97].

La técnica del vecino más cercano es sensible a variables irrelevantes, por lo que también se ha utilizado como medida de evaluación envolvente en sistemas como OBLIVION [LS94b] desarrollado por Langley y Sage. Es un método de SC que, a través de un proceso Greedy de búsqueda hacia atrás, determina el suconjunto de características que maximiza la estimación (mediante la técnica *leaving-one-out*) de la precisión de la regla del vecino más cercano. Funciona bien frente características irrelevantes, aunque como consecuencia del sesgo del proceso inductivo utilizado (el vecino más cercano) no es sensible a variables con correlación alta (que no alteran el funcionamiento del vecino más cercano). Skalak [Ska94] utiliza un proceso de búsqueda de ascensión aleatoria de colinas que, partiendo de un subconjunto aleatorio de variables, determina el subconjunto de variables que maximiza el rendimiento de la regla del vecino más cercano en un número especificado de ciclos. Moore y Lee [ML94] por una parte y Townsend-Weber y Kibler [TWK94] por otra utilizan el mismo enfoque pero con el k-vecino más cercano.

También se han utilizado para proporcionar el valor de la función de evaluación el Clasificador Bayesiano en [LS94a] o las redes bayesianas más complejas en [SP95].

1.5.4 Modelos de Selección de Características Filtro

Los *algoritmos de selección de características filtro* asignan una medida de evaluación a los subconjuntos de variables ignorando el algoritmo de aprendizaje inductivo. Reciben el nombre de su comportamiento que permite filtrar características irrelevantes antes del proceso de aprendizaje inductivo. Para ello suelen utilizar medidas de distancia,

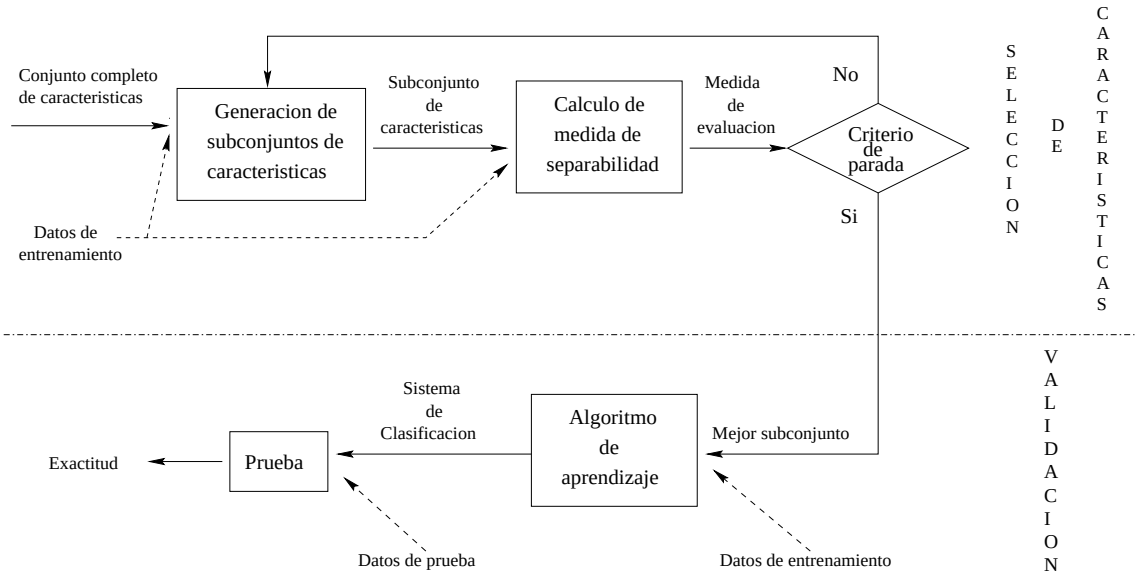

Figura 1.6. Modelo de selección de características filtro

consistencia, dependencia o información como las descritas en la Sección 1.5.2.2.

El modelo filtro tiene las mismas etapas que el modelo envolvente y se diferencia de éste en que el proceso de evaluación no implica la construcción de un Sistema de Clasificación, sino el cálculo de la medida de separabilidad de clases correspondiente. El proceso se describe en la figura 1.6.

Se han realizado distintas propuestas de algoritmos de SC tipo filtro en la literatura especializada. Entre ellas podemos destacar el algoritmo FOCUS, desarrollado por Almuallim y Ditterich [AD91, Alm94], que comienza con un conjunto vacío de características y realiza una búsqueda exhaustiva hasta encontrar la combinación mínima de variables suficiente para construir una hipótesis *consistente* con un conjunto de ejemplos de entrenamiento.

RELIEF es un algoritmo filtro de SC con búsqueda heurística propuesto por Kira y Rendell en [KR92a, KR92b] e inspirado en aprendizaje basado en instancias, que detecta características *irrelevantes* para el concepto a aprender. El proceso asigna un peso de relevancia a cada característica, indicador de la relevancia de la variable para el concepto a aprender. No detecta variables redundantes y sólo trabaja con problemas de clasificación con dos clases, aunque fue extendido a problemas con múltiples clases por Kononenko [Kon94].

Otras propuestas importantes son los algoritmos

- LVF, propuesto por Liu y Setiono [LS96b], que utiliza al igual que FOCUS una medida de inconsistencia pero, a diferencia de éste, realiza una búsqueda aleatoria, a través del algoritmo probabilístico de Las Vegas, del conjunto mínimo de variables

que no supera un nivel de inconsistencia prefijado, y

- MIFS, desarrollado por Battiti en [Bat94] y basado en un proceso de búsqueda secuencial guiado por la medida de información mutua.

En esta memoria se han utilizado ambos algoritmos para comparar sus resultados con los de los algoritmos de SC propuestos, por lo que los describiremos brevemente en las siguientes subsecciones.

Resumiendo lo descrito en esta sección, podemos afirmar que los algoritmos de SC filtro tienen las siguientes características:

- Trabajan con las propiedades de los datos y no con el sesgo de ningún algoritmo de aprendizaje en concreto, lo que hace que las características seleccionadas puedan ser utilizadas para aprender Sistemas de Clasificación con distintos modelos de aprendizaje inductivo.
- El costo del cálculo de la función de evaluación es inferior al de los modelos envolventes.
- La baja complejidad de los modelos filtro y la simplicidad de cálculo de las medidas de separabilidad de clases hacen que este tipo de algoritmos de SC sean más adecuados para el manejo de bases de ejemplos grandes o con un número total de características amplio.

No obstante, tienen el inconveniente de que puede que las variables seleccionadas no presenten el mejor comportamiento para el diseño de un Sistema de Clasificación específico, ya que se han elegido ignorando la heurística, sesgo y compensaciones del algoritmo de aprendizaje a utilizar [KJ97].

1.5.4.1 LVF: Algoritmo Probabilístico Tipo Filtro de Selección de Características de Liu y Setiono

En [LS96b, LM98] Liu y Setiono proponen el algoritmo filtro Las Vegas Filter (LVF), basado en el proceso de búsqueda probabilística de Las Vegas [BB96], que minimiza el *ratio de inconsistencia* [LM98] introducido por el subconjunto de características seleccionado, para así incrementar el poder de discriminación del mismo.

El ratio de inconsistencia de un conjunto de características se calcula de la siguiente forma:

1. Dos instancias se consideran inconsistentes si tienen valores iguales para las variables seleccionadas y difieren en la clase.

Algoritmo LVF
inicio
Mejor_Subcto= $\{1, \ldots, N\}$
Para $i = 1$ **hasta** *Max_Iteraciones* **hacer**
$S \longleftarrow$ Conjunto_Aleatorio
Si $|S| \leq |$Mejor_Subcto$|$
Entonces
Si Inconsistencia$(S,$ Cto_Entr$) \leq \gamma$
Entonces
Mejor_Subcto $\longleftarrow S$
Mostrar_Mejor_Subcto_Actual(S)
Si_No **Si** $|S| = |$Mejor_Subcto$|$ e Inconsistencia $(S, Cto\_Entr) \leq \gamma$
Entonces
Mostrar_Mejor_Subcto_Actual (S)
fin_para
fin

Figura 1.7. LVF. Algoritmo probabilístico tipo filtro de Selección de Características de Liu y Setiono

2. La cuenta de inconsistencia es igual al número total de instancias inconsistentes, menos el número mayor de instancias inconsistentes entre las distintas clases.

3. El ratio de inconsistencia es la suma de todas las cuentas de inconsistencia dividido entre el número total de instancias.

El proceso comienza seleccionando aleatoriamente un subconjunto de características. Si el tamaño del subconjunto generado es menor que el mejor obtenido hasta el momento (el conjunto completo en el caso inicial), se analiza si se cumple el criterio de inconstencia mínima y, en caso afirmativo, se almacena el subconjunto de variables como mejor actual. El proceso continua hasta que se alcanza un número tope de iteraciones dependiente del número total de variables del problema. Los autores proponen distintos valores para este parámetro: $77 * N$, N^2 y $2^N * k$ (siendo k el porcentaje del espacio de búsqueda que queremos explorar). En nuestra experimentación hemos utilizado $77 * N$, uno de los valores para el que se obtuvieron mejores resultados en los experimentos realizados por los autores del método.

El esquema del algoritmo se muestra en la figura 1.7.

El criterio de inconsistencia es la clave del buen funcionamiento de LVF ya que especifica hasta qué punto puede ser aceptable la reducción de la dimensionalidad de los datos.

Algoritmo MIFS
inicio
$V \longleftarrow \{1, \ldots, N\}$
$S \longleftarrow \emptyset$
Para cada $v \in V$ **hacer**
Calcular $I(C, v)$
fin_para
Encontrar v tal que $I(C, v) = \max_{x \in V} I(C, x)$
$V \longleftarrow V - \{v\}$
$S \longleftarrow \{v\}$
Repetir
Para $v \in V$ **hacer**
Para $s \in S$ **hacer**
Calcular $I(v, s)$, si no se ha calculado ya
fin_para
fin_para
Encontrar v que maximice $I(C, v) - \beta \sum_{s \in S} I(v, s)$
$V \longleftarrow V - \{v\}$
$S \longleftarrow S \cup \{v\}$
hasta que $|S| = k$
fin

Figura 1.8. MIFS. Algoritmo Greedy tipo filtro de selección de características de Battiti

Los autores proponen también un algoritmo alternativo a utilizar en el caso de que la base de ejemplos tenga una cardinalidad muy alta [LMD98].

1.5.4.2 MIFS: Algoritmo Greedy Tipo Filtro de Selección de Características de Battiti

En [Bat94] Battiti propone un algoritmo de SC denominado MIFS (Mutual Information based Feature Selection). Es un algoritmo filtro basado en una búsqueda secuencial hacia delante tipo Greedy, que selecciona las variables más informativas sobre la clase, no predecibles a partir de las variables ya seleccionadas. Para ello utiliza la *medida de información mutua* [SW49] en el proceso que se describe en la figura 1.8.

En el algoritmo, k es el número de variables a obtener (fijado a priori), $I(C, v)$ la medida de información mutua entre la variable v y la clase de salida C, $I(v, s)$ la medida de información mutua entre dos variables, y $\beta \in [0, 1]$ un parámetro determinante de la importancia de la medida de información mutua entre la variable candidata y las ya seleccionadas, respecto a la medida de información mutua de la variable candidata y la

variable de clase. En definitiva, este último parámetro representa la importancia, en el proceso de selección, de la eliminación de variables correlacionadas.

1.6 Computación Evolutiva

La *Computación Evolutiva* (CE) se basa en la simulación de varios aspectos de la evolución para el diseño e implementación de sistemas de resolución de problemas basados en el ordenador. Como campo reconocido es relativamente joven, ya que el término se acuñó en 1991. En este área se han propuesto distintos modelos computacionales denominados *Algoritmos Evolutivos* (AEs) [Bac96] y agrupados en torno a tres tipos de AEs distintos:

- los *Algoritmos Genéticos* (AGs) [Hol75, Gol89],
- las *Estrategias de Evolución* (EEs) [BS93, BS95, Sch95], y
- la *Programación Evolutiva* (PE) [FOW66, Fog91].

Estos tipos de AEs tienen un aspecto común: consideran dentro de una población los procesos de reproducción, variación aleatoria, competición y selección de individuos rivales. Estos cuatro aspectos constituyen la esencia de la evolución y, cuando alguno de ellos ocurre, tanto en la naturaleza como en un ordenador, la tenemos como resultado inevitable.

En las siguientes subsecciones describiremos con más detalle las características de los AEs e introduciremos dos de los AEs más conocidos y que serán utilizados a lo largo de esta memoria, los AGs y las EEs.

1.6.1 Algoritmos Evolutivos

Como se ha mencionado anteriormente, todas las instancias o tipos de AEs tratan de modelar la evolución y presentan las siguientes características comunes:

- Utilizan el proceso de aprendizaje colectivo de una población de individuos. Normalmente cada individuo representa o codifica un punto dentro del espacio de búsqueda de todas las soluciones potenciales para un problema dado. Los individuos pueden incorporar adicionalmente otra información como parámetros de la estrategia del AE.

- Los descendientes de los individuos se generan mediante procesos aleatorios que tratan de modelar los procesos de mutación y recombinación. La *mutación* corresponde a una auto-replicación errónea de los individuos, mientras que la *recombinación o cruce* intercambia información entre dos o más individuos ya existentes.

- Se asigna una medida de la calidad (denominada habitualmente *medida de adaptación o fitness*) a cada individuo mediante el proceso de evaluación de los mismos. El operador de selección actúa en base a esta medida de adaptación y favorece, en el proceso de reproducción, a individuos mejores respecto a aquellos con peor valor de la función de adaptación.

Estas son las características más generales de los AEs. Las distintas instancias de los AEs se diferencian fundamentalmente en:

- la representación de los individuos,

- el diseño de los operadores de variación (mutación y/o cruce), y

- el mecanismo de selección o reproducción.

Las características generales de los tipos principales de AEs son:

- **Algoritmos Genéticos**. Fueron propuestos y analizados inicialmente por Holland [Hol75] y han sido estudiados en profundidad posteriormente por Goldberg y Michalewicz [Gol89, Mic96], entre otros. Hacen mayor énfasis en el operador de recombinación o cruce como el operador de búsqueda más importante y aplican el operador de mutación con una probabilidad muy baja. Tradicionalmente han utilizado un operador de selección probabilístico y una representación binaria de los individuos, aunque no están restringidos a su uso.

- **Estrategias de Evolución**. Fueron desarrolladas por Rechenberg y Schwefel [BS93, BS95, Sch95]. Utilizan mutaciones con distribución normal para modificar individuos con codificación real y los operadores de mutación y cruce como operadores de búsqueda dentro del espacio de soluciones y del espacio de parámetros de la estrategia. Normalmente el operador de selección es determinístico y el tamaño de la población padre y de la descendiente es distinto.

- **Programación Evolutiva**. Fue desarrollada originalmente por Fogel [FOW66]. Utiliza el operador de mutación y no incorpora la recombinación de individuos. Al igual que las estrategias de evolución, cuando resuelven problemas de optimización con parámetros reales, también trabajan con mutación normalmente distribuida y extienden el proceso evolutivo a los parámetros de la estrategia. El operador de selección suele ser probabilístico.

Además de estas tres líneas de trabajo principales, hay que señalar otras técnicas como la *Programación Genética*, que aplica el principio de búsqueda evolutiva para desarrollar programas de forma automática en algunos lenguajes de programación o los *Sistemas Clasificadores* que buscan dentro del espacio de reglas de producción de la forma SI *condición* ENTONCES *acción*.

El funcionamiento de cualquier AE se puede describir de la siguiente forma: se mantiene una población de posibles soluciones para el problema, se realizan modificaciones sobre las mismas y se seleccionan en función de una medida de adaptación del individuo al entorno, aquellas que se mantendrán en generaciones futuras y las que serán eliminadas. La población evoluciona a través de las mejores regiones del espacio de búsqueda mediante los procesos de modificación y selección. Las modificaciones sobre la población permiten mezclar información de los padres que debe pasar a los descendientes (operador de cruce) o introducir innovación dentro de la población (operador de mutación).

Podemos definir este funcionamiento en un marco más formal utilizando la siguiente notación:

- Notaremos mediante I el espacio de individuos $C \in I$.
- Utilizaremos como función de evaluación $f : I \longrightarrow I\!R$.
- μ y λ serán los tamaños de las poblaciones padre y descendiente respectivamente.
- $P(t) = (C_1^t, \ldots, C_\mu^t) \in I^\mu$ representará una población en la generación t.
- Los operadores de selección, recombinación y mutación los describiremos respectivamente a través de las funciones $s : I^\lambda \longrightarrow I^\mu$, $r : I^\mu \longrightarrow I^k$, $m : I^k \longrightarrow I^\lambda$, y transformarán poblaciones completas. Para cubrir las distintas instancias de los AEs, hemos descrito todos los operadores a nivel de población, aunque para la mutación el operador puede reducirse para aplicarse a nivel de individuos definiendo m mediante aplicación múltiple de un operador $m' : I \longrightarrow I$ sobre individuos.

  Estos operadores dependen de conjuntos de parámetros Θ_s, Θ_r y Θ_m específicos del operador y del esquema de representación de los individuos. Además el proceso utiliza

  - un procedimiento de generación de una población de individuos (típicamente aleatorio, aunque también es posible una inicialización con individuos que representen puntos conocidos del espacio de búsqueda),
  - una rutina de evaluación que determine el valor de la función de adaptación para cada uno de los individuos de una población, y
  - un criterio de parada, *Parada*, que especifique cuándo debe parar el proceso (determinado por un conjunto de parámetros Θ_p).

Algoritmo Evolutivo
inicio
$t \longleftarrow 0$
Inicializar $(\mu, P(t))$
Evaluar $(P(t), \mu, f)$
Mientras $Parada(P(t), \Theta_p) \neq Verdadero$ **Hacer**
Recombinar$(P(t), \Theta_r, P'(t))$
Mutar $(P'(t), \Theta_m, P''(t))$
Evaluar $(P''(t), \lambda, f)$
Seleccionar $(P''(t), f, \mu, \Theta_s, P(t+1))$
$t \longleftarrow t+1$
fin_mientras
Devolver el mejor individuo C o la poblacion P
fin

Figura 1.9. Algoritmo Evolutivo básico

Con la notación descrita, un AE básico se reduce un ciclo recombinación-mutación-selección como se describe en la figura 1.9.

Los parámetros de entrada de este AE general son los tamaños de población μ y λ, y los conjuntos de parámetros del criterio de parada y de los operadores básicos (selección, recombinación y mutación).

En este esquema general es posible que $\lambda = 1$, es decir, que sólo se genere un individuo mediante las operaciones de recombinación y mutación, por lo que incluye el esquema de selección *de estado estacionario puro* que será estudiado en el Capítulo 4. Además se puede ajustar el valor del *salto generacional*, es decir, del número de individuos nuevos en cada generación, determinando el valor del parámetro λ, $1 \leq \lambda \leq \mu$. De esta forma se puede conseguir una transición entre los algoritmos evolutivos estrictamente *generacionales* y las variantes *de estado estacionario*.

1.6.2 Algoritmos Genéticos

Los AGs son algoritmos de búsqueda de propósito general que utilizan principios inspirados por la genética natural para evolucionar soluciones a problemas [Hol75, Gol89]. Han demostrado de forma teórica y práctica que proporcionan un mecanismo de búsqueda robusto dentro de espacios complejos. Hay tres características que distinguen los AGs -como Holland los propuso inicialmente- de otros AEs: (1) el esquema de codificación binario, (2) el método de selección, la selección proporcional, y (3) el método básico para producir variaciones en la población, el cruce. Es fundamentalmente esta tercera característica, la

Algoritmo Genetico
inicio
$t \longleftarrow 0$
Inicializar $(P(t))$
Evaluar $(P(t), f)$
Mientras $Parada(P(t), \Theta_p) \neq Verdadero$ **Hacer**
$t \longleftarrow t + 1$
Seleccionar $(P(t-1), f, \Theta_s, P(t))$
Recombinar$(P(t), \Theta_r, P'(t))$
Mutar $(P'(t), \Theta_m, P''(t))$
$P(t) \longleftarrow P''(t)$
Evaluar $(P(t), f)$
fin_mientras
Devolver el mejor individuo C o la poblacion P
fin

Figura 1.10. Algoritmo Genético básico

que hace a los AGs diferentes del resto de los AEs.

En propuestas posteriores a la de Holland, se han utilizado métodos alternativos de selección y se han adoptado otros esquemas de codificación más apropiados a los problemas a resolver.

El funcionamiento básico de un AG es el siguiente: el sistema parte de una población inicial de individuos que codifican, mediante alguna representación genética, soluciones candidatas al problema propuesto. Esta población de individuos (a los que se denomina *cromosomas*) evoluciona en el tiempo a través de un proceso de competición y variación controlada. Cada cromosoma de la población tiene asociada una medida de adaptación o fitness para determinar qué cromosomas serán *seleccionados* para formar parte de la nueva población en el proceso de competición. La nueva población será creada usando operadores genéticos de *cruce* y *mutación*.

El esquema general de un AG se muestra en la figura 1.10. Si lo observamos frente al de un AE (figura 1.9), vemos que en el caso de un AG los parámetros $\lambda = k = \mu$, es decir, el tamaño de la población se mantiene en la evolución de la población.

Los AGs se han utilizado con éxito fundamentalmente en problemas de búsqueda y optimización. Una gran parte de este éxito está en su habilidad para explotar la información acumulada sobre un espacio de búsqueda inicialmente desconocido y orientar convenientemente la búsqueda hacia espacios útiles, es decir, *su adaptación*. Ésta es su característica clave, sobre todo en espacios de búsqueda poco conocidos, grandes y complejos, donde las herramientas de búsqueda clásicas son inapropiadas. Ofrece una aproximación válida

para problemas que requieren técnicas de búsqueda eficientes y efectivas.

Generalmente se acepta que la aplicación de un AG para resolver un problema debe tener en cuenta los siguientes componentes:

1. una representación genética de las soluciones del problema,
2. una forma de crear una población inicial de soluciones,
3. una función de evaluación que proporcione un valor de adaptación de cada cromosoma,
4. operadores genéticos que modifiquen la composición genética de la descendencia durante la reproducción, y
5. valores para los parámetros que utilizan los AGs (tamaño de la población, probabilidades de aplicar los operadores genéticos, etc.)

En las siguientes subsecciones analizaremos brevemente estos aspectos de los AGs.

1.6.2.1 Representación y Evaluación de Soluciones

Para poder diseñar un AG que resuelva un problema es necesario establecer una correspondencia entre el espacio de búsqueda de todas las soluciones posibles del problema, al que se denomina *espacio fenotípico*, y un conjunto de cadenas codificadas con un esquema de codificación específico, perteneciente al *espacio genotípico*. El AG trabajará con estas representaciones de las soluciones más que con las soluciones en sí por lo que el esquema de representación adoptado en el diseño de un AG es un aspecto determinante de su rendimiento ya que puede limitar de forma importante la ventana desde la cual el AG afronta el problema.

Como se ha mencionado, el esquema de codificación utilizado tradicionalmente en AGs es el binario [Hol75, Gol89], en el que los cromosomas son cadenas de bits y, en función del problema, cada gen será un bit o una subcadena de bits.

Posteriormente se han propuesto distintos esquemas de codificación más adaptados al problema a resolver, como la codificación real [LK89], que permite que a cada variable se le asocie un único gen que toma un valor real dentro del intervalo especificado. En este esquema de representación no existen diferencias entre el genotipo (la solución codificada) y el fenotipo (la solución sin codificar) por lo que se resuelven los inconvenientes de la codificación binaria en problemas con variables definidas en dominios contínuos.

Los esquemas de codificación distintos al binario necesitan operadores genéticos especiales para modificar los cromosomas, ya que los operadores clásicos no se pueden aplicar directamente sobre los individuos o si se aplican pueden generar cromosomas no válidos, es decir, individuos que al decodificarse no representan una solución válida del problema.

La función de evaluación desempeña, al igual que el esquema de codificación, un papel determinante en el AG, al guiar el proceso evolutivo dentro del espacio de búsqueda. Debe estar bien diseñada para ser capaz no sólo de distinguir individuos bien adaptados, sino también de ordenarlos en función de su capacidad para resolver el problema.

1.6.2.2 El Mecanismo de Selección

El operador de selección determina una población intermedia de individuos a los que se aplicarán los operadores de cruce y mutación para obtener así la nueva población del AG en la siguiente generación. Como en un AG el tamaño de la población se mantiene constante, el operador de selección, a partir de una población P formada por μ cromosomas $C_1, \ldots, C_\mu$, obtendrá una población P' formada por copias de los cromosomas de P. El número de copias de cada cromosoma depende de su adecuación, de forma que los individuos con un mayor valor de la función de adaptación tengan probabilidad de contribuir con un mayor número de copias a la formación de la población de la siguiente generación.

En cualquier operador de selección se pueden distinguir dos componentes: un mecanismo de asignación de probabilidades o de valores esperados de descendientes para cada individuo y un esquema de selección que determina la nueva población en función de esas probabilidades o valores esperados.

Tradicionalmente el operador de selección se ha implementado como un operador probabilístico [Gol89] que utiliza el valor relativo de la función de adaptación $\frac{f(C_i)}{\sum_{j=1}^{\mu} C_j}$ para determinar la probabilidad de selección de un determinado individuo C_i. Es lo que se denomina *selección proporcional*. Esta forma de obtención de probabilidades se utiliza con frecuencia en combinación con uno de los esquemas de selección más conocidos, el *muestreo estocástico con reemplazamiento*, que establece un paralelismo entre la población y una ruleta, y representa cada cromosoma con un sector de la ruleta de tamaño proporcional a su adaptación. Los cromosomas se seleccionan girando la ruleta tantas veces como individuos tengamos que seleccionar para formar la población intermedia. Hay que destacar que este tipo de métodos de selección eligen individuos realmente de forma probabilística más que determinística, ya que no hay garantía de que un individuo, por muy alto que sea su valor de adaptación, sea seleccionado. Lo único que se puede afirmar es que *en media* seleccionaremos los individuos en función de su adaptación.

Frente a los métodos de selección probabilísticos están los esquemas de selección determinísticos [Fal98] que reproducen siempre los mejores individuos en la población actual. Tienen la ventaja de que llevan a una convergencia más rápida al AG aunque pueden provocar con facilidad una convergencia prematura del mismo en un óptimo local de la función a optimizar.

Una alternativa al esquema de selección de muestreo estocástico con reemplazamiento es *muestreo estocástico universal* propuesto por Baker [Bak87] en el cual, el número de

copias de cada individuo en la población intermedia está acotado inferior y superiormente por un número de copias esperado que se calcula en función de su adaptación.

Los operadores de selección que emplean un mecanismo de asignación de descendientes esperados basados en ordenación utilizan los índices de los individuos una vez ordenados respecto al valor de su función de adaptación, en lugar de valores absolutos de adaptación. Dentro de este tipo de métodos de selección se han propuesto alternativas basadas en correspondencias tanto lineales [Bak87] como no lineales [Mic96]. En todos los procesos evolutivos propuestos en esta memoria se utiliza un operador de selección con el mecanismo de asignación de valores esperados por ordenación lineal y esquema de selección basado en el muestreo estocástico universal.

El mecanismo de selección se completa con el *modelo de selección elitista* que asegura que el cromosoma mejor adaptado sobrevivirá de una generación a la siguiente, evitando así que el mejor cromosoma desaparezca por la aplicación del operador de cruce o mutación [Gol89, Mic96].

1.6.2.3 El Operador de Cruce

Como hemos mencionado, los AGs aplican a la población intermedia los operadores de variación de mutación y cruce, haciendo especial énfasis en el de cruce.

El operador de cruce es un mecanismo que permite compartir información entre cromosomas combinando las características de dos cromosomas padre para formar dos descendientes con posibilidad de que los cromosomas hijos estén mejor adaptados que sus padres. Normalmente se aplica a pares de individuos seleccionados aleatoriamente en función de la probabilidad de cruce P_c.

Este operador desempeña un papel fundamental dentro del AG ya que explota el espacio de búsqueda refinando las soluciones obtenidas hasta el momento mediante la combinación de las buenas características que presenten.

La definición de este operador, al igual que la del operador de mutación depende del esquema de representación utilizado. Con un esquema de codificación binario, el operador de cruce estándar es el *cruce simple* que selecciona aleatoriamente dos individuos, determina, también de forma aleatoria, una posición de cruce dentro de la cadena y genera los cromosomas descendientes concatenando la parte derecha de un padre con la izquierda del otro y viceversa. Se han propuesto distintas extensiones de este operador entre las que se encuentra un incremento en el número de puntos de cruce, como ocurre con el operador cruce multipunto [Sys89] utilizado en algunos procesos evolutivos propuestos en esta memoria y descrito gráficamente en la figura 1.11 para el caso de dos puntos. Otras alternativas están representadas por el operador de cruce uniforme [ECS89] o la recombinación en la que intervienen más de dos individuos en la generación de un descendiente [ERR94], entre otros.

A lo largo de esta memoria se describirá el funcionamiento de otros operadores de

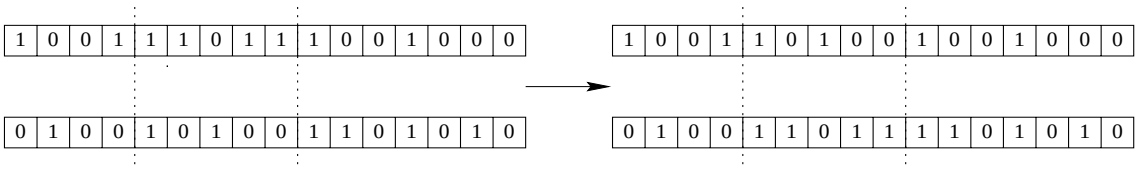

Figura 1.11. Operador de cruce multipunto

cruce adaptados al proceso evolutivo en el que se integran.

1.6.2.4 El Operador de Mutación

El operador de mutación altera aleatoriamente uno o más genes del cromosoma seleccionado para aumentar la diversidad de la población. Su aplicación depende de una probabilidad de mutación P_m.

La propiedad de búsqueda asociada al operador de mutación es la exploración ya que la modificación aleatoria de un componente de un individuo suele conllevar el salto a otra zona del espacio de búsqueda que puede resultar más prometedora.

El operador de mutación utilizado tradicionalmente en los AGs con codificación binaria modifica el valor del bit seleccionado cambiándolo por su valor complementario. Se puede transladar al área de los AGs con codificación real o entera, de forma que el nuevo valor del gen mutado se elija aleatoriamente dentro del intervalo de definición asociado. Al igual que en el caso del operador de cruce existen distintos operadores de mutación alternativos asociados a la codificación real [Mic96, HLV98b]. Entre ellos se puede destacar el operador de *mutación no uniforme* propuesto por Michalewicz en [Mic96]. Este operador ha demostrado tener un buen comportamiento en numerosas aplicaciones evolutivas con codificación real y es el operador utilizado en algunos de los procesos evolutivos propuestos en esta memoria. Su funcionamiento de describe a continuación.

Si $C = (c_1, \ldots, c_i, \ldots, c_n)$ es un cromosoma y $c_i \in [a_i, b_i]$ es el elemento a mutar en la generación t, el cromosoma resultante será $C' = (c_1, \ldots, c_i', \ldots, c_n)$, donde c_i' se determina de acuerdo a la siguiente expresión:

$$c_i' = \begin{cases} c_i + \Delta(t, b_i - c_i), & \text{si } \tau = 0 \\ c_i - \Delta(t, c_i - a_i), & \text{si } \tau = 1 \end{cases}$$

siendo τ un número aleatorio perteneciente al conjunto $\{0, 1\}$, y

$$\Delta(t, y) = y(1 - r^{(1 - \frac{t}{gmax})^b})$$

donde r es un número aleatorio perteneciente al intervalo $[0, 1]$ y b es un parámetro elegido por el usuario, que determina el grado de dependencia del número de iteraciones. Esta función da un valor en el rango $[0, y]$ tal que la probabilidad de devolver un número

cercano a cero se incrementa conforme el algoritmo avanza, es decir, la proporción en la cual un gen es mutado disminuye conforme el AG avanza. Esta propiedad hace que el operador realice una búsqueda uniforme en el espacio inicial cuando t es pequeño, y muy local en las últimas generaciones.

1.6.3 Estrategias de Evolución

Las *Estrategias de Evolución* (ES) fueron propuestas inicialmente por Rechenberg y Schwefel [BS93, BS95, Sch95] para resolver problemas de optimización complejos discretos y contínuos.

Los aspectos más característicos de este tipo de AEs son los siguientes:

1. Suelen utilizar codificación real.
2. En las propuestas iniciales, sólo se utiliza el operador de mutación para producir variación en la población.
3. Habitualmente la mutación se produce sumando a cada componente a mutar un valor con distribución normal.
4. La selección suele ser determinística, de forma que un individuo mutado pasará a la siguiente generación sólo en el caso de que su valor de adaptación sea superior al del individuo original.
5. Incluyen un mecanismo de *autoadaptación* de los parámetros de la estrategia de búsqueda en el proceso evolutivo, es decir, buscan en el espacio de soluciones y en el de parámetros de la estrategia simultáneamente.

Surgieron inicialmente como un algoritmo de selección y mutación simple por el cual un individuo crea un descendiente por generación mediante una mutación Normal. Este algoritmo cambia dinámicamente la desviación estándar de la mutación. Este es el funcionamiento básico de la EE propuesta inicialmente, a la que se denomina *Estrategia de Evolución (1+1)*. Posteriormente se han desarrollado EEs con varios individuos y operadores distintos al de mutación, aunque éste sigue siendo el de uso preferente. En [BS93, BHS97] se puede encontrar más información sobre este tipo de AEs.

Capítulo 2

Métodos de Razonamiento Difuso en Sistemas de Clasificación Basados en Reglas Difusas

Un MRD es un procedimiento de inferencia que deriva conclusiones entre un conjunto de reglas difusas y un ejemplo para su clasificación. La potencia del razonamiento aproximado reside en la posibilidad de obtención de un resultado (una clasificación) incluso cuando no tengamos compatibilidad exacta (con grado 1) entre el ejemplo y el antecendente de las reglas. Es evidente que el nivel de rendimiento del SCBRD dependerá tanto del proceso de elicitación del conocimiento utilizado en la construcción de la BC, como de la adecuación del MRD para el problema concreto a resolver y para la BC con la que se va a aplicar.

Para un ejemplo, el MRD utilizado habitualmente en SCBRDs considera la regla con un valor más alto de combinación entre el grado de compatibilidad del ejemplo con el antecedente de la regla y el grado de certeza para las clases, y clasifica el ejemplo con la clase para la que se obtiene este valor, por lo que se utiliza una única regla en el proceso de inferencia.

Al trabajar con un sistema de control difuso sabemos que los mejores resultados se obtienen cuando se utilizan métodos de defuzzificación que integran los conjuntos difusos obtenidos por todas las reglas disparadas [CHP97]. Por tanto, parece lógico utilizar MRDs en SCBRDs que integren toda la información contenida en la BC.

Esta idea de combinar en un proceso de clasificación toda la información relativa a la misma se ha utilizado también en procesos de multiclasificación. En este campo, distintos autores han examinado diferentes esquemas de combinación de los resultados obtenidos por múltiples sistemas de clasificación [BC94, BS92, Lu96, IMN96].

En este capítulo estudiamos el problema con el objetivo de integrar toda la información disponible en el SCBRD en el proceso de inferencia, es decir, considerar todas las reglas

disparadas para la clasificación. Para ello en las siguientes secciones establecemos el funcionamiento de un modelo general de inferencia y proponemos un conjunto de MRDs alternativos al método clásico, que no clasifican de acuerdo al resultado proporcionado por una única regla (la regla ganadora en algún sentido) sino que integran la información proporcionada por la BR y proporcionan una clasificación resultado de la agregación de la misma.

En primer lugar analizamos el MRD clásico, posteriormente describimos la formalización del modelo general de razonamiento y presentamos nuestras propuestas agrupadas en torno a dos tipos de modelos de inferencia:

- MRDs que utilizan toda la información disponible, es decir, hacen que todas las reglas compatibles con el ejemplo participen en su clasificación, y
- MRDS que utilizan información parcial seleccionando las reglas difusas a considerar en el proceso de inferencia, de forma que no se consideren las reglas con un grado bajo de asociación con el ejemplo. Esta selección se modela a través de operadores de la familia OWA [Yag88, Yag93] bajo el concepto de mayoría difusa.

Tras el estudio de los MRDs propuestos para algunos conjuntos de ejemplos conocidos y un método de aprendizaje concreto, analizamos los resultados alcanzados con otros procesos de generación de la BC y, finalmente, proponemos un proceso de aprendizaje de los parámetros asociados a los MRDs.

2.1 Notación

El MRD es el proceso que se aplica cuando el SCBRD debe tomar una decisión sobre la clase a la cual pertenece un patrón de datos admisible cualquiera. Para describir este proceso de clasificación que lleva a cabo el MRD en un SCBRD, utilizaremos la siguiente notación:

- Sea e el ejemplo a clasificar descrito por N variables, es decir, $e = (e_1, \ldots, e_N)$.
- Consideraremos un problema de clasificación con M clases, por lo que el conjunto de clases será $C = \{C_1, \ldots, C_M\}$.
- La BR contendrá L reglas $R = \{R_1, \ldots, R_L\}$ de cualquiera de los tres tipos descritos en el Capítulo 1:
  - *Reglas difusas tipo (a)*

$$R_k: \text{ Si } X_1 \text{ es } A_1^k \text{ y } \ldots \text{ y } X_N \text{ es } A_N^k \text{ entonces } Y \text{ es } C_j$$

- *Reglas difusas tipo (b)*

$$R_k: \text{ Si } X_1 \text{ es } A_1^k \text{ y } \ldots \text{ y } X_N \text{ es } A_N^k \text{ entonces } Y \text{ es } C_j \text{ con grado } r^k$$

- *Reglas difusas tipo (c)*

$$R_k: \text{ Si } X_1 \text{ es } A_1^k \text{ y } \ldots \text{ y } X_N \text{ es } A_N^k \text{ entonces } (r_1^k, \ldots, r_M^k)$$

El proceso de inferencia se describirá de forma independiente al tipo de regla utilizado en la BR. Para ello, en las reglas tipo (a) se considerará un grado de certeza asociado a la clasificación en las distintas clases, de la siguiente forma:

$$\begin{array}{l} r_v^k = 0, \quad \forall v = 1, \ldots, M, \quad v \neq j \\ r_j^k = 1 \end{array}$$

y para las reglas tipo (b),

$$\begin{array}{l} r_v^k = 0, \quad \forall v = 1, \ldots, M, \quad v \neq j \\ r_j^k = r^k \end{array}$$

- Utilizaremos los siguientes índices:
  - i, para las variables ($i \in \{1, \ldots, N\}$).
  - j y v, para las clases ($j, v \in \{1, \ldots, M\}$).
  - k, para las reglas ($k \in \{1, \ldots, L\}$).

Supongamos que en la BR existen L_j reglas que indican en el consecuente la clase C_j. Notése que en una BR formada por reglas tipo (c), al incluir en el consecuente un grado de certeza asociado a la clasificación en cada una de las clases, $L_j = L \ \ \forall j = 1, \ldots, M$. De esta forma, la BR se puede considerar dividida de la siguiente forma

$$R = R_{C_1} \cup R_{C_2} \cup \ldots \cup R_{C_M}$$

con $|R_{C_1}| = L_1, \ldots, |R_{C_M}| = L_M$.

2.2 Método de Razonamiento Difuso Clásico

Para un ejemplo $e = (e_1, \ldots, e_N)$, el MRD clásico considera la regla con un valor más alto de combinación entre el grado de compatibilidad del ejemplo con el antecedente de la regla y el grado de certeza para las clases, y clasifica el ejemplo con la clase para la que se obtiene este valor. Podemos describir este procedimiento mediante los siguientes pasos:

1. Para cada regla R_k, se calcula el **grado de compatibilidad del ejemplo con el antecedente de la regla**, $R^k(e)$, aplicando un operador de conjunción T a los grados de pertenencia de cada atributo del ejemplo a los conjuntos difusos correspondientes
$$R^k(e) = T(\mu_{A_1^k}(e_1), \ldots, \mu_{A_N^k}(e_N)), \quad k = 1, \ldots L$$

2. El **grado de asociación del ejemplo con cada una de las clases según cada una de las reglas**
$$h(R^k(e), r_j^k), \quad j = 1, \ldots, M, \quad k = 1, \ldots, L$$
se determina mediante un operador de combinación aplicado al grado de compatibilidad del ejemplo con la regla $R^k(e)$ y al grado de certeza de la clasificación del ejemplo en la clase según esa regla, r_j^k.

3. Para cada clase C_j se calcula el **grado de asociación del ejemplo con la clase**, Y_j, mediante la siguiente expresión:
$$Y_j = \max_{k \in L_j} h(R^k(e), r_j^k), \quad j = 1, \ldots, M$$
Este grado de asociación representa la "fuerza" de la clasificación del ejemplo e en la clase C_j.

4. Se asigna al ejemplo e la clase C_v que tiene mayor grado de asociación con el ejemplo, es decir, la clase C_v que verifica que
$$Y_v = \max_{j=1,\ldots,M} Y_j$$

En definitiva, el MRD clásico considera la regla con mayor combinación entre el grado de compatibilidad del ejemplo con el antecedente y el mayor grado de certeza para las clases. Clasifica con la clase para la cual se alcanza ese valor máximo, por lo que se utiliza una única regla *–la regla ganadora–* en el proceso de inferencia. Su funcionamiento se muestra gráficamente en la figura 2.1.

Este MRD clasifica a partir de la información aportada por una única regla y desperdicia la información asociada a todas aquellas reglas cuyo grado de compatibilidad con el ejemplo es menor que el grado de compatibilidad de la regla seleccionada. En definitiva, si consideramos sólo la regla con mayor grado de compatibilidad, estamos considerando un único subconjunto difuso para cada valor de los atributos, y no tenemos en cuenta la información de otras reglas con otros subconjuntos difusos que contienen el valor del atributo en su soporte, y que presentan, por tanto, un grado de compatibilidad positivo con el ejemplo aunque de menor cuantía. Esto se puede observar en la figura 2.2: se utiliza para la clasificación sólo la información que proporciona para cada atributo, la etiqueta de la regla ganadora, sin considerar la información de otras etiquetas cuyos subconjuntos difusos también contienen el valor del atributo en su conjunto soporte. En el gráfico está marcada la etiqueta correspondiente a la regla ganadora (etiqueta M) y la información que no utiliza el MRD (etiquetas B y A con parte del soporte marcado con línea contínua y punteada).

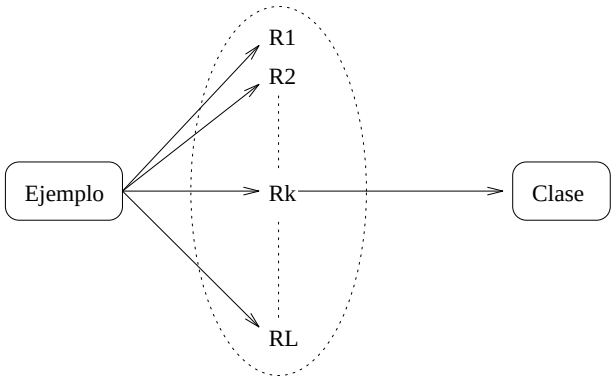

Figura 2.1. Esquema del funcionamiento del Método de Razonamiento Difuso clásico

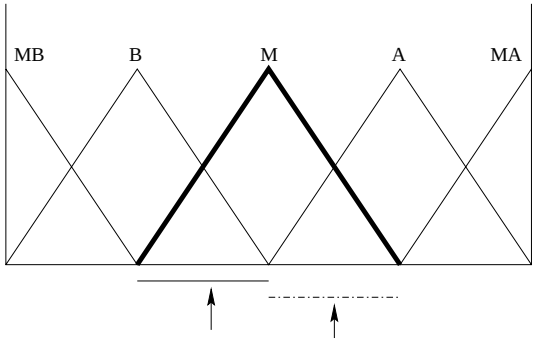

Figura 2.2. Información utilizada por el MRD clásico

2.3 Modelo General de Inferencia

En esta sección presentamos un modelo general de razonamiento difuso que combina la información proporcionada por las reglas difusas compatibles con el ejemplo. Dentro de él se incluye la definición de clasificador difuso presentada por Kuncheva en [Kun96].

En el proceso de clasificación del ejemplo $e = (e_1, \ldots, e_N)$, los pasos del modelo general de un MRD son los siguientes:

1. **Calcular el grado de compatibilidad del ejemplo con el antecedente de las reglas.** Esto implica determinar la *fuerza de la activación del antecedente de todas las reglas de la BR con el ejemplo* e, aplicando una t-norma [ATV83, DP85] a los grados de pertenencia de los atributos del ejemplo (e_i) a los conjuntos difusos correspondientes (A_i^k).

$$R^k(e) = T(\mu_{A_1^k}(e_1), \ldots, \mu_{A_N^k}(e_N)), \quad k = 1, \ldots, L$$

2. **Calcular el grado de asociación del ejemplo** e **con las** M **clases según cada regla de la BR**. Para ello se combina el grado de compatibilidad del ejemplo con

el antecedente de la regla, con el grado de certeza asociado a la clasificación en esa clase según la regla.

$$b_j^k = h(R^k(e), r_j^k), \quad j = 1, \ldots, M, \quad k = 1, \ldots, L$$

3. **Aplicar una función de ponderación a los grados de asociación obtenidos en el paso anterior**. Para esta función de ponderación g parece adecuada una expresión que incremente los valores más altos y penalice los más bajos dentro de este proceso de inferencia.

$$B_j^k = g(b_j^k), \quad j = 1, \ldots, M, \quad k = 1, \ldots, L$$

4. **Determinar el grado de asociación del ejemplo con las distintas clases.** Una vez conocida toda la información que nos proporciona cada una de las reglas sobre la clasificación del ejemplo en las distintas clases, es necesario agregarla y obtener para cada clase, el grado de certeza de la clasificación en la misma, al que denominamos grado de asociación del ejemplo con la clase. Para calcularlo utilizaremos una función de agregación [ATV83, DP85] que combine, para cada clase, los grados de asociación positivos calculados en el paso anterior y proporcione el grado de fuerza para la clasificación del ejemplo en esa clase.

   Esta función de agregación tomará como información de entrada para cada clase C_j los grados de asociación ponderados no nulos entre el ejemplo y la clase C_j de acuerdo a todas las reglas de la BR, es decir

$$(a_1, \ldots, a_{s_j}) = (B_j^k > 0, \ k = 1, \ldots, L)$$

   siendo s_j el número de grados de asociación no nulos entre el ejemplo y la clase C_j.

   De esta forma, el grado de asociación entre el ejemplo y la clase C_j se obtiene aplicando la función de agregación f a estos valores.

$$Y_j = f(a_1, \ldots, a_{s_j})$$

   siendo f un operador de agregación con comportamiento entre el mínimo y el máximo.

   Hay que notar en este punto que, si seleccionamos para la función f la expresión del operador máximo, tendremos el MRD clásico, es decir el MRD basado en la regla ganadora que considera, en el proceso de clasificación, sólo la regla con mayor grado de asociación con el ejemplo.

5. **Clasificación.** El problema de clasificación, una vez calculado el grado de asociación del ejemplo con las distintas clases, se convierte en un problema de decisión. Por ello aplicaremos una función de decisión F sobre el grado de asociación del

ejemplo con las clases que determinará, en base al criterio del máximo, la etiqueta de clase v a la que corresponda el mayor valor.

$$C_v = F(Y_1, \ldots, Y_M) \qquad \text{tal que} \qquad Y_v = \max_{j=1,\ldots,M} Y_j$$

2.4 Métodos de Razonamiento Difuso Alternativos

Según el modelo general de inferencia descrito en la sección anterior, son cuatro los aspectos que se deben definir para determinar un MRD dentro de un SCBRD:

- la forma de obtención del grado de compatibilidad de un ejemplo con el antecedente de una regla,
- la forma de obtención del grado de asociación de un ejemplo con una clase, según una regla,
- la expresión de la función de ponderación, y
- la expresión de la función de agregación para el cálculo del grado global de asociación de un ejemplo con una clase.

La función de agregación es el aspecto más determinante del carácter del MRD que la incluye. Como se ha mencionado en la sección anterior, si seleccionamos para la función de agregación la expresión del máximo, estamos utilizando el MRD clásico. Este método considera sólo la regla con grado de asociación más alto y clasifica en función de ella. Este MRD se describe gráficamente en la figura 2.1, en la que la regla R_k representa la regla con grado de asociación más alto.

La expresión de la función de agregación que representa el MRD clásico es la siguiente:

$$f_0(a_1, \ldots, a_{S_j}) = \max_{m=1,\ldots,S_j} a_m$$

siendo $a_1, \ldots, a_{S_j}$ los valores a agregar para un ejemplo e respecto a la clase C_j.

Con el objetivo de aprovechar toda la potencia del Razonamiento Aproximado considerando toda la información de la zona del espacio en la que se encuentra el ejemplo, en esta sección proponemos MRDs basados en funciones de agregación que combinan la información aportada por las reglas reglas compatibles con el ejemplo de dos formas distintas:

- considerando la información aportada por *todas* las reglas compatibles con el ejemplo para su clasificación, o

- seleccionando las reglas que intervendrán en el proceso de inferencia.

Estos dos modelos de inferencia están basados en funciones de agregación con distintas características.

A continuación describiremos los aspectos del modelo general de inferencia comunes a ambos tipos de modelos de inferencia:

- la forma de obtención del grado de compatibilidad de un ejemplo con una regla,
- el cálculo del grado de asociación, y
- la expresión de la función de agregación.

En las dos secciones siguientes analizaremos los dos modelos de inferencia, distintas propuestas incluidas dentro de ellos y los resultados obtenidos con BCs generadas con la extensión del proceso de generación de Wang y Mendel a problemas de clasificación para tres bases de ejemplos conocidas. Finalmente, para observar la independencia de los resultados del proceso de aprendizaje, mostraremos los resultados obtenidos con los MRDs propuestos con BCs obtenidas con otros procesos de aprendizaje inductivo descritos en los Capítulos 1 y 3.

2.4.1 Grado de Compatibilidad de un Ejemplo con el Antecedente de una Regla

Uno de los primeros aspectos a determinar en la definición de un MRD es la forma de obtener el grado de compatibilidad de un ejemplo con el antecedente de una regla, es decir, la forma de cálculo del grado con el cual el ejemplo pertenece a la zona del espacio delimitada por el antecedente de la regla. Este grado se calculará como una conjunción del grado de pertenencia de los atributos del ejemplo a los subconjuntos difusos especificados en el antecedente de la regla y definidos en la BC del SCBRD.

En la literatura hay distintas propuestas para este operador de conjunción. Ishibuchi y otros utilizan y analizan el comportamiento de las t-normas mínimo y producto [INT92], González y Pérez utilizan el mínimo [GP98], y Mandal y otros en [MMP92] por una parte, y Uebele y otros en [UAL95] por otra, calculan el grado de compatibilidad mediante la media aritmética. En nuestros experimentos hemos calculado $R^k(e)$ utilizando la t-norma mínimo. De esta forma, el grado de compatibilidad tiene esta expresión:

$$R^k(e) = \min_{i=1,\ldots,N} \mu^k_{A_i}(e_i)$$

2.4.2 Grado de Asociación de un Ejemplo con una Clase

Una regla representa la información extraída sobre una zona del espacio. En esta zona pueden existir distintas proporciones de ejemplos de las distintas clases y, en función del tipo de regla elegido para la construcción de la BC, estará representada en el consecuente la clase mayoritaria o todas las clases presentes. En cualquier caso, una vez calculado el grado de pertenencia del ejemplo a la zona del espacio representada por el antecedente de la regla, es necesario combinar éste con el grado de certeza de la regla para predecir cada una de las clases y obtener así el grado de asociación *local* del ejemplo con cada una de las clases.

La función h proporciona, para una regla dada, este grado de asociación del ejemplo con las clases, en función del grado de compatibilidad del ejemplo con el antecedente y de la fuerza de la regla para predecir las distintas clases.

En [BD95], Bardossy y Duckstein utilizan para esta función h los operadores de combinación mínimo y producto e Ishibuchi y otros en [INT92] el producto.

El operador mínimo encuentra el mínimo de cualquier par de elementos y por tanto proporciona información conectiva. Si el mínimo de los dos elementos se mantiene igual, pero el valor del otro se incrementa, este cambio no se refleja en el resultado del operador mínimo. El operador producto refleja el cambio en cualquier argumento de una forma más evidente. Por todo esto, en nuestros experimentos la función h utilizada es el operador producto.

$$h(R^k(e), r_j^k) = (R^k(e) \cdot r_j^k)$$

2.4.3 Función de Ponderación

El modelo general de MRD propuesto combina la información proporcionada por *todas* las reglas compatibles con el ejemplo a clasificar. Es necesario potenciar en esta combinación grados de asociación altos para evitar que en la operación de agregación muchas reglas que aportan un grado de asociación ejemplo-clase bajo, tengan mayor fuerza que una mejor en el proceso de inferencia. En la literatura no existe ningún método de inferencia que considere este tipo de ponderación sobre los valores obtenidos por las reglas.

Para observar el efecto de la ponderación, en esta experimentación hemos utilizado dos funciones de ponderación:

$$g_1(x) = x \quad \forall x \in [0, 1]$$

que deja los grados de asociación sin ponderar, y

$$g_2(x) = \begin{cases} x^2, & \text{si } x < 0.5 \\ \sqrt{x}, & \text{si } x \geq 0.5 \end{cases}$$

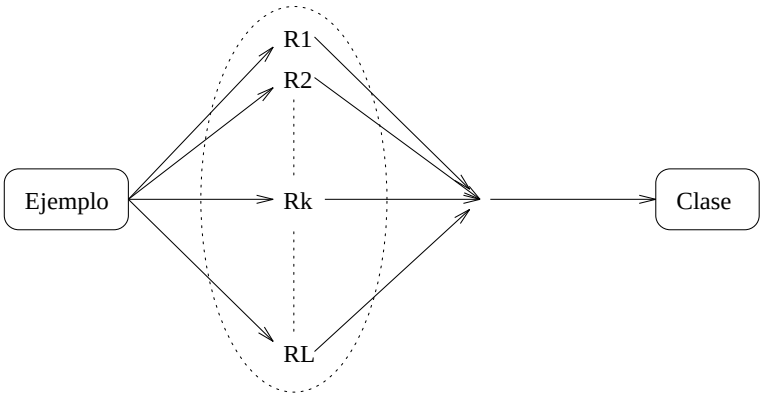

Figura 2.3. Esquema del funcionamiento global de los Métodos de Razonamiento Difuso que integran todas las reglas

que potencia los grados de asociación superiores a 0.5 y penaliza los inferiores a este valor.

En el modelo general de inferencia propuesto se puede incluir como función de ponderación otra función creciente que potencie / penalice valores en el intervalo $[0, 1]$.

2.5 Métodos de Razonamiento Difuso que Integran todas las Reglas

El primer grupo de MRDs alternativos al clásico está formado por los MRDs que utilizan la información proporcionada por todas las reglas difusas para derivar una conclusión entre un conjunto de reglas difusas y un ejemplo. Esta idea se describe gráficamente en la figura 2.3.

Para conseguir estos MRDs alternativos debemos analizar el comportamiento de distintos operadores de agregación y utilizar aquellos que combinen la información proporcionada por todas las reglas de una forma adecuada para un SCBRD.

2.5.1 Funciones de Agregación

Los operadores de agregación desempeñan un papel importante en la teoría de conjuntos difusos. Los más importantes son los conectivos de conjunción y disyunción. Estos operadores están caracterizados formalmente por las t-normas y t-conormas, y la definición de los mismos por defecto –dentro de la teoría de conjuntos difusos– son los operadores mínimo y máximo.

Es claro que si se eligiese como operador de agregación el máximo, estaríamos ante el MRD clásico, con distintas alternativas para la obtención del grado de compatibilidad de un ejemplo con una regla y del grado de asociación de un ejemplo con una clase.

El caso del mínimo ha sido considerado por Bardóssy y Duckstein en [BD95] y su interpretación verbal es la siguiente: se clasifica con el elemento que contradice todas las reglas aplicables en menor grado. Se intenta encontrar una respuesta que tenga al menos un cierto grado de acuerdo con todas las reglas aplicables. No es un criterio válido para Sistemas de Clasificación en los que se tenga que justificar la decisión tomada.

De cualquier forma, en cualquiera de los dos casos sólo se está considerando una regla y se clasifica de acuerdo a ella, ignorando la información que proporcionan otras reglas difusas cuyos antecedentes contienen también al ejemplo en sus conjuntos soportes, aunque en menor grado.

Nuestro objetivo es encontrar una función de agregación que considere, para cada clase, *todas* las reglas aplicables al ejemplo y agregue esa información para dar como salida un valor de la certeza del SCBRD para clasificar el ejemplo como perteneciente a esa clase. Finalmente, entre todos esos valores de certeza, se buscará el máximo y se clasificará con la etiqueta de clase correspondiente.

A continuación describiremos nuestras propuestas para esta función de agregación que toma como entrada, no todos los grados de asociación ponderados correspondientes a todas las reglas sino aquellos que tienen un valor positivo, es decir $(a_1, \ldots, a_{s_j})$, siendo C_j la clase para la cual se va a calcular el grado de asociación.

1. Suma normalizada. Esta función acumula, para todas las reglas, el grado de asociación del ejemplo con la clase C_j. Finalmente divide esta suma, entre la suma máxima para todas las clases para obtener un valor normalizado entre 0 y 1.

$$f_1(a_1, \ldots, a_{s_j}) = \frac{\sum_{m=1}^{s_j} a_m}{f_{1_{max}}}$$

donde

$$f_{1_{max}} = \max_{v=1,\ldots,M} \sum_{m=1}^{s_v} a_m$$

y $(a_1, \ldots, a_{s_v})$ son los grados de asociación ponderados del ejemplo e con la clase C_v de acuerdo a las reglas de la BR.

Bárdossy y Duckstein utilizan este MRD en [BD95] para combinar respuestas de reglas difusas y lo denominan *enfoque aditivo*. Chi y otros presentan este MRD en [CYP96] como un método de defuzzificación para obtener una clasificación en SCBRDs y le asignan el nombre de *compatibilidad máxima acumulada*. Ishibuchi y otros describen este MRD en [IMN96] como un proceso de inferencia en el que el resultado viene dado por la "votación"

de las reglas difusas compatibles con el ejemplo, por lo que lo denominan *método de razonamiento basado en el voto máximo.* Ishibuchi combina la información de las reglas mediante una suma no normalizada y lo utiliza tanto para el proceso de inferencia de un SCBRD, como en el caso de multiclasificadores.

2. Media aritmética.

$$f_2(a_1, \ldots, a_{s_j}) = \frac{\sum_{m=1}^{s_j} a_m}{s_j}$$

La media aritmética es un operador con un grado de compensación entre el máximo y el mínimo que se utiliza para sintetizar juicios dentro de procesos de toma de decisión multicriterio. El uso de este operador es adecuado para combinar la información proporcionada por cada clasificador local (cada regla) y obtener un grado medio que considere la calidad de las reglas en el proceso de inferencia.

3. Media quasiaritmética. El operador media quasiaritmética es una función estrictamente monótona y continua definida mediante la siguiente expresión:

$$f_3(a_1, \ldots, a_{s_j}) = H^{-1}\left[\frac{1}{s_j}\sum_{m=1}^{s_j} H(a_m)\right]$$

siendo H una función continua estrictamente monótona. En nuestros experimentos hemos adoptado la siguiente expresión para la función H,

$$H(x) = x^p, \quad p \in \mathrm{R}$$

que proporciona un efecto compensativo en los valores a agregar [DP84, DP85] y hace que la función f_3 tome la forma del operador media generalizada con pesos igual a la unidad para todos los valores a agregar, con un grado de compensación entre el mínimo y el máximo, incluyendo a las medias aritmética y geométrica. Concretamente,

$$\begin{array}{ll} \text{Si } p \longrightarrow -\infty, & f_3 \longrightarrow \min \\ \text{Si } p \longrightarrow +\infty, & f_3 \longrightarrow \max \end{array}$$

En [FMR95] y [DP84] podemos encontrar un estudio detallado sobre las propiedades y comportamiento de este operador de agregación.

4. SOWA And-Like. En [YF94], Yager propone la utilización, dentro de procesos de defuzzificación (que se pueden ver como procesos de agregación de los elementos de un subconjunto difuso en un elemento representativo), de algunos operadores de agregación con comportamiento entre el mínimo y el máximo. Son los operadores SOWA And-Like,

SOWA Or-Like y Badd, que presentan el comportamiento que buscamos (entre el mínimo y el máximo) y pueden ser integrados en nuestros MRDs.

El operador SOWA And-Like tiene la siguiente expresión:

$$f_4(a_1, \ldots, a_{s_j}) = \alpha \cdot a_{min} + \frac{1}{s_j}(1-\alpha) \sum_{m=1}^{s_j} a_m$$

donde $\alpha \in [0,1]$ y $a_{min} = \min\{a_1, \ldots a_{s_j}\}$.

Este operador tiene un comportamiento entre la media aritmética y el mínimo.

Si $\alpha = 0$ entonces f_4 es la media aritmética
Si $\alpha = 1$ entonces f_4 es el mínimo

5. SOWA Or-Like. El operador SOWA Or-Like se define de la siguiente forma:

$$f_5(a_1, \ldots, a_{s_j}) = \alpha \cdot a_{max} + \frac{1}{s_j}(1-\alpha) \sum_{m=1}^{s_j} a_m$$

con $\alpha \in [0,1]$ y $a_{max} = \max\{a_1, \ldots, a_{s_j}\}$.

En este caso, y en función del valor de α, el operador devolverá un valor entre los siguientes extremos

Si $\alpha = 0$ entonces f_5 es la media aritmética
Si $\alpha = 1$ entonces f_5 es el máximo

En clasificación es adecuado agregar la mayoría de la información que nos proporcionan las reglas por lo que, para las dos últimas funciones de agregación, se determinarán valores para α que den un comportamiento al operador, en el caso de f_4, alejado del mínimo y, entre el máximo y la media, para f_5.

6. Badd. La expresión del operador Badd es:

$$f_6(a_1, \ldots, a_{s_j}) = \frac{\sum_{m=1}^{s_j} (a_m)^{p+1}}{\sum_{m=1}^{s_j} (a_m)^p}$$

con $p \in I\!R$.

La función Badd proporciona una agregación entre el máximo y el mínimo en función del valor de p:

$$\begin{array}{ll} \text{Si } p \longrightarrow +\infty, & f_6 \longrightarrow \text{ máximo} \\ \text{Si } p = 0, & f_6 = \text{ media aritmética} \\ \text{Si } p \longrightarrow -\infty, & f_6 \longrightarrow \text{ mínimo} \end{array}$$

De nuevo, como estamos interesados en una agregación entre la media y el máximo, en nuestros experimentos consideraremos $p \in I\!R^+$.

En [YF94] se puede encontrar información sobre propiedades y comportamiento de los operadores en los que se basan las funciones f_4, f_5 y f_6.

La tabla 2.1 resume la expresión de las funciones de agregación propuestas que determinan el carácter de los MRDs que las incluyen.

f_i	Operador	Expresión	Parámetros
f_0	Clásico	$\max\{a_1, \ldots, a_{s_j}\}$	—
f_1	Suma	$\frac{\sum_{m=1}^{s_j} a_m}{f_{1_{max}}}$	—
f_2	Media aritmética	$\frac{\sum_{m=1}^{s_j} a_m}{s_j}$	—
f_3	Media quasiaritmética	$H^{-1}\left[\frac{1}{s_j}\sum_{m=1}^{s_j} H(a_m)\right]$	$H(x) = x^p,\ p \in I\!R$
f_4	SOWA And-Like	$\alpha \cdot a_{min} + \frac{1}{s_j}(1-\alpha)\sum_{m=1}^{s_j} a_m$	$\alpha \in [0,1]$
f_5	SOWA Or-Like	$\alpha \cdot a_{max} + \frac{1}{s_j}(1-\alpha)\sum_{m=1}^{s_j} a_m$	$\alpha \in [0,1]$
f_6	Badd	$\frac{\sum_{m=1}^{s_j}(a_m)^{p+1}}{\sum_{m=1}^{s_j}(a_m)^p}$	$p \in I\!R$

Tabla 2.1. Operadores de agregación

2.5.2 Experimentación y Análisis de Resultados

Para analizar el comportamiento de los MRDs propuestos, hemos trabajado con tres conjuntos de ejemplos utilizados con mucha frecuencia en el análisis experimental de Sistemas de Clasificación: Iris, Pima y Wine. Las tres bases de ejemplos se pueden obtener en el *UCI Repository of Machine Learning Databases and Domain Theories* a través de la dirección ftp `ftp.ics.uci.edu`, en el directorio `/pub/machine-learning-databases`. A continuación las describiremos brevemente.

Iris. La base de ejemplos de clasificación de plantas de Iris es una de las más conocidas y utilizadas para mostrar el comportamiento de Sistemas de Clasificación. Fue utilizada por primera vez por Fisher [Fis36] y propone la clasificación de tres clases de plantas, *setosa,*

virginica y versicolor, a través de cuatro variables predictivas cuyo rango se describe en la tabla 2.2.

Variable	Rango
Longitud del sépalo	[4.3, 7.9]
Anchura del sépalo	[2, 4.4]
Longitud del pétalo	[1, 6.9]
Anchura del pétalo	[0.1, 2.5]

Tabla 2.2. Características de las variables de la base de ejemplos Iris

El conjunto de datos está formado por 150 muestras, 50 de cada clase. La clase *setosa* es linealmente separable de las otras dos, mientras que éstas no son linealmente separables entre ellas.

Pima. La base de ejemplos Pima es un conjunto de 768 muestras de un problema de diagnóstico en el que, a partir de un conjunto de valores de variables propuestas por la Organización Mundial de la Salud, se determina si el paciente sufre o no diabetes. Los datos fueron recogidos por investigadores del *National Institute of Diabetes and Digestive* de EE.UU. en una población de mujeres indias de la tribu pima. Cada uno de los ejemplos se describe a través de ocho variables descritas en la tabla 2.3.

Variable	Rango
Número de embarazos	[0, 17]
Concentración de glucosa en plasma	[0, 199]
Presión diastólica de la sangre	[0, 122]
Grosor de los dobleces de la piel de los triceps	[0, 99]
Nivel de insulina	[0, 846]
Indice corporal de masa	[0, 67.1]
Estimación de la influencia genética	[0.08, 2.42]
Edad	[21, 81]

Tabla 2.3. Características de las variables de la base de ejemplos Pima

Los datos se reparten en dos clases 1 y 0 según el individuo padezca o no la enfermedad. De los 768 ejemplos, 268 pertenecen a la clase 1 y 500 a la 0.

Wine. Esta base de ejemplos contiene 178 muestras resultado de un análisis químico de vinos de una misma región obtenidos a partir de cepas de orígenes distintos y plantea

un problema de clasificación en tres clases. Para ello, cada una de las muestras contiene datos sobre 13 variables contínuas cuyo rango se describe en la tabla 2.4.

Variable	Rango
1	[11, 15]
2	[0.74, 5.8]
3	[1.36, 3.23]
4	[10.6, 30]
5	[70, 162]
6	[0.9, 3.88]
7	[0.34, 5.08]
8	[0.13, 0.66]
9	[0.41, 3.58]
10	[1.2, 13]
11	[0.48, 1.71]
12	[1.27, 4]
13	[278, 1680]

Tabla 2.4. Características de las variables de la base de ejemplos Wine

Los 178 ejemplos están distribuidos en 59, 71 y 48 ejemplos de las clases 1, 2 y 3 respectivamente.

Para cada una de estas bases de ejemplos, hemos generado bases de reglas de los tres tipos de reglas difusas, (a), (b) y (c), descritos en el Capítulo 1.

El proceso de aprendizaje inductivo utilizado para generar las BRs de cada uno de los tipos indicados es la extensión del proceso de generación de reglas de Wang y Mendel [WM92] a problemas de clasificación (ya descrito en el Capítulo 1). A este proceso de aprendizaje inductivo lo denominaremos Método 1.

Para calcular una estimación del error asociado a los distintos SCBRDs hemos utilizado el método de *remuestreo aleatorio* ([WK91]) con cinco particiones aleatorias de la base de ejemplos en conjuntos de entrenamiento y test (70% y 30%, respectivamente). Los resultados mostrados son medias del porcentaje de clasificación correcta en cada caso.

Respecto a la notación, hay que aclarar que las columnas encabezadas con f especifican el MRD con el que se han obtenido los resultados de la fila. El MRD viene identificado por la función de agregación en la cual se basa (f_0, si es el clásico; f_1, si es la suma normalizada, y así sucesivamente). En la columna encabezada por p se indican los parámetros -si los tiene- del MRD. Por último, las columnas encuadradas bajo el epígrafe g_1 describen los resultados del MRD correspondiente con la función de ponderación g_1 (es decir, sin ponderación) y las encuadradas bajo g_2, los obtenidos con el MRD y la función de ponderación g_2.

f	p	Pr. g_1	Pr. g_2	f	p	Pr. g_1	Pr. g_2	f	p	Pr. g_1	Pr. g_2
f_0		88.25	—	f_0		94.32	—	f_0		94.32	—
f_1		92.88	**94.38**	f_3	20	94.32	94.32	f_3	20	94.32	94.32
f_5	0.5	90.83	92.27	f_5	0.5	94.32	94.32	f_5	0.7	94.32	94.32
f_3	20	90.34	91.78	f_6	5	94.32	94.32	f_6	20	94.32	94.32
BR tipo (a) (40.4 reglas)				BR tipo (b) (40.4 reglas)				BR tipo (c) (37.4 reglas)			

Tabla 2.5. Porcentajes de clasificación sobre datos de prueba de la base de ejemplos de Iris para BRs obtenidas con el Método 1

f	p	Pr. g_1	Pr. g_2	f	p	Pr. g_1	Pr. g_2	f	p	Pr. g_1	Pr. g_2
f_0		64.88	—	f_0		73.23	—	f_0		74.16	—
f_1		**72.11**	70.97	f_3	50	73.33	73.44	f_5	0.3	73.95	74.47
f_5	0.1	59.63	64.56	f_6	5	73.33	**73.53**	f_6	10	74.47	74.58
f_6	50	61.42	67.38	f_6	10	**73.53**	73.43	f_6	20	**74.68**	74.27
BR tipo (a) (420.2 reglas)				BR tipo (b) (420.2 reglas)				BR tipo (c) (382 reglas)			

Tabla 2.6. Porcentajes de clasificación sobre datos de prueba de la base de ejemplos de Pima para BRs obtenidas con el Método 1

En las tablas 2.5, 2.6 y 2.7 mostramos un resumen de los mejores resultados obtenidos para datos de prueba con distintos MRDs para las bases de ejemplos Iris, Pima y Wine. Las tablas con todos los resultados se muestran en el Apéndice 1, situado al final de este capítulo. En negrita se señala el mejor resultado para cada problema y tipo de BR. En las tres tablas hemos incluido el resultado proporcionado por el MRD clásico para poder efectuar una comparación con las nuevas propuestas.

Si analizamos los resultados de los experimentos de acuerdo al tipo de BR podemos observar lo siguiente:

- En los tres conjuntos de ejemplos el MRD que tiene mejores resultados, con BRs tipo (a), es el basado en la suma (f_1). Esta función es la mejor forma de agregar la información proporcionada por las reglas disparadas cuando éstas no proporcionan información sobre el grado de precisión asociado a la clasificación en una clase.

- Para BRs tipo (b) o (c), los MRDs basados en los operadores Badd (f_6), SOWA Or-Like (f_5) y Media Quasiaritmética (f_3) mejoran los resultados del MRD clásico.

f	p	Pr. g_1	Pr. g_2	f	p	Pr. g_1	Pr. g_2	f	p	Pr. g_1	Pr. g_2
f_0		88.36	—	f_0		91.94	—	f_0		91.94	—
f_1		**92.81**	91.29	f_1		92.05	91.50	f_1		92.73	91.94
f_3	50	81.86	83.99	f_3	20	91.96	91.94	f_3	5	92.76	92.29
f_6	20	84.42	84.42	f_5	0.9	**92.29**	91.94	f_5	0.5	**92.97**	91.97
BR tipo (a) (102 reglas)				BR tipo (b) (102 reglas)				BR tipo (c) (101.4 reglas)			

Tabla 2.7. Porcentajes de clasificación sobre datos de prueba de la base de ejemplos de Wine para BRs obtenidas con el Método 1

Respecto a la función de agregación, los experimentos muestran que, para el proceso de aprendizaje y bases de ejemplos considerados,

- La media aritmética (f_2) tiene demasiada compensación entre grados altos y bajos de asociación entre un ejemplo y una clase.
- El MRD basado en el operador SOWA And-Like (f_4) proporciona peores resultados que el resto por su comportamiento entre la media aritmética y el mínimo.
- Los MRDs basados en los operadores media quasiaritmética (f_3), SOWA Or-Like (f_5) y Badd (f_6) proporcionan mejores resultados que el MRD clásico y el MRD basado en la suma, pero no hay un MRD que tenga mejor comportamiento para toda base de ejemplos y tipos de reglas considerados.

La función de ponderación (g_2) mejora, en la mayoría de los casos, la capacidad de predicción del SCBRD pero no tiene un comportamiento uniforme. Esto es debido fundamentalmente a tres factores:

1. *Las características de la base de ejemplos considerada.* Generalmente, en problemas de clasificación en los que las variables tienen un rango de definición amplio, independientemente del proceso de aprendizaje inductivo utilizado, la BR obtenida no cubrirá algunas zonas del espacio de patrones. Los ejemplos de prueba pertenecientes a estas zonas tendrán un grado de asociación muy pequeño que, al ser ponderado, puede anularse llevando a una situación en la que el ejemplo no se puede clasificar. En estos casos, el sistema funciona peor con la función de ponderación propuesta.
2. *Las características del proceso de generación de BRs.* La deficiencia mencionada en el punto anterior se puede acentuar en función del proceso de aprendizaje inductivo utilizado. El método de generación considerado en esta experimentación obtiene BRs muy ajustadas a los ejemplos de entrenamiento ya que va generando

la regla "más compatible" para cada ejemplo. Esto hace que en bases de ejemplos como Wine y Pima se observen con más claridad los efectos negativos de la función de ponderación propuesta de nuevo por disminuir la capacidad de clasificación del sistema.

Este problema se puede solucionar mediante un algoritmo de búsqueda que determine los mejores parámetros de la función de ponderación para un problema, proceso de aprendizaje y tipo de BR específico. En la Sección 2.8.1 presentamos una propuesta basada en esta idea.

3. *Las características de la función de agregación.* Los parámetros de la función de ponderación también dependen de la función de agregación incluida en el MRD. Algunas funciones de agregación como la media quasiaritmética (f_3) o el operador Badd (f_6) llevan implícita una ponderación en el proceso de agregación que representan.

   Se puede realizar un proceso de aprendizaje de parámetros de la función de agregación paralelo a la optimización de los parámetros de la función de ponderación. En la Sección 2.8.2 proponemos un proceso evolutivo para la optimización de estos parámetros.

Finalmente, podemos señalar que los resultados obtenidos con los conjuntos de ejemplos, tipos de reglas difusas y MRDs considerados muestran que:

- Los MRDs alternativos que utilizan la información proporcionada por todas las reglas compatibles con el ejemplo a clasificar muestran mejor comportamiento que el MRD clásico en la clasificación de un nuevo ejemplo, en todas las bases de ejemplos y tipos de reglas consideradas. Ponemos decir que *los nuevos MRDs mejoran el rendimiento de los SCBRDs.*

- Además, como sabemos, en un Sistema de Clasificación se pueden distinguir dos procesos: El proceso de *abstracción*, es decir, la etapa de extracción de la información proporcionada por los ejemplos de entrenamiento para construir una estructura adecuada para el sistema, y el proceso de *generalización* que, haciendo uso de la información extraída, clasifica cualquier nuevo ejemplo. En esta sección hemos analizado el comportamiento de un SCBRD formado por un MRD y una BR generada de forma independiente al MRD. Los resultados muestran que *los MRDs mejoran la capacidad de generalización del sistema sin intervenir en el proceso de abstracción*, por lo que constituyen una buena herramienta para incrementar el rendimiento de un SCBRD.

- Los resultados obtenidos, incluso con el MRD basado en el operador SOWA And-Like (f_4), que tiene un comportamiento parametrizable entre el mínimo y la media aritmética, muestran que la agregación de la información que proporcionan las reglas disparadas es importante en un proceso de inferencia.

- No hay un conjunto de valores para los parámetros de las funciones de agregación que permita obtener los mejores porcentajes de clasificación correcta en todos los casos.

Si observamos las tablas resumen 2.6, 2.7 o las completas del Apéndice 1 (tablas 2.19, 2.20, 2.21 y 2.22), vemos que para problemas como Pima y Wine, en los que con un método de aprendizaje inductivo como el utilizado, se genera una BR con cardinalidad alta, los MRDs alternativos que consideran la información proporcionada por todas las reglas cometen en ocasiones un mayor número de errores en la clasificación. Esto se debe fundamentalmente a la influencia que pueden tener en el proceso de clasificación muchas reglas con grado de asociación pequeño con el ejemplo, frente a otras que representan mejor la zona a la que pertenece el ejemplo. Para este tipo de problemas y procesos de aprendizaje es necesario un MRD que seleccione la información a agregar en el proceso de inferencia. En la siguiente sección describimos distintas propuestas de MRDs con esta filosofía.

2.6 Métodos de Razonamiento Difuso que Seleccionan Reglas

En el modelo general de inferencia descrito en la Sección 2.3, las funciones propuestas como operadores de agregación pertenecientes al primer tipo de modelo de inferencia (tabla 2.1) consideran la información proporcionada por todas las reglas compatibles con el ejemplo a clasificar. Hemos analizado el incremento de la capacidad de generalización aportada por este tipo de MRDs alternativos al clásico, con distintos conjuntos de ejemplos y tipos de BRs. Sin embargo, en BRs grandes, muchas reglas con un grado de asociación ejemplo-clase bajo pueden tener mayor influencia que otras con un grado de asociación más alto, llevando al sistema a una clasificación errónea. La incorporación de un MRD que considere sólo la información proporcionada por el subconjunto más adecuado para el ejemplo a clasificar puede mejorar el rendimiento del sistema en estas situaciones.

En esta sección describiremos MRDs que consideran, en la clasificación de un ejemplo, únicamente la información aportada por un subconjunto de reglas seleccionadas a través de varios operadores de la familia OWA [Yag88, Yag93] bajo el concepto de mayoría difusa. La figura 2.4 representa gráficamente el funcionamiento de este tipo de MRD.

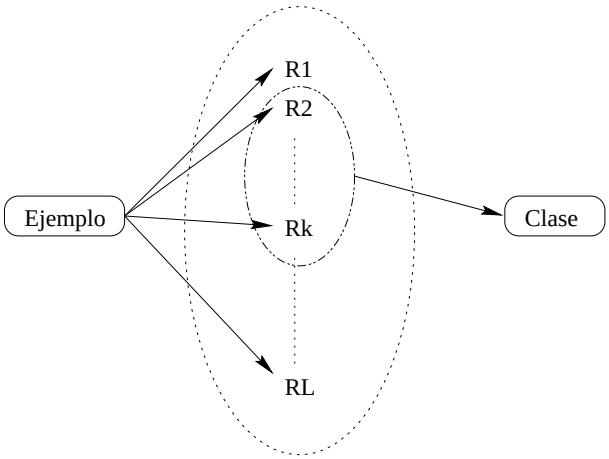

Figura 2.4. Métodos de Razonamiento Difuso que seleccionan un subconjunto de reglas

2.6.1 Métodos de Razonamiento Difuso Basados en Operadores de la Familia OWA bajo el Concepto de Mayoría Difusa

Los operadores OWA (Ordered Weighted Averaging) definidos por Yager [Yag88, Yag93, YF94] constituyen una clase de operadores de agregación que cubre completamente el intervalo entre los operadores mínimo y máximo. Estos operadores se han utilizado en SBRDs para visión por ordenador [MS98], procesos de toma de decisión multicriterio [Yag88], aprendizaje automático [Kac97, Ped97b], multiclasificadores [Kun97], agregación de Redes Neuronales Modulares [Cho95] y agregación de conjuntos difusos en SBRDs para visión por ordenador [MS98], entre otros campos.

La característica más importante de estos operadores es el uso de pesos para los valores a agregar no asociados a sus valores particulares sino a su posición ordenada. Esta propiedad nos permite asignar, dentro del proceso de inferencia, pesos a los grados de asociación del ejemplo con la clase según distintas reglas, de una forma ordenada y así potenciar las reglas más predictivas para el ejemplo. La introducción de un MRD de este tipo en un SCBRD puede decrementar su interpretabilidad lingüística -si lo comparamos con el MRD basado en la regla ganadora- pero incrementa la capacidad de predicción del SCBRD resultante.

A continuación describimos dos MRDs basados en operadores OWA y QuasiOWA.

Métodos de Razonamiento Difuso basados en el operador OWA. Para definir el comportamiento del operador OWA en el proceso de inferencia consideraremos, al igual que en la sección anterior, que $(a_1, \ldots, a_{s_j})$ son los valores a agregar para un ejemplo e, respecto a la clase C_j. Sea $W = (w_1, \ldots, w_{s_j})$ un vector de pesos que verifica las dos condiciones siguientes:

$$\begin{array}{ll} 1) & \omega_m \in [0,1], \quad \forall m = 1, \ldots, s_j \\ 2) & \sum_{m=1}^{s_j} \omega_m = 1 \end{array}$$

La operación de agregación OWA relacionada con el vector de pesos W es la función

$$f_7(a_1, \ldots, a_{s_j}) = \sum_{m=1}^{s_j} \omega_m \cdot b_m$$

donde $(b_1, \ldots, b_{s_j})$ es una permutación del vector $(a_1, \ldots, a_{s_j})$ en la que los elementos están ordenados decrecientemente.

Como resultado, esta función de agregación proporciona el grado de clasificación (o de asociación) del ejemplo e en la clase C_j.

La familia de operadores OWA tiene un conjunto de propiedades [Yag88, Yag93, FMR95] importantes para el proceso de inferencia en el que se van a incluir:

- Como se ha mencionado anteriormente, cualquier operador OWA f_7 verifica que

  $$\min(a_1, \ldots, a_{s_j}) \leq f_7(a_1, \ldots, a_{s_j}) \leq \max(a_1, \ldots, a_{s_j}),$$

  lo que implica que el grado de clasificación del ejemplo e en la clase C_j tendrá un valor entre el grado máximo y el grado mínimo de asociación del ejemplo con la clase C_j. El operador tiene un comportamiento parametrizable entre el mínimo y el máximo.

- Puede verse como un operador conmutativo, ya que el orden inicial de los elementos a agregar no afecta al resultado final de la agregación. Esta propiedad es importante en SCBRDs en los que las reglas no se aplican de forma jerárquica, como los estudiados en esta memoria.

- Verifica la propiedad de idempotencia que proporciona un resultado adecuado en el caso de que todas las reglas compatibles con el ejemplo proporcionen el mismo grado de asociación con una clase.

- Es un operador monótono. Un incremento en uno de los grados de asociación a agregar, sin cambio en el resto, no puede producir un decremento en el valor agregado. Esto asegura que el MRD que incluya este operador tendrá un comportamiento lógico tal que si la fuerza de una regla para predecir una clase se incrementa, el grado de clasificación global no se decrementa.

Se puede comprobar fácilmente que el mínimo, el máximo y la media aritmética son tres casos particulares de esta función de agregación con los siguientes valores para el vector de pesos:

$$\begin{array}{ll} \text{Si } \omega_1 = 1, \text{ y } \omega_m = 0 \quad \forall m \neq 1, & f_7 = \max \\ \text{Si } \omega_{s_j} = 1, \text{ y } \omega_m = 0 \quad \forall m \neq s_j, & f_7 = \min \\ \text{Si } \omega_m = \frac{1}{s_j} \quad \forall m = 1, \ldots, s_j, & f_7 = \text{media aritmética} \end{array}$$

Además, este operador incluye cualquier ordenación estática, la mediana, y la media aritmética excluyendo los dos valores extremos. Se puede encontrar más información sobre las propiedades de esta familia de operadores en [Yag88, Yag93, FMR95, RR97].

Hay que destacar en este punto, que la asignación de pesos se realiza de forma dinámica en cada clasificación, en función de los grados de asociación de las reglas disparadas. No es una asignación de importancia relativa de las reglas que se establece de forma independiente a la clasificación y que permanece constante en cualquier proceso de clasificación como realizan Chi y Yan en [CWY95, CY96], sino que en la clasificación de cada ejemplo se asocian pesos a los grados de asociación en función de la información proporcionada por las reglas disparadas para ese ejemplo.

El comportamiento parametrizable de la clase de operadores OWA está determinado por el vector de pesos W, que tradicionalmente se ha obtenido de dos formas distintas. La primera implica la utilización de un mecanismo de aprendizaje que use un conjunto de ejemplos, sus argumentos y los valores agregados, e intente ajustar los pesos a esa colección de datos mediante algún tipo de modelo de regresión. Otra alternativa, en la cual se centra nuestra propuesta, consiste en dar significado a los pesos. Para asignar contenido semántico a los pesos, calcularemos su valor utilizando *cuantificadores linguísticos* que representen el concepto de *mayoría difusa*.

La mayoría se representa tradicionalmente como un número umbral de elementos. La *mayoría difusa* es una relajación del concepto de mayoría representado normalmente mediante un cuantificador difuso [Zad83].

Los cuantificadores se utilizan para representar la cantidad de items que satisfacen un predicado dado. La lógica binaria clásica permite la representación de dos cuantificadores "al menos uno" y "para todos", muy relacionados con las conectivas "o" e "y" respectivamente. En el lenguaje natural hay muchos más cuantificadores como "casi todos", "pocos", "algunos", "muchos", "tantos como sea posible", "casi la mitad", "al menos la mitad", etc. En un intento tanto de establecer un puente entre los sistemas formales y el discurso humano, como de proporcionar una herramienta de representación del conocimiento más flexible, Zadeh introdujo el concepto de *cuantificador difuso* ([Zad83]). En ese artículo sugiere que la semántica del cuantificador se puede representar con un conjunto difuso y distingue entre dos tipos de cuantificadores: absolutos (por ejemplo, "sobre 2") y proporcionales (como "al menos la mitad"). Un cuantificador absoluto puede representarse como uno proporcional y viceversa. Nosotros trabajaremos con cuantificadores proporcionales no decrecientes. En un cuantificador proporcional, para cualquier $r \in [0, 1]$, $Q(r)$ indica el grado con el cual la proporción r es compatible con el significado que representa el cuantificador. En el caso de los cuantificadores proporcionales no

decrecientes, la función de pertenencia puede representarse de la forma

$$Q(r) = \begin{cases} 0, & \text{si } r < a \\ \frac{r-a}{b-a}, & \text{si } a \leq r \leq b \\ 1, & \text{si } r > b \end{cases}$$

con $a, b, r \in [0, 1]$.

En la figura 2.5, se muestran algunos ejemplos de cuantificadores proporcionales, en los que los parámetros a y b toman los valores (0.3, 0.8), (0, 0.5), y (0.5, 1) respectivamente.

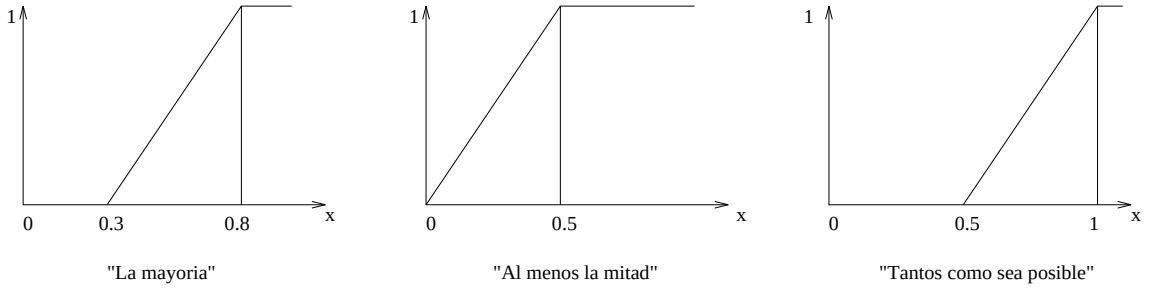

Figura 2.5. Algunos ejemplos de cuantificadores difusos proporcionales

Para un cuantificador difuso proporcional no decreciente Q, los pesos del operador OWA se pueden calcular a partir de la siguiente expresión [Yag88, Yag93]:

$$\omega_m = Q\left(\frac{m}{s_j}\right) - Q\left(\frac{m-1}{s_j}\right), \quad m = 1, \ldots, s_j$$

siendo s_j el número de valores a agregar.

El operador OWA, junto con esta forma de cálculo de los pesos asociados, nos permite determinar la proporción de reglas que queremos que intervenga en el proceso de inferencia. Para ello, sólo tendremos que especificar los parámetros a y b que definen el cuantificador difuso y calcular el vector de pesos para el proceso de agregación bajo el cual subyace ese cuantificador. De esta forma, estamos definiendo un MRD en el que no intervienen todas las reglas sino "la mayoría" y este concepto de mayoría difusa está representado por un cuantificador difuso. El MRD resultante muestra similitudes con los esquemas de clasificación utilizados por expertos humanos que, en muchos casos, no clasifican en función de una sóla regla (MRD clásico), ni de acuerdo a lo que indican todas, sino que deciden en función de lo que establecen la mayoría de las reglas.

En problemas de clasificación parece natural establecer para el parámetro a el valor 0, ya que un valor de $a > 0$ implica que las reglas con mayor grado de asociación no se utilicen en el proceso de inferencia.

Métodos de Razonamiento Difuso basados en el operador QuasiOWA. Basándonos en el concepto de agregación proporcionado por la familia de operadores OWA, podemos definir otro MRD en función del operador QuasiOWA definido de la siguiente forma [FMR95]:

$$f_8(a_1,\ldots,a_{s_j}) = H^{-1}\left[\sum_{m=1}^{s_j} \omega_m \cdot H(b_m)\right]$$

siendo H una función contínua y estrictamente monótona.

En nuestros experimentos hemos utilizado la función $H(x) = x^p$ con $p \in I\!R$.

Este operador realiza el mismo tipo de selección de información que el operador OWA y además, por la inclusión de la función $H(x) = x^p$, proporciona un efecto compensativo [DP84, DP85] de los valores seleccionados para la agregación.

2.6.2 Experimentación y Análisis de Resultados

Para analizar el comportamiento de los MRDs propuestos, hemos calculado los porcentajes de clasificación correcta sobre ejemplos de prueba para las mismas bases de ejemplos -Iris, Pima y Wine- con las mismas BRs que en la sección anterior. En las tablas 2.8, 2.9 y 2.10 mostramos los resultados obtenidos con los MRDs propuestos que seleccionan reglas, comparándolos con los alcanzados con el MRD clásico y los del MRD que considera la información de todas las reglas con mejor comportamiento. De nuevo, en negrita se marca el mejor resultado para cada problema y tipo de BR.

f	p	Pr. g_1	Pr. g_2	f	p	Pr. g_1	Pr. g_2	f	p	Pr. g_1	Pr. g_2
f_0	—	88.25	—	f_0	—	94.32	—	f_0	—	94.32	—
f_1	—	92.88	**94.38**	f_3	10	94.32	94.32	f_3	10	94.32	94.32
f_8	0,0.3,20	89.85	91.29	f_7	0,0.3	**95.23**	**95.23**	f_8	0,0.8,20	**94.81**	94.38
f_8	0,0.8,50	89.85	90.79	f_7	0,0.5	94.81	94.81	f_8	0,0.8,50	94.38	94.38
f_7	0,0.8	88.41	90.76	f_8	0,0.3,20	94.81	94.81	f_7	0,0.3	92.36	94.32
BR tipo (a) (40.4 reglas)				BR tipo (b) (40.4 reglas)				BR tipo (c) (37.4 reglas)			

Tabla 2.8. Porcentajes de clasificación sobre datos de prueba de la base de ejemplos de Iris para BRs obtenidas con el Método 1

Los resultados obtenidos muestran que, para el conjunto de ejemplos Iris, los MRDs basados en los operadores OWA (f_7) y QuasiOWA (f_8) incrementan el poder de clasifi-

f	p	Pr. g_1	Pr. g_2	f	p	Pr. g_1	Pr. g_2	f	p	Pr. g_1	Pr. g_2
f_0	—	64.88	—	f_0	—	73.23	—	f_0	—	74.16	—
f_1	—	**72.11**	70.97	f_6	10	73.53	73.43	f_6	20	**74.68**	74.27
f_8	0,0.3,50	60.19	60.49	f_8	0,0.5,20	**73.95**	73.54	f_8	0,0.8,20	74.26	74.16
f_7	0,0.3	60.36	60.06	f_8	0,0.5,50	73.64	73.75	f_8	0,0.3,20	74.16	74.16
f_8	0,0.3,20	59.77	60.29	f_8	0,0.5,10	73.12	**73.95**	f_8	0,0.3,50	74.16	74.05
BR tipo (a) (420.2 reglas)				BR tipo (b) (420.4 reglas)				BR tipo (c) (382 reglas)			

Tabla 2.9. Porcentajes de clasificación sobre datos de prueba de la base de ejemplos de Pima para BRs obtenidas con el Método 1

f	p	Pr. g_1	Pr. g_2	f	p	Pr. g_1	Pr. g_2	f	p	Pr. g_1	Pr. g_2
f_0	—	88.36	—	f_0	—	91.94	—	f_0	—	91.94	—
f_1	—	**92.81**	91.29	f_5	0.9	92.29	91.94	f_5	0.5	**92.97**	91.97
f_8	0,0.8,50	83.75	85.07	f_8	0,0.8,20	**92.37**	92.35	f_8	0,0.3,20	92.29	91.94
f_8	0,0.8,20	83.05	84.05	f_8	0,0.8,50	92.35	92.35	f_2	0,0.8,20	92.29	91.94
f_8	0,0.3,20	81.22	82.25	f_8	0,0.3,20	91.94	91.92	f_7	0,0.8	91.70	91.94
BR tipo (a) (102 reglas)				BR tipo (b) (102 reglas)				BR tipo (c) (101.4 reglas)			

Tabla 2.10. Porcentajes de clasificación sobre datos de prueba de la base de ejemplos de Wine para BRs obtenidas con el Método 1

cación de los SCBRDs en los que se incluyen por encima de cualquier otro MRD (95 % de acierto en datos de prueba para BRs tipo (b)).

En el caso de los conjuntos de ejemplos Pima y Wine esta mejora es más sensible en las reglas tipo (b). Para las BRs tipo (a), la cantidad de información a agregar es menor (al no aportar las reglas un grado de certeza de la clasificación en las distintas clases). En las BRs tipo (c), la aportación del grado de certeza de la clasificación en todas las clases para cada regla introduce ruido en el proceso de inferencia que no se trata de forma óptima con los MRDs basados en los operadores OWA. No obstante, y salvo para BRs tipo (a), se supera en todos los casos el porcentaje de clasificación correcta sobre ejemplos de prueba obtenido por el MRD clásico.

2.7 Experimentación con otros Procesos de Aprendizaje Inductivo

Hemos observado el efecto de la inclusión de MRDs alternativos en SCBRDs obtenidos con la extensión del algoritmo de Wang y Mendel [WM92] a problemas de clasificación [CWY95, CYP96] (Método 1). Queremos analizar el comportamiento de los distintos MRDs con BRs generadas mediante distintos procesos de aprendizaje inductivo. Para ello, hemos generado BRs de tipo (a), (b) y (c) con los siguientes procesos de aprendizaje (en el Capítulo 3 se puede encontrar una descripción de los dos procesos evolutivos, el resto se describen en el Capítulo 1):

Método 2: Método de Hong y Lee [HL96].

Método 3: Método iterativo de Ishibuchi y otros [INT92]

Método 4: Método genético de Ishibuchi y otros [INY93].

Método 5: Método genético de Yuan y Zhuang [YZ96].

Método 6: Proceso de generación de Pal y otros [PM92].

La experimentación se ha realizado con las bases de ejemplos Iris y Pima. En las tablas 2.11 y 2.12 se muestra un resumen de los resultados obtenidos por los tres MRDs alternativos con mejor comportamiento (de cualquiera de los dos tipos de modelos de inferencia) comparándolos con los alcanzados con el MRD clásico.

Los resultados obtenidos con los MRDs alternativos, en dos problemas de clasificación distintos, con BRs de tres tipos y generadas mediante seis procesos de aprendizaje inductivo con distintas heurísticas y sesgo, muestran que *los MRDs alternativos incrementan el poder de generalización del SCBRD que los incluye en la mayoría de los casos. En los casos en los que no se supera, el porcentaje de clasificación obtenido es igual al conseguido con el MRD clásico.*

Esto muestra que las propuestas realizadas permiten aprovechar en mayor grado la potencia del Razonamiento Aproximado en SCBRDs y constituyen una buena alternativa para incrementar el rendimiento de un SCBRD obtenido a través de cualquier proceso de elicitación del conocimiento.

Hay que observar que los MRDs presentan distintos comportamientos ante distintos procesos de aprendizaje inductivo, tipos de reglas y problemas, por lo que es necesario un estudio previo que nos permita determinar el tipo de MRD más adecuado en cada caso. No obstante, el estudio experimental realizado nos permite determinar algunas líneas generales de comportamiento.

BR tipo (a)				BR tipo (b)				BR tipo (c)			
f	p	Pr. g_1	Pr. g_2	f	p	Pr. g_1	Pr. g_2	f	p	Pr. g_1	Pr. g_2
MÉTODO 2											
Nro. reglas: 3				Nro. reglas: 3				Nro. reglas: 3			
f_0	—	95.27	—	f_0	—	94.78		f_0	—	94.78	—
f_3	2	95.27	95.27	f_5	0.1	94.78	94.78	f_7	0.3,0.8	94.69	**95.27**
f_5	0.1	95.27	95.27	f_3	2	94.78	94.78	f_1	—	94.78	94.78
f_7	0,0.3	95.27	95.27	f_7	0,0.3	94.78	94.78	f_2	—	94.78	94.78
MÉTODO 3											
Nro. reglas: 109.8				Nro. reglas: 109.8				Nro. reglas: 109.8			
f_0	—	94.44	—	f_0	—	94.32	—	f_0	—	94.32	—
f_5	0.1	94.44	**95.36**	f_8	0.3,0.8,20	**95.76**	**95.76**	f_8	0,0.3,20	**94.81**	**94.81**
f_3	10	93.89	94.87	f_2	—	94.32	94.32	f_7	0,0.3	94.41	**94.81**
f_6	10	94.38	94.87	f_5	0.1	94.32	94.32	f_3	10	94.32	94.32
MÉTODO 4											
Nro. reglas: 11.6				Nro. reglas: 11.6				Nro. reglas: 10.6			
f_0	—	86.43	—	f_0	—	88.54	—	f_0	—	82.54	—
f_1	—	87.06	**87.48**	f_3	10	88.54	88.54	f_7	0,0.5	**83.45**	82.96
f_3	2	**87.48**	86.96	f_5	0.9	88.54	88.54	f_8	0,0.3,20	82.96	82.96
f_5	0.3	86.96	86.96	f_6	2	88.54	88.54	f_5	0.1	82.96	82.54
MÉTODO 5											
Nro. reglas: 23.2				Nro. reglas: 23.2				Nro. reglas: 20			
f_0	—	91.28	—	f_0	—	90.26	—	f_0	—	90.75	—
f_2	—	91.28	91.28	f_2	—	90.26	90.26	f_3	5	90.75	90.75
f_5	0.1	91.28	91.28	f_3	5	90.26	90.26	f_5	0.5	90.75	90.75
f_6	2	91.28	91.28	f_7	0,0.8	90.26	90.26	f_6	2	90.75	90.75
MÉTODO 6											
Nro. reglas: 43.6				Nro. reglas: 43.6				Nro. reglas: 36.4			
f_0	—	82.93	—	f_0	—	81.17	—	f_0	—	83.32	—
f_1	—	80.70	77.40	f_1	—	**81.62**	81.17	f_3	10	**83.91**	83.32
f_3	5	78.94	75.50	f_3	20	81.17	81.09	f_6	10	83.32	**83.91**
f_3	10	78.41	77.50	f_8	0,0.8,20	76.82	81.27	f_6	5	83.38	83.32

Tabla 2.11. Porcentajes de clasificación sobre datos de prueba de la base de ejemplos de Iris para BRs obtenidas con otros procesos inductivos

BR tipo (a)				BR tipo (b)				BR tipo (c)			
f	p	Pr. g_1	Pr. g_2	f	p	Pr. g_1	Pr. g_2	f	p	Pr. g_1	Pr. g_2
MÉTODO 2											
Nro. reglas: 2				Nro. reglas: 2				Nro. reglas: 2			
f_0	—	65.61	—	f_0	—	65.11	—	f_0	—	65.00	—
f_1	—	65.61	65.61	f_1	—	65.11	**65.23**	f_7	0,0.3	**65.41**	**65.41**
f_2	—	65.61	65.61	f_2	—	65.11	**65.23**	f_7	0,0.3	65.20	65.20
f_3	0.3	65.61	65.61	f_7	—	65.11	**65.23**	f_8	0,0.3,20	65.20	65.20
MÉTODO 3											
Nro. reglas: 573.8				Nro. reglas: 573.8				Nro. reglas: 573.8			
f_0	—	73.56	—	f_0	—	73.73	—	f_0	—	67.48	—
f_7	0,0.8	72.71	**74.26**	f_4	0.3	**74.77**	74.35	f_1	—	**68.00**	**68.00**
f_6	10	74.18	73.98	f_6	2	74.76	74.14	f_3	2	**68.00**	**68.00**
f_5	0.1	74.05	74.17	f_8	0,0.5,20	74.14	74.64	f_3	20	67.89	67.69
MÉTODO 4											
Nro. reglas: 198				Nro. reglas: 198				Nro. reglas: 168.8			
f_0	—	63.76	—	f_0	—	66.95	—	f_0	—	65.72	—
f_1	—	**67.47**	65.82	f_1	—	**69.74**	68.40	f_1	—	**66.44**	**66.44**
f_3	2	65.82	65.52	f_3	2	68.40	67.88	f_3	2	**66.44**	**66.44**
f_3	5	65.22	64.91	f_3	5	67.88	67.57	f_3	5	**66.44**	66.34
MÉTODO 5											
Nro. reglas: 434.2				Nro. reglas: 434.2				Nro. reglas: 454.2			
f_0	—	67.67	—	f_0	—	73.00	—	f_0	—	74.36	—
f_1	—	**71.36**	70.64	f_8	0,0.8,20	73.31	**73.42**	f_6	5	**74.77**	74.67
f_3	2	71.25	69.82	f_6	10	73.31	73.21	f_6	20	74.77	74.46
f_3	5	69.30	68.59	f_8	0,0.3,20	73.01	73.22	f_3	20	74.46	74.57
MÉTODO 6											
Nro. reglas: 398.2				Nro. reglas: 398.2				Nro. reglas: 382			
f_0	—	61.51	—	f_0	—	60.87	—	f_0	—	60.58	—
f_3	2	64.08	63.94	f_1	—	62.84	**63.88**	f_4	0.5	65.93	**66.24**
f_3	50	63.36	**64.08**	f_3	2	**63.88**	62.96	f_3	0.3	66.12	65.73
f_1	—	63.04	63.90	f_3	5	62.75	61.61	f_7	0,0.3	64.62	65.14

Tabla 2.12. Porcentajes de clasificación sobre datos de prueba de la base de ejemplos de Pima para BRs obtenidas con otros procesos inductivos

1. Respecto al MRD:

   - En SCBRDs con BRs tipo (a) el MRD basado en la suma (f_1) proporciona buenos resultados.
   - Los MRDs basados en las funciones de agregación media quasiaritmética (f_3), SOWA Or-Like (f_5) y Badd (f_6) incrementan el rendimiento del SCBRD en los tres tipos de BRs.
   - En SCBRDs con BRs con cardinalidad alta es adecuada la inclusión de un MRD basado en la selección de reglas (f_7 ó f_8).
   - También tienen buen comportamiento estos MRDs en SCBRDs en los que la BR no contiene demasiadas reglas, pero éstas representan zonas del espacio de patrones bastante solapadas.

2. Respecto a los parámetros del MRD, su valor depende también del número de reglas y del grado de solapamiento de las regiones difusas que éstas representan. Así, en los operadores que incluyen en su definición una ponderación-penalización de valores a agregar (como f_3, f_6 o f_8), el valor del paramétro determinante de la misma (p) se incrementará en SCBRDs con gran número de reglas o con BRs con reglas muy solapadas. Los parámetros que han mostrado mejor comportamiento son:

   - Para el MRD basado en la media quasiaritmética (f_3), $p \in \{5, 10, 20, 50\}$.
   - Para el MRD basado en el operador SOWA Or-Like (f_5), $\alpha \in \{0.1, 0.3\}$.
   - Para el MRD basado en el operador Badd (f_6), $p \in \{10, 20, 50\}$.
   - Para el MRD basado en el operador OWA, $a = 0, \quad b \in \{0.3, 0.5, 0.8\}$
   - Para el MRD basado en el operador QuasiOWA, $a = 0, \quad b \in \{0.3, 0.5, 0.8\}$, $p \in \{20, 50\}$

   Hay que destacar que los valores de estos parámetros están directamente relacionados con el tipo de función de ponderación y parámetros de la misma.

Es difícil determinar el mejor valor de los parámetros de la agregación y ponderación de forma general, por lo que un proceso de aprendizaje de los mismos puede permitir la adaptación completa del MRD al problema y proceso de aprendizaje inductivo con el que se trabaje. Esta posibilidad se estudia en la sección siguiente.

2.8 Aprendizaje Evolutivo de los Parámetros del Método de Razonamiento Difuso

Los resultados obtenidos para las distintas bases de ejemplos, tipos de reglas y métodos

de aprendizaje muestran que no existen valores fijos para los distintos parámetros implicados en un MRD -los parámetros de los operadores de agregación y los de la función de ponderación- con un mejor comportamiento en todos los casos considerados. El aprendizaje de los parámetros mediante un proceso de búsqueda puede adaptar el MRD al problema y proceso de diseño específico y aportar mejoras en los resultados.

En esta sección presentamos dos procesos evolutivos para el aprendizaje de parámetros que permiten la obtención de un MRD adaptado al proceso de aprendizaje y problema de clasificación a resolver:

- un proceso de optimización de los parámetros de la función de ponderación para un MRD con los parámetros de la función de agregación prefijados, y
- un proceso de búsqueda de los mejores valores tanto para los parámetros del operador de agregación como para la ponderación.

Describiremos ambos en las siguientes subsecciones.

2.8.1 Proceso de Aprendizaje de los Parámetros de la Función de Ponderación

En la propuesta desarrollada en este capítulo, la función de ponderación tiene la siguiente expresión:

$$g_2(x) = \begin{cases} x^2, & \text{si } x < 0.5 \\ \sqrt{x}, & \text{si } x \geq 0.5 \end{cases}$$

es decir, potencia los grados de asociación superiores a 0.5 y penaliza los inferiores a este valor (figura 2.6).

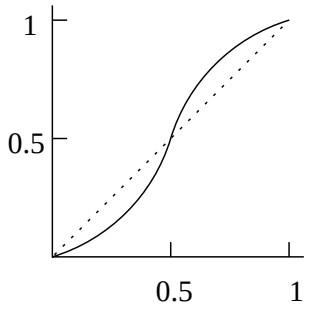

Figura 2.6. Efecto de la función de ponderación g_2

Como se ha observado en la experimentación realizada, no existen valores fijos válidos para cualquier problema ni para el punto límite entre ponderación y penalización, ni para la potencia aplicada en cualquiera de los dos casos. Dependen tanto de la dimensión

del espacio de patrones como de las características del proceso de aprendizaje inductivo utilizado. La adaptación de este tipo de función de ponderación a un problema dado implica la determinación de valores óptimos para los parámetros a, b y c de la siguiente expresión generalizada de esta función de ponderación:

$$g(x) = \begin{cases} x^a, & \text{si } x < b \\ x^c, & \text{si } x \geq b \end{cases}$$

siendo a un valor real positivo y b y c parámetros reales con dominio $b, c \in [0, 1]$.

Utilizaremos un AG para buscar los valores de estos tres parámetros que maximicen el porcentaje de clasificación correcta sobre ejemplos de entrenamiento para un SCBRD específico.

Los elementos principales del proceso genético desarrollado son los siguientes:

1. *Representación y evaluación de soluciones.* Cualquier solución al problema planteado estará formada por tres parámetros de tipo real $a \in I\!R^+$, $b \in [0, 1]$ y $c \in [0, 1]$ respectivamente, por lo que utilizaremos un AG con codificación real [HLV98b], en el que cada cromosoma está compuesto por tres genes.

2. La *función de adaptación* devolverá, para cada cromosoma, el porcentaje de clasificación correcta sobre ejemplos de entrenamiento con un MRD que incluye la ponderación-penalización que representa el individuo.

3. *Generación de la población inicial.* Para introducir en el AG la información disponible sobre el problema, la población inicial estará formada por individuos que representen la ponderación prefijada que hemos usado en las secciones anteriores (no ponderar y ponderar según el criterio del cuadrado y la raiz cuadrada con punto de corte 0.5). El resto se generará aleatoriamente.

4. Utilizaremos como mecanismo de selección el *muestreo estocástico universal* propuesto por Baker [Bak87] junto con el esquema de asignación de probabilidades por ordenación lineal. Este mecanismo de selección se completa con el *modelo de selección elitista* que asegura que el cromosoma mejor adaptado sobrevivirá de una generación a la siguiente [Gol89, Mic96].

5. Como operadores de variación, el AG incluye el *operador de mutación no uniforme* descrito en el Capítulo 1 y el *operador de cruce dinámico de Dubois* cuyo funcionamiento se muestra en la siguiente subsección.

2.8.1.1 El Operador de Cruce Dinámico de Dubois

En este proceso de optimización hemos utilizado el *operador de cruce dinámico de Dubois*, un operador de cruce para AGs con codificación real, basado en conectivas difusas. Este operador ha sido definido por Herrera y otros en [HLV96] como una propuesta

al problema de convergencia prematura presente en los AGs. Recordemos que la *convergencia prematura* es un estancamiento en la búsqueda debido a la falta de diversidad en la población y a una relación desproporcionada entre explotación y exploración. En el problema práctico planteado -la búsqueda de valores para tres parámetros que optimicen una función- los individuos estarán formados por tres genes, por lo que es necesario introducir operadores de este tipo que eviten la falta de diversidad y la convergencia hacia un óptimo local.

El funcionamiento del operador de cruce dinámico de Dubois se basa en la definición de unas funciones de combinación de genes cuya actuación varía durante la ejecución del AG, proporcionando la relación deseada entre explotación y exploración. Para ver la actuación del operador de cruce, debemos definir previamente este conjunto de funciones.

Funciones de combinación de genes. Consideremos dos genes que van a ser combinados, $c_i^1, c_i^2 \in [a_i, b_i]$. Si $\alpha_i = \min\{c_i^1, c_i^2\}$ y $\beta_i = \max\{c_i^1, c_i^2\}$, el intervalo de acción de los dos genes se puede dividir en tres intervalos $[a_i, \alpha_i]$, $[\alpha_i, \beta_i]$ y $[\beta_i, b_i]$. Estos intervalos determinan tres regiones a las que pueden pertenecer los genes resultantes de alguna combinación de los originales (c_i^1 y c_i^2). También podría ser razonable considerar una región $[\alpha_i', \beta_i']$ como se observa en la figura 2.7.

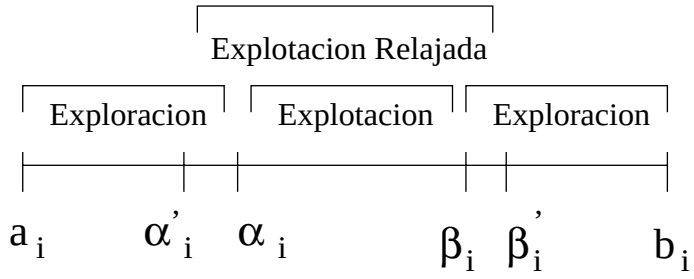

Figura 2.7. Intervalos de acción para un gen

Estos intervalos se pueden considerar zonas de exploración o explotación: El intervalo cuyos extremos son los genes es una zona de explotación, los dos intervalos que tiene a ambos lados, zonas de exploración, y la región con extremos α_i' y β_i', zona de explotación relajada.

De acuerdo a estos intervalos, se pueden proponer cuatro funciones F, S, M y L definidas de $[a_i, b_i] \times [a_i, b_i]$ en $[a_i, b_i]$ que verifiquen las propiedades siguientes,

P1. $\forall c_i^1, c_i^2 \in [a_i, b_i]$ $F(c_i^1, c_i^2) \leq \min\{c_i^1, c_i^2\}$,

P2. $\forall c_i^1, c_i^2 \in [a_i, b_i]$ $S(c_i^1, c_i^2) \geq \max\{c_i^1, c_i^2\}$,

P3. $\forall c_i^1, c_i^2 \in [a_i, b_i]$ $\min\{c_i^1, c_i^2\} < M(c_i^1, c_i^2) < \max\{c_i^1, c_i^2\}$,

P4. $\forall c_i^1, c_i^2 \in [a_i, b_i]$ $F(c_i^1, c_i^2) \leq L(c_i^1, c_i^2) \leq S(c_i^1, c_i^2)$,

P5. F, S, M y L son funciones monótonas y no decrecientes.

Cada una de estas funciones combina dos genes y proporciona un resultado perteneciente a uno de los intervalos mencionados anteriormente, por lo que un operador de cruce basado ellas (denominado operador de cruce tipo F, S, M o L, respectivamente) tiene asociado un carácter de exploración o explotación en función del rango que cubra.

Podemos construir estas funciones utilizando t-normas (T), t-conormas (G), operadores promedio (P) y operadores de compensación generalizados (Q) [Miz89a, Miz89b] de la siguiente forma: Si $c_i^1, c_i^2 \in [a_i, b_i]$, entonces

$$\begin{aligned} F(c_i^1, c_i^2) &= a_i + (b_i - a_i) \cdot T(s_i^1, s_i^2), \\ S(c_i^1, c_i^2) &= a_i + (b_i - a_i) \cdot G(s_i^1, s_i^2), \\ M(c_i^1, c_i^2) &= a_i + (b_i - a_i) \cdot P(s_i^1, s_i^2), \\ L(c_i^1, c_i^2) &= a_i + (b_i - a_i) \cdot Q(s_i^1, s_i^2), \end{aligned}$$

donde $s_i^1 = \frac{c_i^1 - a_i}{b_i - a_i}$ y $s_i^2 = \frac{c_i^2 - a_i}{b_i - a_i}$

Estos operadores verifican las cinco propiedades indicadas anteriormente y nos permiten definir operadores de cruce F, S, M o L, en función del comportamiento que queramos en el AG.

Un mecanismo para evitar la convergencia prematura es inducir, a través de los operadores genéticos, equilibrios entre exploración y explotación en función del tiempo (que en un AG se mide mediante el número de generaciones) bajo el principio básico de proteger la exploración en las etapas iniciales y la explotación despues. Con la exploración la diversidad es mayor y se incrementa la probabilidad de encontrar zonas cercanas a soluciones óptimas. En el momento en el que se supone que la población tiene información sobre estas zonas, la explotación produce la convergencia al óptimo. Para mantener esta secuencia exploración-explotación se han propuesto distintos mecanismos, entre los que podemos mencionar la disminución de la probabilidad de mutación durante la ejecución del AG [Fog89], el escalado de la función de adaptación o el uso de operadores como la mutación no uniforme (utilizado en esta propuesta y descrito en el Capítulo 1) o un operador de cruce con comportamiento dinámico con propiedades de exploración o explotación en función de la etapa del AG en que se aplique. Esto último se puede conseguir a través de un operador de cruce dinámico basado en la definición de las funciones F,S, M y L mediante t-normas, t-conormas y operadores de media parametrizados [Miz89a, Miz89b]. Para ello, en [HLV96] se proponen tres familias de funciones parametrizadas cuyos parámetros varian durante la ejecución del AG, variando de esta forma la relación entre exploración y explotación. Para un AG con número máximo de generaciones g_{max}, se pueden definir tres familias de funciones $F = (F^1, \ldots, F^{g_{max}})$, $S = (S^1, \ldots, S^{g_{max}})$ y $M = (M^1, \ldots, M^{g_{max}})$, que para una generación t, $1 \leq t \leq g_{max} - 1$, verifican lo siguiente:

$$\begin{array}{ll} \forall c_i^1, c_i^2 \in [a,b] & F^t(c_i^1, c_i^2) \leq F^{t+1}(c_i^1, c_i^2) \text{ y } F^{g_{max}} = \min\{c_i^1, c_i^2\} \\ \forall c_i^1, c_i^2 \in [a,b] & S^t(c_i^1, c_i^2) \geq S^{t+1}(c_i^1, c_i^2) \text{ y } S^{g_{max}} = \max\{c_i^1, c_i^2\} \\ \forall c_i^1, c_i^2 \in [a,b] & M^t(c_i^1, c_i^2) \geq M^{t+1}(c_i^1, c_i^2) \text{ o } M^t(c_i^1, c_i^2) \leq M^{t+1}(c_i^1, c_i^2) \forall t, \\ & \text{y } M^{g_{max}}(c_i^1, c_i^2) = M_{lim}(c_i^1, c`) \end{array}$$

donde M_{lim} es una función M denominada función límite M.

En realidad, el último punto denota tres familias de funciones M: M^+, M^- y M según verifiquen la primera, segunda o tercera parte de la expresión completa.

Las familias de funciones F y S se pueden construir con una t-norma parametrizada T^q que converja al mínimo y una t-conorma parametrizada G^q que converja al máximo. Para poder aplicarlas, necesitamos transformar los valores de los genes al intervalo [0,1] y posteriormente volverlos a llevar a puntos de su dominio, además de una función de transformación $\delta(\cdot)$ del valor de la generación en la que se encuentra el AG ($\{1, \ldots, g_{max}\}$), en el rango del parametro q que depende del operador parametrizado que utilicemos. Con esto, las familias de funciones F y S tienen la siguiente expresión:

$$\begin{array}{llll} \forall c_i^1, c_i^2 \in [a,b], & 1 \leq t \leq g_{max}, & F^t(c_i^1, c_i^2) = a + (b-a) \cdot T^{\delta_F(t)}(s_i^1, s_i^2) \\ \forall c_i^1, c_i^2 \in [a,b], & 1 \leq t \leq g_{max}, & S^t(c_i^1, c_i^2) = a + (b-1) \cdot G^{\delta_S(t)}(s_i^1, s_i^2) \end{array}$$

donde $s_i^1 = \frac{c_i^1 - a}{b-a}$, $s_i^2 = \frac{c_i^2 - a}{b-a}$, y $\delta_F(\cdot)$, $\delta_S(\cdot)$ son dos funciones de transformación.

Para el operador dinámico de Dubois, se utilizan la t-norma y t-conorma parametrizadas de Dubois:

$$T^q(x,y) = \frac{xy}{\max\{x, y, q\}}$$

y

$$G^q(x,y) = 1 - \frac{(1-x)(1-y)}{\max\{(1-x), (1-y), q\}}$$

con $0 \leq q \leq 1$.

Para esta t-norma y t-conorma, se utiliza la función de transformación

$$\delta_F(t) = \delta_S(t) = \frac{1}{\sqrt{t}}$$

Para obtener familias de funciones M, se utilizan operadores promedio parametrizados. Un ejemplo de estas funciones es

$$\forall x, y \in [0,1], \quad P^q(x,y) = \left(\frac{x^q + y^q}{2}\right)^{1/q} \qquad -\infty \leq q \leq +\infty$$

Utilizando esta función se puede obtener una familia $M = (M^1, \ldots, M^{g_{max}})$

$$\forall c_i^1, c_i^2 \in [a,b], \quad 1 \leq t \leq g_{max} \quad M^t(c_i^1, c_i^2) = a + (b-a) \cdot P^{\delta_M(t)}(s_i^1, s_i^2)$$

Para el operador de cruce dinámico de Dubois, se han utilizado las siguientes funciones de transformación para obtener las familias de funciones M^+ y M^- [HLV96]:

$$\delta_{M^+}(t) = 1 + \ln\left(\frac{g_{max}}{t}\right)$$

$$\delta_{M^-}(t) = 1 + \ln\left(\frac{t}{g_{max}}\right)$$

Cada una de estas familias de funciones nos permite combinar genes con resultados pertenecientes a cualquiera de los intervalos mencionados (figura 2.7).

Operador de cruce dinámico de Dubois. Una vez conocida la actuación de estas familias de funciones, podemos definir el *cruce dinámico de Dubois*. Sea $O \in \{F, S, M, L\}$ y sean $C_1^t = (c_1^{1t}, \ldots, c_n^{1t})$, y $C_2^t = (c_1^{2t}, \ldots, c_n^{2t})$, los dos cromosomas seleccionados para el cruce. Podemos generar el cromosoma $H^t = (h_1^t, \ldots, h_n^t)$ como:

$$h_i^t = O(c_i^{1t}, c_i^{2t}), \quad i = 1, \ldots, n$$

Este operador aplica la misma función tipo F, tipo S, tipo M^+ o tipo-M^- para todos los genes dentro de los cromosomas a cruzar. Por esta razón, se denomina F-cruce, S-cruce o M-cruce, dependiendo de la función aplicada.

En nuestros experimentos hemos aplicado el operador de cruce de la siguiente forma: dados dos cromosomas padres seleccionados para el cruce, se obtendrán cuatro descendientes como resultado de aplicar las funciones F, S, M^+ y M^- con los parámetros correspondientes. De los cuatro descendientes generados, se incorporarán los dos mejor adaptados. En [HLV96] se analizan otros criterios de selección.

Este operador constituye un mecanismo para encontrar un balance adecuado entre exploración y explotación bajo el principio básico: *proteger la exploración en las etapas iniciales del algoritmo y la explotación después.*

2.8.2 Proceso de Aprendizaje de todos los Parámetros de un Método de Razonamiento Difuso

Los valores de los parámetros de la función de ponderación están directamente relacionados con los de la función de agregación ya que, como se ha mencionado, algunos operadores de agregación realizan una ponderación-penalización en el proceso de agregación. Se puede conseguir una adaptación mayor del MRD al problema y al proceso de

Algoritmo Genetico con Busqueda Local
inicio
$t \longleftarrow 0$
Inicializar $(P(t))$
Evaluar $(P(t), f)$
Mientras $Parada(P(t), \Theta_p) \neq Verdadero$ **Hacer**
$t \longleftarrow t + 1$
Seleccionar $(P(t-1), f, \Theta_s, P(t))$
Recombinar$(P(t), \Theta_r, P'(t))$
Mutar $(P'(t), \Theta_m, P''(t))$
Estrategia_Evolucion $(P''(t), k, P'''(t))$
$P(t) \longleftarrow P'''(t)$
Evaluar $(P(t), f)$
fin_mientras
Devolver el mejor individuo C
fin

Figura 2.8. Estructura de un Algoritmo Genético con búsqueda local realizada mediante una Estrategia de Evolución

inducción mediante un proceso de aprendizaje no sólo de los parámetros de la función de ponderación sino también de los valores que determinan la función de agregación.

El proceso de adaptación propuesto es un método evolutivo que utiliza como algoritmo base un AG con codificación real en el que se incluye una EE aplicada a un porcentaje de los individuos en cada generación. El AG determina valores adecuados para los parámetros de la función de ponderación y la EE realiza una búsqueda local en esos individuos, para encontrar los mejores valores de los parámetros de la función de agregación. De esta forma, se limita de forma considerable el espacio de búsqueda. El proceso completo realiza una búsqueda local dentro del AG. Es lo que en la bibliografía especializada se ha denominado *genetic local search* [YI96].

El método propuesto utiliza la EE como una técnica que explota intensivamente el conocimiento aportado por un individuo perteneciente al AG que lo engloba. Concretamente, utilizaremos una EE (1+1) que, dados unos valores para la función de ponderación, modifica los valores del parámetro o parámetros de la función de agregación para mejorar la adaptación global del individuo. Esta EE se integra en el AG actuando individualmente sobre un porcentaje de los mejores cromosomas, despues de la recombinación de la población mediante cruce y mutación. El funcionamiento global de este AG con búsqueda local, se representa en la figura 2.8, en la que k representa el número de mejores individuos a los que se va a aplicar la EE.

Los elementos característicos del AG con búsqueda local son los descritos para el AG que optimiza los valores de la función de ponderación:

1. Codificación real.
2. Función de adaptación igual al porcentaje de clasificación correcta sobre los ejemplos de entrenamiento.
3. Población inicial formada por individuos que representen los valores de los parámetros utilizados en la sección 2.4 y el resto generados aleatoriamente.
4. Operador de cruce dinámico de Dubois y operador de mutación no uniforme.

Los elementos específicos de la EE se definen en el siguiente apartado.

2.8.2.1 Elementos de la Estrategia de Evolución

La EE(1+1) hace evolucionar un individuo aplicándole mutaciones a todos sus genes. En el proceso evolutivo en el que se integra, la EE se aplica sobre los genes que codifican los parámetros del operador de agregación para la función de ponderación representada por el resto del cromosoma.

La mutación aplicada sigue una distribución normal, cuya desviación típica se modifica conforme lo haga el individuo padre. El individuo hijo sustituirá al padre cuando presente mejor adaptación que él. Este proceso se repite hasta que se verifique una condición de parada, habitualmente que el proceso se estabilice durante un número de generaciones determinado.

El operador de mutación, suma a cada componente del cromosoma un valor distribuido según una distribución normal con desviación típica σ. Así, si $C = (c_1, \ldots, c_n)$ es el individuo padre, su descendiente tendrá la siguiente expresión:

$$C' = (c_1 + z_1, \ldots, c_n + z_n)$$

donde los valores z_i siguen una distribución normal de media 0 y desviación típica σ.

Esta desviación típica se modifica con el padre según la *regla de éxito* 1/5 *de Rechemberg*:

$$\sigma' = \begin{cases} \frac{\sigma}{c^{1/n}}, & \text{si } p > \frac{1}{5} \\ \sigma \cdot c^{1/n}, & \text{si } p < \frac{1}{5} \\ \sigma, & \text{si } p = \frac{1}{5} \end{cases}$$

siendo p la frecuencia relativa de mutaciones efectuadas con éxito y c una constante que determina la cantidad en que se actualiza σ.

2.8.3 Experimentación y Análisis de Resultados

Tal y como hemos descrito los procesos de aprendizaje, la experimentación se ha dividido en dos partes:

Parte I: Aprendizaje genético de los parámetros de la función de ponderación para los MRDs basados en las funciones de agregación siguientes:

- suma normalizada (f_1),
- media aritmética (f_2),
- media quasiaritmética (f_3) con algunos de los valores del parámetro p que han proporcionado mejores resultados en la experimentación anterior ($p \in \{20, 50\}$),
- SOWA Or-Like (f_5), con $\alpha \in \{0, 5, 0.7, 0.9\}$,
- Badd (f_6), con $p \in \{20, 50\}$,
- OWA (f_7), con $a = 0, \quad b \in \{0.3, 0.5, 0.8)\}$, y
- QuasiOWA (f_8), con $a = 0, b \in \{0.3, 0.5, 0.8\}, \quad p \in \{20, 50\}$.

Parte II: Aprendizaje de todos los parámetros de los MRDs basados en

- media quasiaritmética (f_3),
- SOWA Or-Like (f_5),
- Badd (f_6),
- OWA (f_7), y
- QuasiOWA (f_8)

mediante el proceso genético con búsqueda local.

Los parámetros de los procesos evolutivos se describen en la tabla 2.13.

La experimentación se ha realizado para las bases de ejemplos Iris, Pima y Wine, con los tres tipos de reglas difusas y con el proceso de aprendizaje de Wang y Mendel (notado como Método 1). Los resultados se muestran en las tablas 2.14, 2.16 y 2.15.

	Parte I	Parte II
Número de generaciones	300	300
Longitud población	21	41
% de individuos a aplicar la EE	—	50
b (Mutación No Uniforme)	5	5
c (EE)	—	0.9
P_c	0.6	0.6
P_m	0.1	0.1

Tabla 2.13. Parámetros de los procesos evolutivos descritos como Parte I y Parte II

	f	p	En.	Pr.	p	En.	Pr.	p	En.	Pr.
	f_0	—	90.97	88.25	—	97.31	94.32	—	96.96	94.32
I	f_1	—	99.28	**94.87**	—	98.57	94.38	—	98.02	94.81
	f_2	—	91.90	90.34	—	97.65	93.73	—	98.21	94.32
	f_3	20	91.90	90.34	20	97.31	94.32	20	96.96	94.32
	f_5	0.5	92.80	90.18	0.5	97.49	93.83	0.7	97.83	94.32
	f_6	20	91.74	88.74	20	97.31	94.32	20	96.96	94.32
	f_7	0,0.3	92.00	90.18	0,0.3	97.11	**94.81**	0,0.8	97.65	94.81
	f_8	0,0.8,20	91.89	89.85	0,0.3,20	97.11	**94.81**	0,0.8,20	96.96	94.38
II	f_3	—	92.08	90.34	—	97.65	93.73	—	98.22	94.81
	f_5	—	92.62	89.12	—	97.84	94.22	—	98.21	95.23
	f_6	—	92.50	91.20	—	97.33	93.43	—	97.86	**95.78**
	f_7	—	94.03	89.65	—	97.84	**94.81**	—	97.65	94.81
	f_8	—	93.67	87.90	—	98.19	94.22	—	97.65	94.38
			BR tipo (a)			BR tipo (b)			BR tipo (c)	

Tabla 2.14. Resultados del proceso de aprendizaje de parámetros para Iris con BRs obtenidas con el Método 1

	f	p	En.	Pr.	p	En.	Pr.	p	En.	Pr.
	f_0	—	89.51	64.88	—	85.81	73.23	—	81.94	74.16
I	f_1	—	91.67	**70.47**	—	86.02	72.91	—	82.08	74.16
	f_2	—	85.15	58.23	—	86.40	72.19	—	82.01	74.16
	f_3	20	84.94	57.91	50	86.37	72.49	20	82.08	74.06
	f_5	0.5	87.45	63.34	0.7	86.51	73.12	0.5	82.08	74.37
	f_6	50	91.43	65.52	20	86.51	72.81	20	82.61	**74.47**
	f_7	0,0.3	81.60	60.38	0,0.3	85.29	72.82	0,0.3	81.31	74.16
	f_8	0,0.3,20	81.49	60.29	0,0.5,20	84.94	**73.64**	0,0.3,50	81.28	73.95
II	f_3	—	84.91	54.00	—	86.26	71.38	—	82.08	70.16
	f_5	—	90.59	65.48	—	86.54	72.91	—	82.47	74.26
	f_6	—	91.46	60.75	—	86.79	73.22	—	82.78	74.16
	f_7	—	85.12	57.71	—	86.44	72.50	—	82.26	73.85
	f_8	—	85.22	54.30	—	86.75	71.58	—	82.26	72.63
			BR tipo (a)			BR tipo (b)			BR tipo (c)	

Tabla 2.15. Resultados del proceso de aprendizaje de parámetros para Pima con BRs obtenidas con el Método 1

	f	p	En.	Pr.	p	En.	Pr.	p	En.	Pr.
	f_0	—	99.53	88.36	—	98.88	91.94	—	98.54	91.94
I	f_1	—	100.00	**90.10**	—	98.88	**92.70**	—	98.71	92.29
	f_2	—	97.64	82.05	—	98.84	92.47	—	98.39	92.29
	f_3	50	97.64	83.26	20	99.37	90.53	20	98.54	91.21
	f_5	0.5	97.80	82.73	0.5	99.37	91.96	0.5	98.87	92.35
	f_6	20	99.53	87.60	20	98.88	91.21	20	98.54	91.21
	f_7	0,0.8	97.64	83.13	0,0.8	99.20	92.02	0,0.8	98.39	92.29
	f_8	0,0.8,20	97.64	84.02	0,0.8,20	99.20	91.29	0,0.5,50	98.37	91.94
II	f_3	—	97.64	67.75	—	99.37	89.88	—	98.71	90.92
	f_5	—	98.59	82.99	—	99.37	92.29	—	98.87	92.35
	f_6	—	99.53	70.84	—	99.52	90.51	—	99.03	89.53
	f_7	—	98.10	83.72	—	99.20	92.00	—	98.85	**92.70**
	f_8	—	97.95	82.62	—	99.20	91.59	—	98.85	92.03
			BR tipo (a)			BR tipo (c)			BR tipo (c)	

Tabla 2.16. Resultados del proceso de aprendizaje de parámetros para Wine con BRs obtenidas con el Método 1

Los experimentos realizados muestran un incremento del porcentaje de clasificación correcta sobre ejemplos de prueba superior al obtenido por el MRD clásico.

Hay que señalar que, al utilizar sólo datos de entrenamiento, los procesos de aprendizaje de parámetros propuestos no llevan necesariamente a un incremento de la capacidad de generalización del sistema. Esto se debe a que, en ocasiones, se sobreajustan los parámetros a los ejemplos de entrenamiento y el SCBRD resultante clasifica peor los ejemplos de prueba.

No obstante, con los procesos evolutivos de aprendizaje propuestos se obtiene el mejor porcentaje de clasficación sobre ejemplos de prueba para SBCRDs obtenidos con el Método 1 para IRIS con BRs tipo (a) y (c) y para Wine con BR tipo (b).

2.9 Comentarios Finales

Como resumen, en las gráficas 2.9, 2.10 y 2.11 se muestran los mejores resultados obtenidos con todos los MRDs propuestos, con función de ponderación g_1 y g_2. Son los resultados alcanzados con el conjunto de ejemplos Iris tras los procesos de aprendizaje correspondientes a la Parte I y Parte II con BRs tipo (a), (b) y (c) aprendidas con el Método 1.

El estudio realizado sobre MRDs en SCBRDs nos permite concluir con las siguientes observaciones:

1. El MRD que clasifica en base a la regla ganadora desperdicia la información de otras reglas compatibles con el ejemplo. Por tanto, pierde parte de la potencia del Razonamiento Aproximado.

2. Los MRDs alternativos propuestos representan esquemas de clasificación que van desde decidir en base a la regla con máxima compatibilidad, hasta hacerlo respecto a aquella que tiene mínimo grado, pasando por la media. Son parametrizables y representan cualquier grado de compensación entre estos tres.

3. La experimentación realizada sobre distintos conjuntos de ejemplos, tipos de reglas y procesos de aprendizaje inductivo, muestra que los MRDs alternativos incrementan la capacidad de generalización del SCBRD. Estos MRDs constituyen una buena alternativa para incrementar el rendimiento de un SCBRD obtenido a través de cualquier proceso de elicitación del conocimiento.

4. Los MRDs basados en operadores que seleccionan la información a agregar en base al concepto de mayoría difusa tienen buen comportamiento en BRs con cardinali-

dad alta o BRs con reglas que representan regiones del espacio de patrones muy solapadas.

5. Todos los resultados expuestos tienen la limitación de la BR ya que ésta tiene un límite en el porcentaje de clasificación correcta por su propia composición. De hecho, hay un número de ejemplos no cubiertos por las reglas, que no podrá clasificarse bien con ninguna alternativa.

6. Los MRDs con mejor comportamiento son:

   - suma (f_1),
   - media quasiaritmética (f_3),
   - SOWA Or-Like (f_5),
   - Badd (f_6),
   - OWA (f_7), y
   - quasiOWA (f_8)

   Los parámetros de los mismos se pueden determinar en base al conocimiento teórico de los operadores en que se basan, las características del problema y la BR a la que se aplican.

7. La inclusión de una función de ponderación en el MRD suele incrementar el rendimiento del SCBRD aunque su expresión y parámetros también dependen del problema.

8. No existe un MRD con mejor comportamiento en todos los problemas y procesos de aprendizaje, por lo que será necesario realizar una pequeña experimentación que determine la mejor elección para el SCBRD específico.

9. Los procesos evolutivos de aprendizaje de parámetros de la función de ponderación y de la función de agregación permiten adaptar el MRD al problema y proceso de generación del SCBRD. Con ellos hemos obtenido los mejores resultados para uno de los conjuntos de ejemplos. No obstante, este proceso de aprendizaje tiene la limitación de la BR en cuanto que ésta tiene un límite en el porcentaje de clasificación debido a su propia composición. Esto hace que, a veces, el proceso de optimización ajuste el MRD demasiado al conjunto de entrenamiento al intentar mejorar el porcentaje de clasificación, lo que da lugar a que se produzca sobreaprendizaje.

10. Es importante destacar que el análisis de MRDs se ha realizado de forma independiente a la etapa de aprendizaje del SCBRD, por lo que los MRDs propuestos pueden integrarse en los distintos SCBRDs desarrollados en la literatura especializada.

11. A pesar de la limitación de la composición de la BR, los MRDs propuestos han incrementado los porcentajes de clasificación, lo que hace pensar que con la integración

del MRD en el proceso de aprendizaje del SCBRD, se pueden obtener mejores resultados. Este aspecto se considerará en el proceso de extracción de la BC que se propone en el siguiente capítulo.

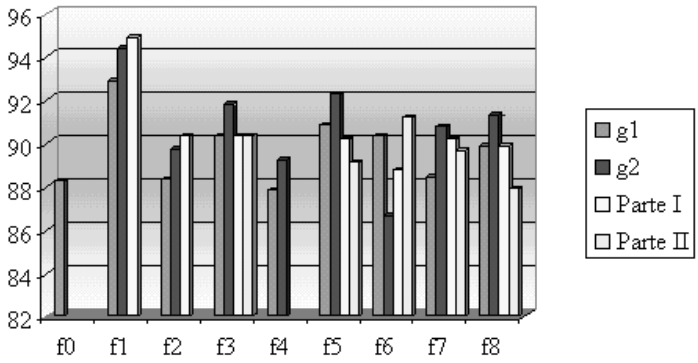

Figura 2.9. Resultados para Iris con un SCBRD con BR tipo (a) y distintos MRDs

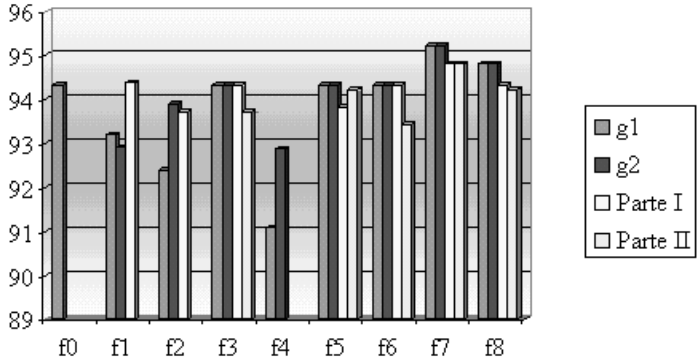

Figura 2.10. Resultados para Iris con un SCBRD con BR tipo (b) y distintos MRDs

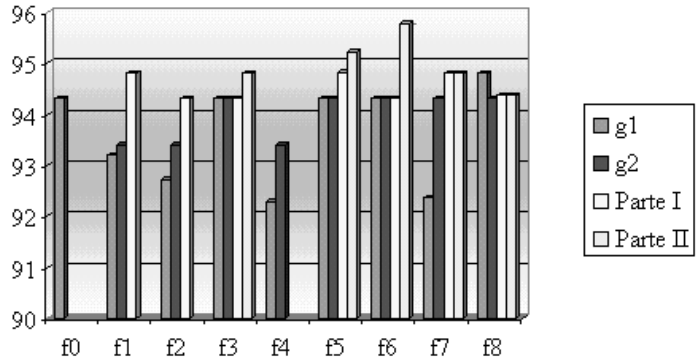

Figura 2.11. Resultados para Iris con un SCBRD con BR tipo (c) y distintos MRDs

2.10 Apéndice: Resultados con Distintos Métodos de Razonamiento Difuso

f	p	Ent.	Pr.	Ent.	Pr.	Ent.	Pr.	Ent.	Pr.
		g_1		g_2		g_1		g_2	
f_0	—	90.97	88.25	—	—	97.31	94.32	—	—
f_1	—	97.29	92.88	98.56	94.38	96.43	93.20	97.13	92.92
f_2	—	90.80	88.31	89.72	89.71	96.04	92.39	96.23	93.89
f_3	2	91.35	88.80	89.73	91.78	96.23	93.89	96.41	93.89
f_3	5	91.53	90.83	89.56	91.78	96.41	93.89	96.77	94.32
f_3	10	91.18	90.34	89.56	91.78	96.77	94.32	97.31	94.32
f_3	20	91.18	90.34	89.20	91.78	97.31	94.32	97.31	94.32
f_3	50	89.90	90.34	88.81	91.19	97.31	94.32	97.31	94.32
f_4	0.3	88.26	87.82	89.18	89.22	95.11	91.09	96.04	92.88
f_4	0.5	86.82	86.26	88.09	88.25	93.83	89.57	95.86	92.88
f_5	0.1	90.98	88.31	90.64	91.25	96.39	92.39	96.59	93.89
f_5	0.3	92.08	90.34	90.64	92.27	96.77	93.89	96.77	94.32
f_5	0.5	92.08	90.83	90.64	92.27	96.77	94.32	96.77	94.32
f_5	0.7	92.08	90.83	90.64	92.27	97.13	94.32	97.31	94.32
f_5	0.9	92.08	90.83	90.64	92.27	97.31	94.32	97.31	94.32
f_6	2	89.90	90.34	86.45	86.65	96.23	93.89	96.94	94.32
f_6	5	86.45	85.80	86.45	86.65	97.12	94.32	97.31	94.32
f_6	10	85.37	85.80	86.63	87.13	97.31	94.32	97.31	94.32
f_6	20	85.19	85.63	87.00	86.65	97.31	94.32	97.31	94.32
f_6	50	85.91	86.12	90.97	88.25	97.31	94.32	97.31	94.32
f_7	0,0.3	89.26	90.66	89.61	90.70	96.39	95.23	96.57	95.23
f_7	0,0.5	90.70	89.32	90.54	90.70	96.39	94.81	96.39	94.81
f_7	0,0.8	91.15	88.41	91.17	90.76	96.76	92.98	96.41	93.89
f_7	0.3,0.8	83.53	80.80	82.27	78.87	93.88	91.05	94.05	91.48
f_8	0,0.3,20	88.90	89.69	89.45	90.79	97.11	94.81	97.11	94.81
f_8	0,0.5,20	90.15	89.26	89.63	90.70	97.31	94.81	97.31	94.81
f_8	0,0.8,20	91.34	89.85	90.07	91.29	97.31	94.31	97.32	94.32
f_8	0.3,0.8,20	84.42	80.94	83.16	80.84	95.69	93.44	95.70	93.93
f_8	0,0.3,50	88.54	89.69	89.28	90.11	97.11	94.81	97.11	94.81
f_8	0,0.5,50	89.44	89.26	89.46	90.11	97.31	94.81	97.31	94.81
f_8	0,0.8,50	90.43	89.85	89.51	90.79	97.31	94.32	97.31	94.32
f_8	0.3,0.8,50	84.95	80.41	83.17	80.84	95.70	94.36	95.88	94.36
		BR tipo (a) (40.4 reglas)				BR tipo (b) (40.4 reglas)			

Tabla 2.17. Resultados para Iris con BRs generadas con el Método 1 (1)

f	p	Ent.	Pr.	Ent.	Pr.
		g_1		g_2	
f_0	—	96.96	94.32	—	—
f_1	—	96.25	93.20	96.95	93.40
f_2	—	96.25	92.72	96.95	93.40
f_3	2	96.95	93.50	96.77	94.32
f_3	5	96.96	94.32	96.96	94.32
f_3	10	96.96	94.32	96.96	94.32
f_3	20	96.96	94.32	96.96	94.32
f_3	50	96.96	94.32	96.96	94.32
f_4	0.3	95.35	92.23	97.13	93.40
f_4	0.5	94.41	92.29	97.13	93.40
f_5	0.1	96.93	92.88	96.77	92.92
f_5	0.3	96.77	93.40	96.96	93.89
f_5	0.5	96.77	92.92	96.96	94.32
f_5	0.7	97.14	94.32	96.96	94.32
f_5	0.9	96.96	94.32	96.96	94.32
f_6	2	96.05	93.89	96.41	94.32
f_6	5	96.77	94.32	96.96	94.32
f_6	10	96.96	94.32	96.96	94.32
f_6	20	96.96	94.32	96.96	94.32
f_6	50	96.96	94.32	96.96	94.32
f_7	0,0.3	96.03	92.36	95.84	94.32
f_7	0,0.5	95.48	92.33	95.84	93.40
f_7	0,0.8	95.68	92.29	96.20	92.88
f_7	0.3,0.8	93.84	89.25	94.49	91.48
f_8	0,0.3,20	96.42	93.89	96.42	93.89
f_8	0,0.5,20	96.60	94.32	96.60	93.89
f_8	0,0.8,20	96.96	94.81	96.96	94.38
f_8	0.3,0.8,20	94.93	92.39	94.93	92.88
f_8	0,0.3,50	96.42	93.89	96.42	93.89
f_8	0,0.5,50	96.60	93.89	96.60	93.89
f_8	0,0.8,50	96.96	94.38	96.96	94.38
f_8	0.3,0.8,50	94.93	92.88	94.93	92.88

BR tipo (c) (37.4 reglas)

Tabla 2.18. Resultados para Iris con BRs generadas con el Método 1 (2)

f	p	Ent.	Pr.	Ent.	Pr.	Ent.	Pr.	Ent.	Pr.
		g_1		g_2		g_1		g_2	
f_0	—	89.51	64.88	—	—	85.81	73.23	—	—
f_1	—	83.97	72.11	90.83	70.97	81.94	72.21	83.93	72.60
f_2	—	72.81	57.47	83.65	59.12	82.57	71.36	84.77	72.71
f_3	2	76.99	57.16	84.49	58.73	84.77	72.81	85.84	72.29
f_3	5	81.63	56.87	84.52	58.33	85.71	72.50	85.81	72.81
f_3	10	82.99	57.71	84.52	58.02	85.81	72.81	85.78	73.43
f_3	20	84.38	57.60	84.52	58.54	85.78	73.43	85.85	73.23
f_3	50	84.59	58.85	84.56	59.95	85.78	73.33	85.78	73.44
f_4	0.3	69.91	55.20	82.95	57.69	81.45	70.44	84.14	72.19
f_4	0.5	68.21	54.69	81.80	56.34	80.30	70.03	83.58	71.37
f_5	0.1	78.66	59.63	85.78	64.56	84.03	71.46	85.53	72.39
f_5	0.3	83.37	62.01	85.81	60.88	85.60	72.08	85.60	72.60
f_5	0.5	85.67	62.73	85.81	60.98	85.74	72.80	85.78	72.70
f_5	0.7	86.97	63.46	85.81	61.19	85.88	72.80	85.92	73.01
f_5	0.9	87.55	64.48	85.81	61.80	85.81	73.22	85.85	73.22
f_6	2	81.04	56.46	84.63	57.09	85.64	72.30	85.92	73.23
f_6	5	83.48	57.91	84.73	59.46	85.92	73.33	85.88	73.53
f_6	10	84.80	59.35	84.56	59.87	85.85	73.53	85.81	73.43
f_6	20	84.52	59.66	85.60	61.73	85.81	73.43	85.81	73.33
f_6	50	85.39	61.42	91.32	67.38	85.81	73.33	85.81	73.23
f_7	0,0.3	75.92	60.36	80.48	60.06	83.97	73.12	84.39	72.71
f_7	0,0.5	73.98	59.20	80.24	59.44	83.37	73.12	84.21	72.71
f_7	0,0.8	74.41	57.68	83.62	59.12	83.37	71.77	85.11	72.09
f_7	0.3,0.8	62.18	53.14	64.23	53.15	77.93	70.23	78.70	70.85
f_8	0,0.3,20	80.90	59.77	80.93	60.29	84.63	73.23	84.63	73.53
f_8	0,0.5,20	80.52	58.01	80.76	59.05	84.52	73.12	84.59	73.95
f_8	0,0.8,20	84.18	57.71	84.24	58.44	85.78	72.81	85.78	73.43
f_8	0.3,0.8,20	63.92	52.76	64.41	53.17	79.36	71.06	79.95	70.75
f_8	0,0.3,50	80.90	60.19	80.83	60.50	84.63	73.53	84.63	73.33
f_8	0,0.5,50	80.72	59.05	80.68	59.57	84.59	73.95	84.63	73.54
f_8	0,0.8,50	84.31	58.85	84.24	58.96	85.78	73.43	85.81	73.13
f_8	0.3,0.8,50	64.55	53.38	64.62	53.28	79.95	70.75	80.02	71.05
		BR tipo (a) (420.2 reglas)				BR tipo (b) (420.2 reglas)			

Tabla 2.19. Resultados para Pima con BRs generadas con el Método 1 (1)

f	p	Ent.	Pr.	Ent.	Pr.
		g_1		g_2	
f_0	—	81.94	74.16	—	—
f_1	—	79.22	72.14	80.24	72.92
f_2	—	79.22	73.24	80.20	73.13
f_3	2	80.20	73.13	81.25	73.74
f_3	5	81.45	73.74	81.70	74.06
f_3	10	81.70	74.06	81.94	74.26
f_3	20	81.94	74.26	81.94	74.27
f_3	50	81.94	74.27	81.94	74.16
f_4	0.3	79.02	73.24	80.24	73.14
f_4	0.5	78.67	72.83	80.13	73.35
f_5	0.1	80.48	73.14	81.28	73.64
f_5	0.3	81.21	73.95	81.73	74.47
f_5	0.5	81.70	73.95	81.70	74.16
f_5	0.7	81.80	74.16	82.01	74.16
f_5	0.9	82.01	74.06	82.01	74.15
f_6	2	81.66	73.54	82.05	74.06
f_6	5	81.91	74.16	81.94	74.47
f_6	10	81.94	74.47	81.91	74.58
f_6	20	81.91	74.68	81.98	74.27
f_6	50	81.98	74.27	81.98	74.16
f_7	0,0.3	79.54	73.14	80.16	73.33
f_7	0,0.5	79.02	73.34	79.57	73.14
f_7	0,0.8	79.40	73.75	80.16	73.44
f_7	0.3,0.8	75.50	72.20	76.33	72.11
f_8	0,0.3,20	81.18	74.16	81.18	74.16
f_8	0,0.5,20	81.21	74.05	81.25	74.06
f_8	0,0.8,20	81.87	74.26	81.87	74.16
f_8	0.3,0.8,20	75.95	72.41	75.84	72.62
f_8	0,0.3,50	81.18	74.16	81.18	74.06
f_8	0,0.5,50	81.25	74.06	81.25	73.95
f_8	0,0.8,50	81.87	74.16	81.87	74.06
f_8	0.3,0.8,50	75.88	72.62	75.74	72.31

BR tipo (c) (382 reglas)

Tabla 2.20. Resultados para Pima con BRs generadas con el Método 1 (2)

f	p	Ent.	Pr.	Ent.	Pr.	Ent.	Pr.	Ent.	Pr.
		g_1		g_2		g_1		g_2	
f_0	—	99.53	88.36	—	—	98.88	91.94	—	—
f_1	—	97.55	92.81	99.68	91.29	97.89	92.05	98.23	91.50
f_2	—	85.29	70.01	94.24	77.19	95.10	87.42	96.26	88.28
f_3	2	89.66	76.22	97.64	81.26	96.26	88.28	98.57	90.15
f_3	5	93.77	79.40	97.64	82.38	98.57	90.91	98.72	91.29
f_3	10	95.24	80.21	97.64	82.61	98.72	91.29	98.87	91.96
f_3	20	96.70	81.61	97.64	82.96	98.87	91.96	98.88	91.94
f_3	50	97.02	81.96	97.64	83.99	98.88	91.94	98.88	91.94
f_4	0.3	76.38	62.49	92.65	70.16	92.98	82.98	95.93	87.31
f_4	0.5	67.12	59.11	87.95	66.20	87.49	75.50	95.27	85.09
f_5	0.1	89.68	74.94	97.33	79.52	96.58	88.07	98.10	89.10
f_5	0.3	94.54	78.36	97.64	83.13	98.23	89.53	98.55	91.35
f_5	0.5	96.32	80.97	97.64	83.13	98.72	91.78	98.87	91.29
f_5	0.7	97.48	81.29	97.64	83.05	98.72	91.29	98.72	91.61
f_5	0.9	97.64	81.56	97.64	83.67	98.88	92.29	98.88	91.94
f_6	2	94.43	79.07	97.64	81.70	98.57	90.83	98.72	91.53
f_6	5	96.70	81.28	97.95	82.64	98.72	91.53	98.88	91.94
f_6	10	97.48	82.32	98.11	84.42	98.88	91.94	98.88	91.94
f_6	20	97.48	84.42	98.27	84.42	98.88	91.94	98.88	91.94
f_6	50	98.11	84.42	99.53	88.00	98.88	91.94	98.88	91.94
f_7	0,0.3	91.95	80.01	95.37	81.18	97.41	90.42	97.56	90.83
f_7	0,0.5	89.18	78.41	95.07	79.31	97.10	89.10	97.59	89.64
f_7	0,0.8	86.74	76.52	94.89	79.80	95.91	88.36	96.26	89.31
f_7	0.3,0.8	74.88	66.21	77.17	66.95	90.21	79.28	89.87	78.10
f_8	0,0.3,20	96.03	81.22	96.81	82.25	99.03	91.94	98.88	91.92
f_8	0,0.5,20	96.20	81.01	96.98	82.34	99.03	91.94	98.88	91.92
f_8	0,0.8,20	96.86	83.05	97.64	84.05	99.03	92.37	98.88	92.35
f_8	0.3,0.8,20	80.78	71.49	81.43	71.57	90.29	83.19	91.25	84.35
f_8	0,0.3,50	96.18	81.25	96.81	82.25	98.88	91.92	98.88	91.92
f_8	0,0.5,50	96.36	81.68	96.98	82.69	98.88	91.92	98.88	91.92
f_8	0,0.8,50	97.02	83.75	97.64	85.07	91.92	92.35	98.88	92.35
f_8	0.3,0.8,50	81.75	71.90	82.39	72.58	91.25	94.35	91.25	84.35
		BR tipo (a) (102 reglas)				BR tipo (b) (102 reglas)			

Tabla 2.21. Resultados para Wine con BRs generadas con el Método 1 (1)

f	p	Ent.	Pr.	Ent.	Pr.
		g_1		g_2	
f_0	—	98.54	91.94	—	—
f_1	—	97.91	92.73	98.23	91.94
f_2	—	97.58	92.05	97.74	91.94
f_3	2	97.74	91.94	98.08	92.02
f_3	5	98.07	92.76	98.23	92.29
f_3	10	98.23	92.29	98.39	92.29
f_3	20	98.39	92.29	98.54	91.94
f_3	50	98.54	91.94	98.54	91.94
f_4	0.3	96.76	90.97	97.74	91.94
f_4	0.5	95.77	92.49	97.74	91.94
f_5	0.1	97.74	92.05	97.74	91.94
f_5	0.3	98.22	92.43	98.22	92.70
f_5	0.5	98.38	92.97	98.53	91.97
f_5	0.7	98.53	92.30	98.54	92.29
f_5	0.9	98.54	91.94	98.54	91.94
f_6	2	98.39	92.24	98.23	91.53
f_6	5	98.23	91.53	98.39	91.94
f_6	10	98.39	91.94	98.23	91.94
f_6	20	98.23	91.94	98.54	91.94
f_6	50	98.54	91.94	98.54	91.94
f_7	0,0.3	96.92	91.18	97.39	91.94
f_7	0,0.5	97.05	91.15	97.39	91.50
f_7	0,0.8	97.58	91.70	97.73	91.94
f_7	0.3,0.8	88.98	87.36	89.72	85.72
f_8	0,0.3,20	98.37	92.29	98.37	91.94
f_8	0,0.5,20	98.21	92.29	98.37	91.94
f_8	0,0.8,20	98.39	92.29	98.54	91.94
f_8	0.3,0.8,20	90.35	85.42	89.84	85.85
f_8	0,0.3,50	98.37	91.94	98.37	91.94
f_8	0,0.5,50	98.37	91.94	98.37	91.94
f_8	0,0.8,50	98.54	91.94	98.54	91.94
f_8	0.3,0.8,50	89.84	85.85	89.67	85.52

BR tipo (c) (101.4 reglas)

Tabla 2.22. Resultados para Wine con BRs generadas con el Método 1 (2)

Capítulo 3

Método Multietapa de Aprendizaje Genético de Sistemas de Clasificación Basados en Reglas Difusas

En este capítulo presentamos un proceso genético multietapa de aprendizaje para la obtención de SCBRDs lingüísticos. Este proceso está basado en una metodología específica de aprendizaje genético, MOGUL (metología para la obtención genética de Sistemas Basados en Reglas Difusas bajo el enfoque de aprendizaje iterativo de reglas). El método integra los MRDs en la obtención de una BC, para conseguir la máxima cooperación con el MRD utilizado en la clasificación, y además determina el mejor conjunto de modificadores lingüísticos para los términos de las variables lingüísticas. De acuerdo con las líneas determinadas por la metodología MOGUL, el proceso contiene las siguientes etapas:

- La primera etapa genera una BR de forma independiente al MRD utilizado.
- En la segunda etapa, un proceso basado en AGs selecciona el mejor subconjunto de reglas difusas y aprende el mejor conjunto de modificadores para las variables lingüísticas en cooperación con el MRD.
- Finalmente, se realiza un proceso de ajuste genético de las particiones difusas con los modificadores lingüísticos aprendidos en la etapa anterior.

Previo a la presentación del proceso de aprendizaje, en la Sección 3.1 se presenta un estudio de las diferentes propuestas de diseño de SCBRDs utilizando AGs.

3.1 Obtención de la Base de Conocimiento de un Sistema de Clasificación Basado en Reglas Difusas mediante Procesos Evolutivos

En el Capítulo 1, hemos indicado que las tareas implicadas en el proceso de obtención de la BC de un SCBRD se pueden agrupar en torno a tres puntos:

- generación de la BR,
- simplificación de la BR, y
- ajuste de las particiones difusas utilizadas para cada una de las variables.

Estos problemas se pueden resolver de forma individual o conjunta. En esta sección describiremos brevemente algunos procesos evolutivos para la obtención de la BC de un SCBRD.

3.1.1 Generación Evolutiva de Reglas Difusas

Los AGs, aunque no son algoritmos de aprendizaje, se han utilizado para evolucionar conjuntos de reglas que describen clases o conceptos desde tres puntos de vista diferentes:

- *El enfoque Michigan*, que considera un cromosoma como una regla individual y representa la BR mediante la población al completo.
- *El enfoque Pittsburgh*, en el que cada cromosoma codifica una BR completa.
- *El enfoque de aprendizaje iterativo de reglas*, que considera, al igual que el enfoque de Michigan, que cada cromosoma codifica una única regla pero, a diferencia de éste, la solución está formada por el mejor individuo del AG. De este modo, el AG proporciona una solución parcial al problema de aprendizaje en cada ejecución.

En las siguientes subsecciones describimos estos modelos de aprendizaje y su aplicación al diseño de SCBRDs.

3.1.1.1 El Enfoque Michigan

Este modelo se basa en un AG en el que los cromosomas representan reglas individuales y la población al completo una BR. Los sistemas obtenidos de esta forma, a los que en la bibliografía especializada se les ha denominado *Sistemas Clasificadores*, son el resultado

de la evolución de reglas individuales a través de operaciones de asignación de méritos, de descubrimiento de reglas y operadores genéticos aplicados a nivel de regla individual.

Los algoritmos de aprendizaje que utilizan este enfoque tienen tres componentes principales:

1. Un *sistema de rendimiento* que interactua con el entorno.
2. Un *sistema de asignación de méritos* que determina la recompensa asociada a cada regla individual respecto al comportamiento global de la BR representada por la población al completo. En este aspecto se han utilizado tradicionalmente dos esquemas diferentes: el algoritmo de Bucket Brigade [Hol85] y el plan Profit-Sharing [Gre88].
3. Un *procedimiento de descubrimiento de clasificadores*, que genera reglas nuevas de un conjunto de reglas mediante AGs.

Holland fue el primero en presentar las ideas iniciales sobre Sistemas Clasificadores [Hol75] y quien desarrolló, junto con Reitman en [HR78], el primer Sistema Clasificador, CS-1. A partir de ese momento se han propuesto distintos Sistemas Clasificadores no difusos [Gol89, MS90, Wil94, Wil95] y algunos algoritmos basados en este enfoque para el diseño de SBRDs para control [VR91, PB93, NFUM96, BB97, Vel98].

En el diseño de SCBRD se sigue la filosofía del enfoque Michigan en el método propuesto por Yuan y otros en [YZ96] para la generación de BRs con cardinalidad mínima, que maximicen la propiedad de consistencia [Mic83] y el número de ejemplos clasificados correctamente. Este algoritmo se ha utilizado en esta memoria (en el Capítulo 2) para comparar sus resultados con los obtenidos con otros métodos.

El proceso evolutivo aprende reglas (expresadas en la forma normal disyuntiva propuesta por Michalski [Mic83]) en las que tanto las variables utilizadas para describir los ejemplos como la variable de clase, se consideran variables lingüísticas. Cada regla se codifica en binario en un cromosoma, en el que se distingue un segmento por variable. En cada segmento se incluyen tantos bits como etiquetas lingüísticas se utilicen para la variable correspondiente de forma que un 1 en la posición i implica que la variable a la que se asocia el segmento toma el valor de la etiqueta i en la regla. No se limita el número de unos en cada segmento, por lo que el esquema de codificación permite que el sistema determine las variables más relevantes para cada regla. Este esquema de codificación ya fue propuesto por González y otros en [GPV94] para SLAVE que utiliza un modelo de aprendizaje iterativo descrito en la sección 3.1.1.3.

La población inicial incluye un 50 % de individuos que representan reglas generadas a partir de los ejemplos, y el resto aleatoriamente. Esta población proporciona al proceso evolutivo un punto de partida mejor que la población totalmente aleatoria: incluye reglas que representan información precisa pero ajustada a los ejemplos que pueden ser generalizadas con la información que aportan las reglas generadas aleatoriamente.

A esta población se aplican dos operadores genéticos:

- El cruce en dos puntos aplicado a nivel de segmento para que las alteraciones genéticas se produzcan entre unidades con significado común. Además, el cruce se realiza entre dos individuos seleccionados aleatoriamente *de la misma especie*, es decir, que representan reglas con la misma clase en el consecuente. Esta limitación se impone por la creencia de que el cruce entre individuos de distinta especie (que representan información distinta) tiende a generar descendientes con rendimiento bajo [Hol75].
- La mutación aleatoria realizada a nivel de segmento, modificando la combinación de bits del segmento por otra combinación aleatoria.

Después de la aplicación de los operadores genéticos se verifica la viabilidad de los individuos para asegurar que no haya redundancia innecesaria en la población. El criterio de viabilidad determina la eliminación de individuos que representan una regla cubierta por otra regla en la población.

La función de adaptación auna tres objetivos: la maximización de la precisión en la clasificación de la regla, el grado de cobertura de la regla y su contribución en la clasificación correcta de los ejemplos respecto a la acción de la BR completa. El último objetivo evita la necesidad de un mecanismo de asignación de méritos característico de los procesos de aprendizaje evolutivos con el enfoque Michigan y orienta la búsqueda hacia un conjunto de reglas con mejor cooperación y menor solapamiento.

El criterio de selección es determinístico ya que, tras la aplicación de los operadores genéticos, los nuevos individuos sustituirán a los individuos más débiles y similares a ellos. De esta forma se preserva la calidad y diversidad de la población.

El proceso incorpora un *proceso de extracción de reglas*, realizado inicialmente cada generación y planteado posteriormente cada cierto número de generaciones (por eficiencia), que hace que este algoritmo ocupe un punto intermedio entre los algoritmos del enfoque Michigan y los del aprendizaje iterativo de reglas descrito en la Sección 3.1.1.3. Para generar un conjunto de reglas compacto y con calidad alta, cada cierto número de generaciones se seleccionan las mejores reglas en base a tres criterios aplicados secuencialmente: precisión en la clasificación, cobertura y valor de la función de adaptación. Las reglas seleccionadas se eliminan de la población, se incorporan a la BR final y se eliminan del conjunto de ejemplos de entrenamiento los ejemplos clasificados correctamente con ellas. El AG continúa hasta que no queden ejemplos en el conjunto de entrenamiento o no queden reglas candidatas.

Ishibuchi y otros desarrollan en [INM95] una propuesta de obtención genética de SCBRDs incluida en el modelo Michigan, en la que la función de adaptación es una combinación lineal de los ejemplos correcta e incorrectamente clasificados con la regla. El algoritmo no incluye ningún mecanismo de asignación de méritos. Utiliza los operadores

de cruce y mutación uniforme aplicados a nivel de etiqueta lingüística y un esquema de selección determinístico.

3.1.1.2 El Enfoque Pittsburgh

En este modelo, cada cromosoma codifica una BR completa [Smi80]. Los procesos incluidos en el mismo se describen a través de las características del AG:

- *Espacio de búsqueda y representación de reglas.* Habitualmente utilizan dos lenguajes para representar las reglas, la forma normal disyuntiva de Michalski [Mic83] o el lenguaje VL1 propuesto posteriormente por Michalski en [Mic86]. En la mayoría de los casos se utilizan, con cualquiera de estos lenguajes, cromosomas de longitud variable por el desconocimiento inicial del número de reglas necesarias para describir el sistema.

- *Función de adaptación.* En este caso el valor de adaptación se asigna a una BR completa y refleja habitualmente la verificación de conceptos de la teoría clásica de aprendizaje como consistencia y completitud [Mic83].

- *Operadores genéticos.* Los operadores genéticos propuestos son adaptaciones de los operadores clásicos al problema del aprendizaje de conceptos y a la longitud variable de los cromosomas, entre otros aspectos. En cualquier caso, el operador de cruce proporciona BRs nuevas y el de mutación incorpora nuevas reglas a la BR. Además, en ocasiones se introducen operadores nuevos para provocar una evolución más rápida hacia buenas soluciones, como por ejemplo operadores de generalización y especialización de reglas [DeJSG93].

Smith fue el primero que propuso esta filosofía de aprendizaje de reglas en [Smi80]. Algunos ejemplos de propuestas en entornos no difusos son los sistemas GABIL [DeJSG93] y GIL [Jan93].

Este enfoque no se ha aplicado a la generación de SCBRDs (en [CFM96, HP97, Mag97] se pueden encontrar algunos ejemplos de diseño de SBRD para control basados en el mismo), aunque las características del modelo y su codificación se reflejan en algunos procesos de selección y ajuste que se describen en las subsecciones 3.1.2 y 3.1.3.

Si comparamos los dos enfoques genéticos para el aprendizaje de reglas vemos que sus diferencias residen en el nivel en el que se aplican los AGs [CFM96]. Ambos tienen ventajas e inconvenientes. Como González y Herrera señalan en [GH97], el principal problema del enfoque Michigan es la forma de resolver los conflictos entre intereses particulares y colectivos de las reglas que integran el sistema. En un método del modelo Michigan, con los operadores de selección y variación aplicados a nivel de regla individual, dentro del AG compiten reglas que cooperan en procesos correctos de clasificación con otras que también tienen buen comportamiento. Además, en un SCBRD con un MRD que

integre la información proporcionada por las reglas disparadas (como los desarrollados en el Capítulo 2), este problema adquiere mayor dimensión.

En un método de aprendizaje del modelo Pittsburgh este conflicto entre intereses individuales y colectivos desaparece porque la competición se desarrolla entre BRs completas. Su inconveniente es el costo de mantenimiento y evaluación de poblaciones completas (tanto respecto a memoria como a tiempo de procesamiento).

3.1.1.3 El Enfoque de Aprendizaje Iterativo de Reglas

El enfoque de aprendizaje iterativo de reglas es un modelo de aprendizaje que aparece en referencias bibliográficas recientes [Ven93, CH97, GH97, GP98, CdJHL99] como alternativa a los enfoques Michigan y Pittsburgh. Este nuevo modelo considera, al igual que el enfoque Michigan, que cada individuo representa una regla individual pero, a diferencia de éste, aporta como solución sólo el mejor individuo. De este modo, el AG proporciona una solución parcial al problema de aprendizaje y necesita ejecutarse varias veces para obtener una BC completa. Cada vez que se genera una nueva regla difusa, se penaliza la zona del espacio en la que se localiza para evitar que se considere en sucesivas ejecuciones.

Este modelo reduce de forma considerable el espacio de búsqueda por la descomposición que realiza del problema. En la tabla 3.1 se muestra el tamaño del espacio de búsqueda considerado en los tres enfoques para un problema con M clases, N variables, con V valores por variable y que necesite k reglas para describir el sistema.

Modelo Genético	Tamaño
Pittsburgh	$k^{M(V+1)^M}$
Michigan	$M(V+1)^N$
Iterativo	2^N

Tabla 3.1. Tamaño del espacio de búsqueda para los distintos modelos genéticos de aprendizaje

Los primeros algoritmos que han utilizado este enfoque son SIA (Supervised Inductive Algorithm) [Ven93], para Sistemas de Clasificación no difusos, y SLAVE (Structure Learning Algorithm in Vague Environment) [GPV94, GP98], que permite el desarrollo de sistemas basados en reglas nítidas o difusas, para problemas de control o de clasificación.

En SLAVE, las características del AG incluido en el proceso de aprendizaje son:

- Genera reglas en forma normal disyuntiva modificada con consecuente crisp. Utiliza el mismo esquema de codificación binario que Yuan y Zhuang utilizaron posteriormente en [YZ96], al que añade la posibilidad de cambiar la granularidad del dominio de las variables combinando distintas etiquetas lingüísticas.

- Para orientar la búsqueda, la población inicial se obtiene a partir de reglas más específicas generadas a partir de un subconjunto aleatorio de los ejemplos de entrenamiento.
- Utiliza como operadores de variación el operador de cruce en dos puntos y la mutación aleatoria, ambos aplicados a nivel de bit.
- La función de evaluación proporciona una medida de la consistencia, completitud, simplicidad y nivel de descripción de las reglas generadas.

3.1.2 Proceso Genético de Selección de Reglas Difusas

La selección de un conjunto no redundante de reglas con un alto grado de cooperación entre ellas consistituye un problema importante en el diseño de un SCBRD. Esto es debido a que en muchos problemas reales de clasificación el número de variables es alto, por que cualquier proceso de aprendizaje tiende a proporcionar una BR con cardinalidad alta. Además, algunos algoritmos de aprendizaje inductivo utilizan heurísticas que favorecen la generación de gran número de reglas, lo que acentúa el problema.

Ishibuchi y otros desarrollan en [INYT93, INY93, INT94, INYT95, IMT97] un proceso genético para simplificar BRs en problemas de clasificación. Este proceso no modifica la semántica de las reglas difusas y parte de una BR obtenida mediante un proceso de generación desarrollado tambien por los autores en [INT92], y descrito en la sección 1.4.2.

El método de selección genética desarrollado por los autores en [INY93] y utilizado para la experimentación del Capítulo 2, tiene las siguientes características:

- Sigue la filosofía del enfoque Pittsburgh para la codificación de los cromosomas, ya que en cada cromosoma se representa una BR completa. Para ello, se utilizan individuos con codificación entera con un número de genes igual al número total de reglas de la BR sin simplificar, en los que un 1 en el gen i nota que la regla i se incluye en la BR simplificada, un -1 indica que no se incluye, y un 0 que la regla i no se ha generado según el proceso descrito en [INT92].
- Utiliza el operador de cruce aleatorio en un punto.
- Para potenciar la eliminación de reglas, emplea un operador de mutación sesgada, que considera los cambios de inclusión o eliminación de una regla, con distinta probabilidad.
- La función de adaptación es una combinación ponderada de dos objetivos, la maximización del número de ejemplos correctamente clasificados y la minimización de la cardinalidad de la BR. Concretamente, para un cromosoma C, tiene la siguiente expresión $f(C) = \max\{W_{NCC}(C) \cdot NCC(C) - W_s \cdot |C|, 0\}$, siendo $NCC(C)$ el

número de ejemplos correctamente clasificados con la BR que representa el cromosoma C, $|C|$ el número de reglas de la BR, y W_{NCC} y W_s dos pesos que verifican que $0 < W_s << W_{NCC}$.

Posteriormente, en otros trabajos como en [INT94], se elimina el segundo objetivo y se añade la minimización de la suma de términos lingüísticos utilizados por las variables. De esta forma se potencian las reglas correspondientes a una parción amplia, ya que desde el punto de vista de la adquisición del conocimiento una regla difusa con etiquetas lingüísticas correspondientes a una partición amplia son reglas más generales y válidas en un subespacio mayor del espacio de patrones. En [IMT97] se sigue este último enfoque, pero se modifican los pesos asociados a los dos objetivos de forma dinámica en cada generación para así orientar en distintas direcciones la búsqueda del AG de forma simultánea.

3.1.3 Ajuste Genético de las Particiones Difusas de las Variables Lingüísticas

Uno de los aspectos más difíciles de determinar en el diseño de cualquier Sistema Basado en Reglas Difusas (SBRD) y, al mismo tiempo, más determinante de su precisión, es la partición difusa utilizada. Tanto si el conjunto de reglas se obtiene a partir de la información aportada por un experto, como si se genera a partir de un proceso de aprendizaje automático, es necesario determinar o ajustar las funciones d e pertenencia para conseguir un mejor rendimiento en el sistema. Este ajuste se puede hacer a dos niveles:

- *A nivel de BD*, modificando los parámetros que definen las funciones de pertenencia de las etiquetas lingüísticas de forma global a todas las reglas. De esta forma el proceso de ajuste mantiene el carácter descriptivo del SBRD.
- *A nivel de BC*, ajustando para cada regla los parámetros que definen las funciones de pertenencia de las etiquetas lingüísticas que aparecen en la misma. Es un proceso de ajuste local que hace que la semántica de los términos lingüísticos dependa de la regla específica en que aparezcan. La BR obtenida de esta forma tiene carácter aproximativo [CH97, CH99].

En procesos de ajuste genético para SBRDs en modelado y control se han utilizado los dos enfoques [Kar91, HLV95]. Para problemas de clasificación podemos destacar los procesos evolutivos para el ajuste de un SCBRD descriptivo propuesto en [IM97], y aproximativo en [MPB95]. Como hemos mencionado, en esta memoria nos centramos en el diseño de SCBRDs con carácter descriptivo, por lo que consideraremos procesos de ajuste que no modifiquen la descriptibilidad lingüística del sistema.

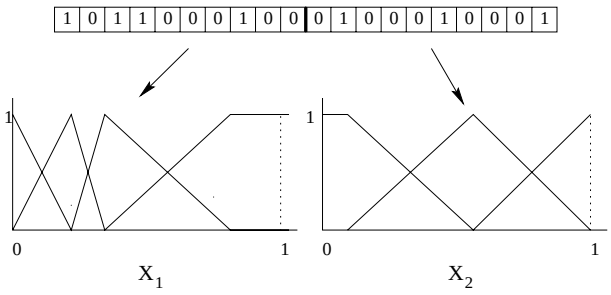

Figura 3.1. Esquema de codificación del proceso evolutivo de ajuste [IM97]

En [IM97] Ishibuchi y Murata desarrollan un proceso genético para determinar la partición difusa del espacio de patrones en un problema de clasificación. El proceso utiliza el método de generación de reglas descrito en [INT92] (Sección 1.4.2) para obtener la partición difusa (y por tanto la BC) que permite la obtención de un SCBRD con mejor comportamiento. La inclusión del método de generación de reglas en el proceso de ajuste, convierte a éste último en un método de aprendizaje de reglas difusas completo ya que el AG determina simultáneamente el número de reglas difusas y la función de pertenencia para cada conjunto difuso utilizado en el antecedente de las reglas.

El AG propuesto utiliza como esquema de codificación una extensión del esquema propuesto por Nomura y otros en [NHW92] en un proceso de ajuste de funciones de pertenencia para un sistema de control, con las siguientes diferencias:

- Permite que la partición para una variable esté formada por una única etiqueta, lo que es equivalente a no considerar la variable en la regla correspondiente.
- Permite funciones de pertenencia trapezoidales para las etiquetas de los extremos de los dominios, en contraste con la propuesta original diseñada para ajustar funciones de pertenencia triangulares.

Cada cromosoma representa una propuesta de particiones difusas para las variables lingüísticas consideradas. Para ello se codifica en binario, con longitud fija, y empleando un segmento de longitud predefinida por cada variable. En cada segmento, un 1 determina el centro de una función de pertenencia triangular (y a su vez, los extremos de los conjuntos soporte de los dos conjuntos difusos vecinos ya que todos los conjuntos difusos tienen punto de corte a la altura 0.5). En los bits de los extremos del segmento, un 0 indica que la función de pertenencia del valor lingüístico extremo es trapezoidal. En la figura 3.1 se puede ver un ejemplo de este tipo de codificación.

La población se genera aleatoriamente y se altera mediante tres operadores de recombinación: un operador de cruce múltiple entre segmentos, que equivale a intercambiar la partición de una variable entre dos individuos sin pérdida de significado, y dos operadores de mutación. El primer operador de mutación intercambia bits adyacentes dentro de un segmento, por lo que realiza un ajuste fino de la partición, una modificación leve del ancho

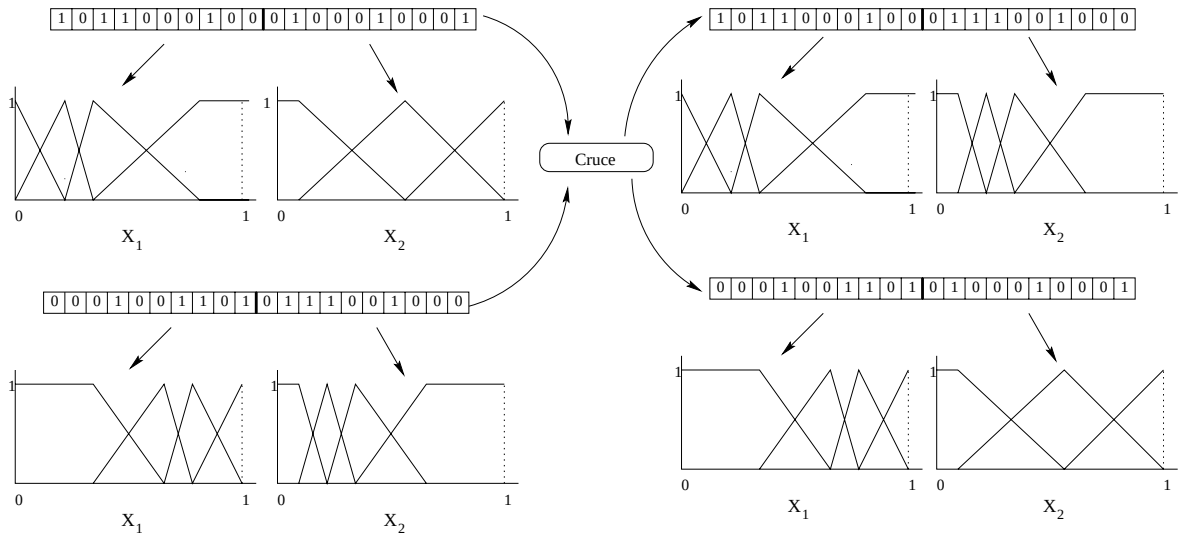

Figura 3.2. Operador de cruce del proceso evolutivo de ajuste [IM97]

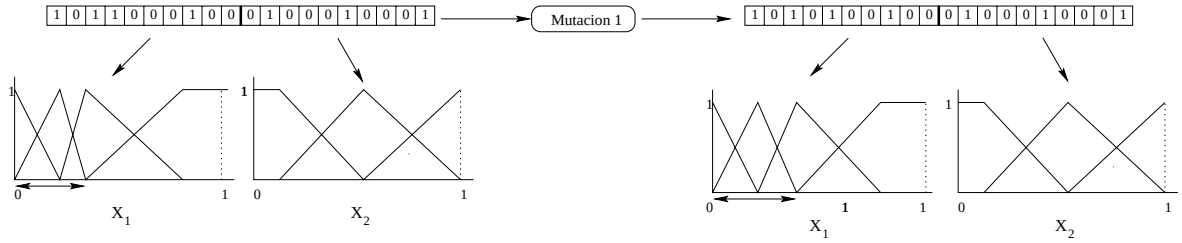

Figura 3.3. Operador de mutación tipo 1 del proceso evolutivo de ajuste [IM97]

del conjunto soporte del correspondiente conjunto difuso. El segundo operador cambia el valor de un bit con una determinada probabilidad, lo que equivale a añadir o eliminar un término lingüístico de una variable. Las figuras 3.2, 3.3 y 3.4 muestran el efecto de estos operadores.

La función de adaptación en este proceso de ajuste, al igual que en el proceso de selección propuesto por los mismos autores (Sección 3.1.2), agrupa dos objetivos: maximizar el número de ejemplos clasificados correctamente y minimizar el número de reglas.

El proceso depende en gran medida de la longitud de los segmentos determinante de la precisión del ajuste y por tanto del porcentaje de clasificación correcta alcanzable por el SCBRD. Un incremento excesivo de la longitud de los segmentos podría producir, como consecuencia de la expansión del espacio de búsqueda, un decremento de la eficiencia del AG, y sería necesario orientar la búsqueda del AG con la información disponible para obtener buenos resultados.

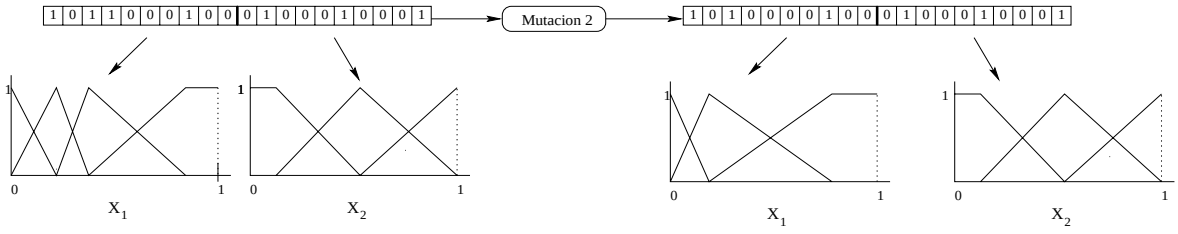

Figura 3.4. Operador de mutación tipo 2 del proceso evolutivo de ajuste [IM97]

3.2 Método Multietapa de Aprendizaje Genético de Sistemas de Clasificación Basados en Reglas Difusas

El proceso multietapa de aprendizaje genético de SCBRDs que presentamos en este capítulo está basado en la metodología de aprendizaje genético MOGUL [CdJHL99], consistente en un conjunto de pautas de diseño para la obtención de SBRDs capaces de resolver problemas de distinta naturaleza (modelado difuso, control difuso y clasificación difusa, entre otros).

MOGUL permite a distintos usuarios obtener sus propios SBRDs capaces de resolver diferentes problemas. Además, cualquier usuario puede añadir sus requerimientos particulares a las líneas de diseño de MOGUL para obtener cualquier tipo de SBRD que resuelva su problema de forma adecuada. Para ello, los usuarios deben diseñar sus propios procesos evolutivos para cada una de las etapas asegurándose que verifican las pautas de MOGUL.

En la sección siguiente describimos brevemente los aspectos clave de MOGUL, centrándonos en el enfoque de aprendizaje genético para un SBRD en general.

3.2.1 Aspectos Básicos de la Metodología: MOGUL

El principal problema a resolver en el diseño de un SBRD es la determinación de una representación adecuada capaz de reunir las características del problema y representar soluciones potenciales. Este aspecto se ha resuelto, como hemos visto en la sección anterior, desde tres enfoques genéticos distintos: El enfoque *Michigan*, *Pittsburgh* y de *aprendizaje iterativo de reglas*. MOGUL sigue este último enfoque.

El aprendizaje iterativo de reglas tiene como inconveniente que la cooperación entre las reglas de la BC generada no es tan buena como se desearía, debido a que naturaleza iterativa del proceso de generación provoca que éste no considere relaciones entre las reglas generadas. Éste es un problema importante puesto que la cooperación entre las

reglas de la BR es una de las principales características de los SBRDs por determinar el razonamiento interpolativo que desarrolla. Para resolver este problema, los SBRDs obtenidos mediante aprendizaje iterativo de reglas utilizan distintas técnicas [CGHP99]:

- Añadir nuevos criterios a la evaluación de cada regla para incluir este tipo de cooperación dentro del enfoque.
- Dividir el proceso de aprendizaje genético en al menos dos etapas, obteniendo así un método multietapa de aprendizaje genético de SBRDs. El objetivo de la segunda etapa (la etapa de postprocesamiento) es obtener la mejor cooperación posible entre reglas difusas generadas en la primera etapa (la etapa de generación) para conseguir una BC con buen comportamiento.

MOGUL está basado en la segunda técnica y define una etapa de postprocesamiento que mejora el nivel de cooperación entre las reglas difusas generadas en la etapa anterior, refinando o eliminando las reglas redundantes o innecesarias. Para mejorar la precisión del SBRD diseñado se tratan ambas tareas, la simplificación de la BC y el refinamiento de las reglas difusas que la componen, mediante el ajuste de sus funciones de pertenencia.

La etapa de postprocesamiento se divide así en dos etapas diferentes: el *proceso genético de multisimplificación* y el *proceso genético de ajuste*. El proceso de multisimplificación es capaz no sólo de obtener una BC simplificada, sino de proporcionar distintas definiciones de la misma con la mejor cooperación posible entre las reglas que la componen. El proceso evolutivo de ajuste se aplica a estas definiciones proporcionando como salida el SBRD finalmente obtenido. Este proceso puede provocar que una BC que no presenta el mejor comportamiento al final de la etapa de multisimplificación pueda ser la mejor después de la etapa de ajuste porque las modificaciones en las funciones de pertenencia permitan que las reglas cooperen de una forma mejor.

Otros aspectos importantes de la metodología MOGUL son los siguientes:

- El diseñador puede establecer la etapa de generación utilizando distintos tipos de algoritmos, no necesariamente de tipo evolutivo como ocurría en las propuestas existentes de procesos basados en el enfoque de aprendizaje iterativo de reglas. Se puede emplear un algoritmo no evolutivo (como se propone en esta memoria) o una EE en lugar del AG habitual. El modo de operación es el mismo, pero la diferencia es la velocidad del proceso de generación, que será mayor en el caso de un enfoque no evolutivo.
- La BC de un sistema basado en reglas tiene que verificar varias propiedades estadísticas importantes para presentar un buen comportamiento [GP98]. En una BC obtenida a través de MOGUL se considera la verificación de las propiedades de *completitud* y *consistencia* para mejorar el comportamiento del SBRD obtenido.

- Centrándonos en la búsqueda evolutiva, es necesario el uso de técnicas apropiadas para el desarrollo de una evolución adecuada en los espacios de búsqueda considerados en cada etapa. Para ello deben considerarse distintos factores para reducir la complejidad del espacio de búsqueda y realizar una exploración y explotación adecuada que permita que el proceso de búsqueda sea efectivo. En [CdJHL99] se propone el uso de distintas técnicas para estos aspectos.

- El conocimiento disponible se puede incorporar al proceso de aprendizaje genético para mejorar su rendimiento, bien incorporando directamente definiciones parciales obtenidas a partir de conocimiento experto, o bien utilizando el conocimiento disponible para generar la población inicial de los algoritmos evolutivos considerados en cada etapa.

3.2.2 Etapas del Método de Aprendizaje Genético de un Sistema de Clasificación Basado en Reglas Difusas

El proceso de aprendizaje de un SCBRD se aborda en esta memoria desde el punto de vista descriptivo, es decir, se busca un SCBRD que pueda trabajar de forma independiente o como herramienta de ayuda en procesos de decisión. Esto hace que deban ser consideradas en su diseño características como la descripción lingüística que haga posible la comprensión de los resultados aportados, robustez, versatilidad, facilidad de modificación y coherencia con el conocimiento previo.

El objetivo del método de aprendizaje propuesto es la obtención de un SCBRD capaz de alcanzar un alto grado de precisión en la predicción manteniendo las características mencionadas anteriormente. Esto determina la estructura de las reglas difusas utilizadas y el proceso con el cual se obtienen. Consideraremos un SCBRD formado por una BC descriptiva y un MRD específico. La BC estará formada por un conjunto de reglas difusas en el que las variables lingüísticas utilizadas en el antecedente tendrán un conjunto de valores asociados con significado real.

El diseñador del sistema tendrá que determinar tanto el número de términos lingüísticos utilizado para obtener un determinado nivel de granularidad, como las funciones de pertenencia iniciales de los conjuntos difusos asociados. Es un aspecto importante ya que que la discretización realizada sobre el dominio de las variables ejerce una fuerte influencia en la capacidad del sistema diseñado. Las limitaciones impuestas por una partición prefijada se superan con la fase de ajuste final del método de aprendizaje propuesto.

Sabemos que es difícil, incluso para un experto en el problema a resolver, conocer de forma exacta el significado más adecuado para una etiqueta lingüística específica y el más apropiado para el buen funcionamiento del sistema. Éste es un factor determinante en procesos de extracción automática de conocimiento ya que la representación de las etiquetas lingüísticas establece un valor numérico sobre la adecuación del concepto que represen-

tan las etiquetas. Una forma de resolver este problema sin perder el carácter lingüístico del sistema de clasificación, ni la granularidad determinada por el experto, es el uso de *modificadores lingüísticos* que nos permiten modificar la función de pertenencia utilizada para adaptarla al conjunto de entrenamiento. Los modificadores lingüísticos, cuya utilidad para representación de conocimiento en procesos de razonamiento aproximado ya fue subrayada por Zadeh en [Zad75], se han utilizado en SBRDs en [BD95, CYP96, MP96], entre otros.

Un modificador lingüístico es una función que permite alterar la función de pertenencia de los conjuntos difusos asociados a las etiquetas lingüísticas, dando como resultado una definición más o menos precisa dependiendo del caso. Dos de los modificadores más conocidos son el modificador de concentración "muy" y el de dilación "más o menos". El primero produce una reducción del grado de pertenencia de un valor en el conjunto difuso al que se aplique. El segundo es un modificador de dilación porque incrementa el grado de pertenencia. Sus expresiones son las siguientes:

$$\mu_{\text{Muy } A_i^k}(x) = \left(\mu_{A_i^k}(x)\right)^2$$
$$\mu_{\text{Más o menos } A_i^k}(x) = \sqrt{\mu_{A_i^k}(x)}$$

En la figura 3.5 podemos ver gráficamente su efecto sobre un conjunto difuso con función de pertenencia triangular.

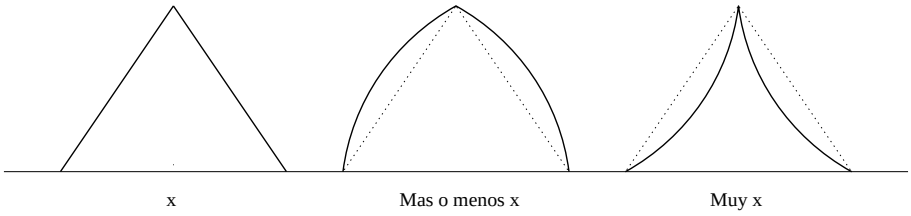

Figura 3.5. Modificadores Lingüísticos

Como veremos en las siguientes secciones, el proceso de aprendizaje incorpora una etapa de aprendizaje de modificadores lingüísticos incluida en el proceso de multiselección.

De acuerdo con lo expuesto, el algoritmo de aprendizaje se divide en tres etapas:

- Un *proceso de generación de reglas difusas*, que obtiene un conjunto de reglas lingüísticas de clasificación que representan el conocimiento presente en los ejemplos de entrenamiento.
- Un *proceso genético de multiselección* que, integrando el MRD y el aprendizaje de modificadores lingüísticos, genera distintas definiciones de la BC.
- Un *proceso genético de ajuste* a través del cual se obtienen los mejores valores para los parámetros de las funciones de pertenencia.

Las tres secciones siguientes están dedicadas a introducir cada uno de estos procesos.

3.3 Proceso de Generación de Reglas Difusas

El proceso de generación de reglas difusas de clasificación es la única etapa de diseño del SCBRD no resuelta con un AG. Su enfoque iterativo proporciona mayor eficiencia al proceso de generación. Ésto lo convierte en una etapa adecuada para realizar pequeños estudios experimentales del conjunto de parámetros más adecuado para el diseño de un SCBRD para un problema concreto, como el MRD más adecuado, o los parámetros del operador de agregación, entre otros.

Este proceso tiene dos componentes:

1. Un método de generación de reglas que extrae reglas difusas de clasificación del conjunto de ejemplos de entrenamiento.
2. Un método iterativo de cobertura que representa el modo de operación normal de la primera fase de los procesos de aprendizaje evolutivos basados en el enfoque de aprendizaje iterativo de reglas.

Las siguientes subsecciones muestran ambos métodos.

3.3.1 Método de Generación de Reglas Difusas

Este método comienza con una BD predefinida formada por una partición uniforme con funciones de pertenencia triangulares cruzadas a la altura 0.5 para cada variable. El diseñador del sistema especifica el número de términos lingüísticos que constituyen la partición de cada variable para obtener el nivel de granularidad deseado. En la figura 3.6 se muestra un ejemplo de este tipo de partición para una variable lingüística con cinco etiquetas {MP, P, M, G, MG}.

Hay que destacar en este punto que el método de generación de reglas difusas puede utilizar una BD definida completamente por el experto, si éste puede proporcionarla al sistema.

Como se indicó en el Capítulo 1, las reglas difusas tipo (c) incluyen en su expresión a las reglas tipo (a) y (b) por lo que, sin pérdida de generalidad y salvo que se indique lo contrario, describiremos el proceso de generación en base a ellas.

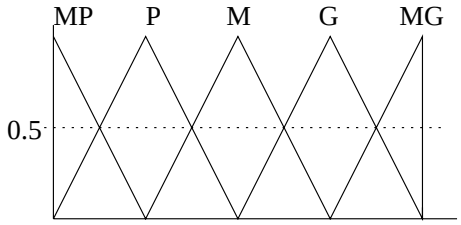

Figura 3.6. Partición difusa uniforme con funciones de pertenencia triangulares

En cada iteración del método de generación se obtiene un conjunto de reglas difusas candidato, generando una regla difusa para cada ejemplo de entrenamiento y excluyendo las reglas repetidas. Para ello el método determina el antecedente de la regla difusa que mejor cubre cada ejemplo de entrenamiento. Para reglas difusas tipo (b) o (c), el grado de certeza asociado a la clasificación del ejemplo en la clase C_j para la regla R_k se calcula con la expresión $\frac{S_j^k}{S^k}$ siendo S_j^k la suma de los grados de asociación para los ejemplos de la clase C_j pertenecientes al subespacio difuso delimitado por el antecedente, y S^k la misma suma, pero para ejemplos de cualquier clase.

De esta forma se obtiene un conjunto de reglas difusas candidatas y se selecciona la mejor regla considerando dos propiedades deseables en cualquier BR, la completitud y consistencia.

La propiedad de *completitud* implica que el sistema debe ser capaz siempre de proporcionar una salida (independientemente de su valor) ante cualquier entrada que reciba. Hemos modificado esta propiedad para incluir que la salida sea lo más cercana posible a la salida real, conocida al estar en un proceso de aprendizaje inductivo supervisado. Por esto consideraremos que una BR es completa cuando cubra todos los ejemplos de entrenamiento.

La propiedad de *consistencia* en una BR implica la ausencia de contradicciones en la misma, es decir, la ausencia de reglas con el mismo antecedente y distinto consecuente. Esta propiedad es difícil de verificar en la mayoría de los problemas reales y además difícil de analizar en el caso de SBRDs, en los que la consistencia se convierte en un problema de grado. Consideraremos una relajación de la propiedad de consistencia mediante el concepto de *ejemplos positivos* y *negativos* [GP98, HLV98a]. Un ejemplo se considera positivo para una regla difusa cuando pertenece a la región difusa delimitada por su antecedente y pertenece a la clase indicada en el consecuente. Por el contrario, un ejemplo se considera negativo cuando pertenece a la zona que representa el antecedente pero no pertenece a ninguna de las clases del consecuente con grado de certeza positivo.

La selección de la mejor regla intenta reflejar la verificación en cierto grado de las propiedades mencionadas y se realiza de acuerdo a una función multicriterio que considera los siguientes criterios [CH97]:

(a) *Un alto valor de frecuencia.* La frecuencia de una regla de clasificación R_k en un

conjunto de ejemplos $E = \{e^1, \ldots, e^p\}$ viene dada por la siguiente expresión:

$$\Psi_E(R_k) = \frac{\sum_{l=1}^{p} \sum_{j=1}^{M} b_j^k(e^l)}{p}$$

donde $e^l = (e_1^l, \ldots, e_N^l, C_j)$ y b_j^k es el grado de asociación del ejemplo e^l con la clase C_j según la regla R_k. Este grado de asociación será distinto de cero sólo para la clase a la que pertenece el ejemplo.

Las reglas que alcancen un valor más alto de frecuencia serán reglas más generales por representar de forma correcta una zona del espacio que incluye a una cantidad mayor de los ejemplos de entrenamiento.

b) *Un alto grado de cobertura sobre los ejemplos positivos.* El conjunto de ejemplos positivos para una regla R_k con un grado de asociación mayor o igual que ω se define mediante la siguiente expresión:

$$E_\omega^+(R_k) = \left\{ e^l \in E \ / \ \exists j \in \{1, \ldots, M\} \ \text{ tal que } \ b_j^k(e^l) \geq \omega \right\}$$

donde $n_\omega^+(R_k) = |E_\omega^+(R_k)|$.

El grado medio de cobertura en $E_\omega^+(R_k)$ se puede definir como indica la siguiente expresión:

$$G_\omega(R_k) = \sum_{e^l \in E_\omega^+(R_k)} \frac{\sum_{j=1}^{M} b_j^k}{n_\omega^+(R_k)}$$

c) *Penalización asociada a la no verificación de la propiedad de k-consistencia.* El conjunto de ejemplos negativos para R_k se define como:

$$E^-(R_k) = \{ e^l \in E \ / \ b_j^k(e^l) = 0 \ \ \forall j \in \{1, \ldots, M\} \ \text{ y } \ R^k(e^l) > 0 \}$$

Se considera que un ejemplo es negativo para una regla cuando pertenece a la zona difusa determinada por su antecedente pero no pertenece a la clase (o clases) con grado de certeza positivo en el consecuente de la misma. El conjunto de ejemplos negativos se considera siempre sobre el conjunto de entrenamiento al completo ya que si se calculase a partir del conjunto de ejemplos actual podrían generarse reglas difusas inconsistentes con otras obtenidas anteriormente, por no considerarse la influencia de ejemplos eliminados en su generación.

Este criterio penaliza reglas difusas con muchos ejemplos negativos respecto al número de ejemplos positivos con un grado de asociación mayor o igual que ω. De esta forma, penaliza la no verificación de la k-consistencia [GP98]. La *función de penalización sobre el conjunto de ejemplos negativos de la regla* R_k [HLV98a] es

$$g_n(R_k{}^-) = \begin{cases} 1, & \text{si } n_{R_k}^- \leq k \cdot n_\omega^+(R_k) \\ \frac{1}{n_{R_k}^- - k n_\omega^+(R_k) + exp(1)}, & \text{en otro caso} \end{cases}$$

donde $n^-_{R_k} = |E^-(R_k)|$ es el número de ejemplos negativos.

Los tres criterios se combinan en una función de evaluación a través de una función de agregación creciente en sus tres componentes. En esta memoria trabajaremos con el producto:

$$F(R_k) = \Psi_E(R_k) \cdot G_\omega(R_k) \cdot g_n(R_k^-)$$

De esta forma, el método de generación de reglas difusas sigue el siguiente esquema:

1. Asignar al conjunto de reglas candidatas, B^c, el conjunto vacío.
2. Para cada ejemplo de entrenamiento $e^l \in E$, generar la regla difusa R_k que mayor grado de asociación tiene con el ejemplo, tomando para cada atributo la etiqueta lingüística a la que tenga mayor grado de pertenencia el valor del ejemplo para ese atributo. Incluir esta regla en el conjunto de reglas candidatas B^c si no se ha incluido antes.
3. Evaluar todas las reglas difusas de B^c y seleccionar la que maximice el valor de la función de seleccion de reglas.

3.3.2 Método de Cobertura

El método de cobertura está basado en un algoritmo iterativo que permite la obtención de un conjunto de reglas que cubre el conjunto de ejemplos [HLV98a]. En cada iteración, se ejecuta el *método de generación* para obtener la mejor regla de clasificación en función del estado actual del conjunto de entrenamiento, se considera el valor de la cobertura relativa que la regla ocasiona sobre el conjunto de ejemplos, y se eliminan de él los ejemplos con un grado de cubrimiento mayor que un valor ϵ proporcionado por el diseñador.

El *método de cobertura* se muestra en la figura 3.7 y se desarrolla en tres etapas:

1. *Inicialización*, en la que se introducen los valores de los principales parámetros del método, es decir, k, ω y ϵ. Además se inicializa el valor del grado de cobertura (CV) de cada uno de los ejemplos de entrenamiento a cero y el conjunto final de reglas (B^g) al conjunto vacío. En esta etapa se puede añadir conocimiento experto en forma de reglas. Para ello, se almacenan las reglas disponibles en B^g y se considera su cobertura, eliminando ejemplos del conjunto E.
2. Se aplica el *método de generación de reglas difusas* sobre el conjunto de ejemplos E para la obtención de la mejor regla difusa de clasificación R_k de acuerdo al estado actual de E. Esta regla se incluye en el conjunto final de reglas B^g.

Metodo de cobertura
inicio
Introducir el valor de los parametros k, ω, ϵ
Para $l = 1$ **hasta** *N_ejemplos_entr* **hacer**
$CV[l] \longleftarrow 0$
fin_para
$B^g \longleftarrow \emptyset$
Mientras $E \neq \emptyset$ **hacer**
Aplicar el metodo de generacion para obtener R_k
$B^g \longleftarrow B^g \cup R_k$
Para cada $e^l = (e^l_1, \ldots, e^l_N, C_j) \in E$ **hacer**
$CV[l] \longleftarrow CV[l] + b^k_j(e^l)$
Si $CV[l] \geq \epsilon$
entonces $E \longleftarrow E - e^l$
fin_para
fin_mientras
fin

Figura 3.7. Método de cobertura

3. Se incrementa el grado de cobertura de cada ejemplo con el grado de asociación con la regla R_k. Se eliminan del conjunto E aquellos ejemplos que superen el grado de cobertura máximo establecido por el diseñador ϵ.

4. Se repiten los pasos 2 y 3 hasta que $E = \emptyset$.

3.4 Proceso de Multiselección Genética

El método de generación de reglas descrito en la sección anterior obtiene un conjunto de reglas difusas que verifica las propiedades de completitud y k-consistencia [GP98, HLV98a]. No obstante, por el carácter iterativo del proceso, la BC resultante puede contener reglas redundantes o innecesarias que lleven a un funcionamiento inadecuado del SCBRD. Para resolver este problema, en esta sección proponemos un proceso de multiselección genética que, partiendo de la BC de la etapa de generación, obtiene distintas BCs simplificadas formadas por subconjuntos de reglas con buena cooperación entre ellas y con el MRD utilizado por el sistema.

Este proceso de multiselección incluye:

- La *técnica secuencial de nichos* [BBM93] que induce nichos, utilizando como técnica básica de optimización, el proceso de selección genética propuesto por Herrera y otros en [HLV98a], iterado en cada ejecución del proceso de multiselección.
- Un proceso de búsqueda del mejor conjunto de modificadores lingüísticos asociados a las etiquetas lingüísticas de las variables.
- La intervención del MRD utilizado por el sistema en el proceso de selección de reglas y modificadores.
- Un proceso de búsqueda local posterior a cada proceso de selección que, para el mejor individuo, es decir, para la mejor BC, determina la mejor modificación añadiendo o eliminando una regla y/o añadiendo o eliminando un modificador lingüístico.

Mediante estos subprocesos se obtienen distintas definiciones de BCs seleccionando las reglas que mejor cooperan entre ellas y con el MRD, y que mejor comportamiento muestran con los modificadores seleccionados.

En las siguientes subsecciones introducimos los AGs con nichos, describimos el método genético básico de selección de reglas de clasificación y la composición del proceso de multiselección.

3.4.1 Algoritmos Genéticos con Nichos

En los distintos procesos evolutivos descritos en esta memoria hemos señalado la capacidad de búsqueda de los AGs para determinar un óptimo global de una función en espacios complejos. No obstante, en muchos casos la función a optimizar puede tener varios óptimos que nos interese obtener (con lo cual estaríamos ante un *proceso de optimización multimodal*) o puede estar formada por más de un criterio u objetivo (*proceso de optimización multiobjetivo*). En el diseño de un SCBRD se deben optimizar distintos criterios en función del objetivo final del mismo: precisión, simplicidad, consistencia, completitud, etc. Además, puede interesarnos enfocar el problema de simplificación de forma que el objetivo final sea la obtención no de la mejor solución, el mejor SCBRD, sino de un conjunto de soluciones no dominantes entre las que los usuarios finales seleccionen una solución en función de su preferencia. Esta idea ya la utilizan Ishibuchi y otros en el proceso de selección propuesto en [IMT97], transformando el AG propuesto en un AG multiobjetivo que obtiene el mejor conjunto de soluciones no dominantes para el problema de clasificación.

Para ambos tipos de problemas, optimización multimodal y multiobjetivo, es necesario el uso de técnicas alternativas en el AG que eviten su convergencia hacia zonas del espacio donde se encuentra el mejor o los mejores óptimos locales abandonando la búsqueda en las zonas restantes. La teoría de nichos y especies es una alternativa para evitar este tipo de comportamiento [Hol75, DG89, Gol89, Mic96].

Intuitivamente podemos definir un *nicho* como el papel o trabajo desarrollado por un individuo dentro de un entorno y una *especie* como una clase de organismos con características comunes. Esta separación del entorno y de los individuos que explotan dicho entorno en distintos subconjuntos es muy común en la naturaleza y, al igual que otros aspectos biológicos, se ha trasladado al área de los AGs. Los *AGs con nichos* provocan la formación de subpoblaciones de individuos estables (especies) que exploran zonas parciales del espacio de búsqueda (nichos) obligando a individuos similares a compartir los recursos disponibles entre ellos.

Existen distintos métodos para inducir la formación de nichos en una población. Una de las formas de provocar la formación de especies y la creación de nichos se basa en el *esquema de compartición de valores de adaptación* entre individuos que determina una degradación del valor de adaptación del individuo en función de su distancia a individuos vecinos (métrica que puede ser definida a nivel genotípico o fenotípico). La función de compartición se basa en un parámetro $\sigma \in [0,1]$ que determina el radio del nicho definido en cualquiera de los espacios solución. Una de las posibilidades para su expresión es la *ley de la potencia* definida de la siguiente forma:

$$P(d) = \begin{cases} 1 - \left(\frac{d}{\sigma}\right)^{\alpha}, & \text{si } d < \alpha \\ 0, & \text{en otro caso} \end{cases}$$

siendo d un valor proporcionado por una métrica $d(C_r, C_s)$, que calcula la distancia entre dos cromosomas, y α un valor real que determina el grado de penalización a aplicar.

La nueva función de adaptación viene dada por

$$F(C_r) = \frac{f(C_r)}{\sum_{t=1}^{M} P(d(C_r, C_t))}$$

siendo M el tamaño de la población. De esta forma, se penaliza el valor de adaptación inicial asociado al cromosoma por la presencia de otros individuos en su vecindario. Como resultado, esta técnica limita el crecimiento incontrolado de una especie dentro de una población [Gol89].

El funcionamiento correcto de este esquema de nichos depende del valor del parámetro σ que debe ser igual a la máxima distancia necesaria entre los cromosomas para que se formen el mismo número de nichos que de picos en el espacio de búsqueda multimodal. En [DG89] se proponen dos métodos teóricos para el cálculo de este valor según se trabaje con un esquema de compartición fenotípico o genotípico.

Para evitar la complejidad de cálculo de esta función de compartición para todos los individuos, Beasly y otros proponen en [BBM93] la *técnica secuencial de nichos*. El algoritmo utiliza una función de distancia y la función de adaptación del AG y se basa en la siguiente idea: cuando se encuentra una solución, se modifica la función de evaluación para penalizar la zona del espacio de búsqueda en la que ésta se encuentra (puesto que ya se ha encontrado y no nos interesa reencontrar la misma solución de nuevo). De esta forma, las siguientes generaciones incorporan el conocimiento descubierto en generaciones anteriores,

a diferencia de la técnica iterativa simple en la que, para obtener varios óptimos, cada ejecución comienza con una población generada aleatoriamente. Los pasos del algoritmo son los siguientes:

1. *Inicialización:* Asignar a la función de adaptación modificada, la expresión de la función de adaptación original.
2. Ejecutar el AG almacenando el mejor individuo encontrado en la ejecución.
3. Actualizar la función de adaptación modificada para penalizar la región del espacio de búsqueda cercana al individuo.
4. Si no se han generado todas las soluciones que buscamos, volver al paso 2.

Este es el esquema que se sigue en el proceso de multiselección propuesto. En la siguiente sección describimos el método básico de selección que constituye el paso 2, y posteriormente analizaremos las características del proceso de multiselección en el que se encuadra.

3.4.2 Método Básico de Selección Genética

El proceso de selección genética elimina reglas innecesarias de la BR obtenida en la etapa anterior y busca el mejor conjunto de modificadores para ese conjunto de reglas. El aprendizaje de modificadores se puede realizar desde distintos puntos de vista:

- Asociando un modificador al conjunto difuso asociado a cada etiqueta lingüística de las particiones de la BD. En este caso, el conjunto de modificadores es común para todas las reglas de la BR.
- Asociando un modificador a cada etiqueta lingüística en cada regla difusa de la BR.

En el primer caso, la semántica relacionada con las variables lingüísticas es uniforme para todas las reglas y se especifica en la BD. En el segundo, el significado es específico para cada regla, pero se mantiene la naturaleza descriptiva del SCBRD. En esta memoria, nos referiremos al primer y segundo tipo de modificadores como modificadores tipo I y II, respectivamente.

El proceso de selección está basado en un AG que utiliza asignación de probabilidades por ordenación lineal y el procedimiento de muestreo estocástico universal junto con el esquema de selección elitista como operador de selección. La descendencia se genera mediante el cruce clásico multipunto (en este caso, en dos puntos) y el operador de mutación uniforme.

El esquema de codificación genera individuos de longitud fija con dos partes diferenciadas, una relativa a las reglas seleccionadas y la otra referente a los modificadores asociados a las etiquetas lingüísticas. Si consideramos que en la etapa de generación se ha obtenido un conjunto B^g con L reglas, dependiendo del tipo de proceso de aprendizaje de modificadores lingüísticos que queramos realizar, tendremos uno de los dos siguientes esquemas de codificación:

- *Modificadores tipo I:* En este caso la longitud del cromosoma es $h = L + \sum_{i=1}^{N} l_i$ siendo l_i el número de etiquetas lingüísticas de la variable i. Un cromosoma $C = (c_1, \ldots, c_h)$ se divide en dos partes: La primera contiene tantos bits como reglas se generen en la primera etapa, es decir L bits. De esta forma el primer fragmento del cromosoma $c_1, \ldots, c_L$ representa un subconjunto candidato de reglas B^s de la siguiente forma:

  $$\text{Si } c_i = 1 \text{ entonces } R_i \in B^s, \text{ si no } R_i \notin B^s$$

  La segunda parte tendrá tantos genes como términos lingüísticos diferentes se consideren por variable. Para estos genes se utilizan tantos dígitos distintos como modificadores lingüísticos distintos intervengan en este proceso de aprendizaje. Por ejemplo, si utilizamos los modificadores "más o menos" y "muy" podemos codificar la información de cada gen con uno de los siguientes valores: 0, si el término lingüístico correspondiente no tiene modificador, 1 si se le aplica el modificador "más o menos" y 2 si el modificador considerado es "muy".

  En la figura 3.8 mostramos un ejemplo de este esquema de codificación y la BC resultante (con una BR formada por reglas tipo (c)). En el esquema, los valores 1 y 2 en la parte del cromosoma relativa a modificadores, representan los modificadores "más o menos" y "muy", y el valor 0 está asociado a la función de pertenencia original sin modificadores.

- *Modificadores tipo II:* La longitud del cromosoma es $h' = L \cdot (N + 1)$ siendo N el número total de variables. De nuevo, el cromosoma se divide en dos partes. En la primera se sigue el mismo esquema de codificación indicado en el apartado anterior. Los $L \cdot N$ genes restantes representan los modificadores para cada una de las etiquetas lingüísticas de cada una de las reglas.

  En la figura 3.9 mostramos el esquema de codificación y el tipo de BC resultante.

En ambos casos, la población inicial se genera introduciendo un cromosoma que representa el conjunto de reglas obtenido en la fase de generación al completo, es decir con $c_i = 1 \quad \forall i \in \{1, \ldots, L\}$ y sin modificadores. Además, para cada tipo de modificador considerado, se introduce un individuo que representa la BR inicial al completo y con todos los genes relativos a modificadores igual a ese modificador. El resto de la población se genera aleatoriamente.

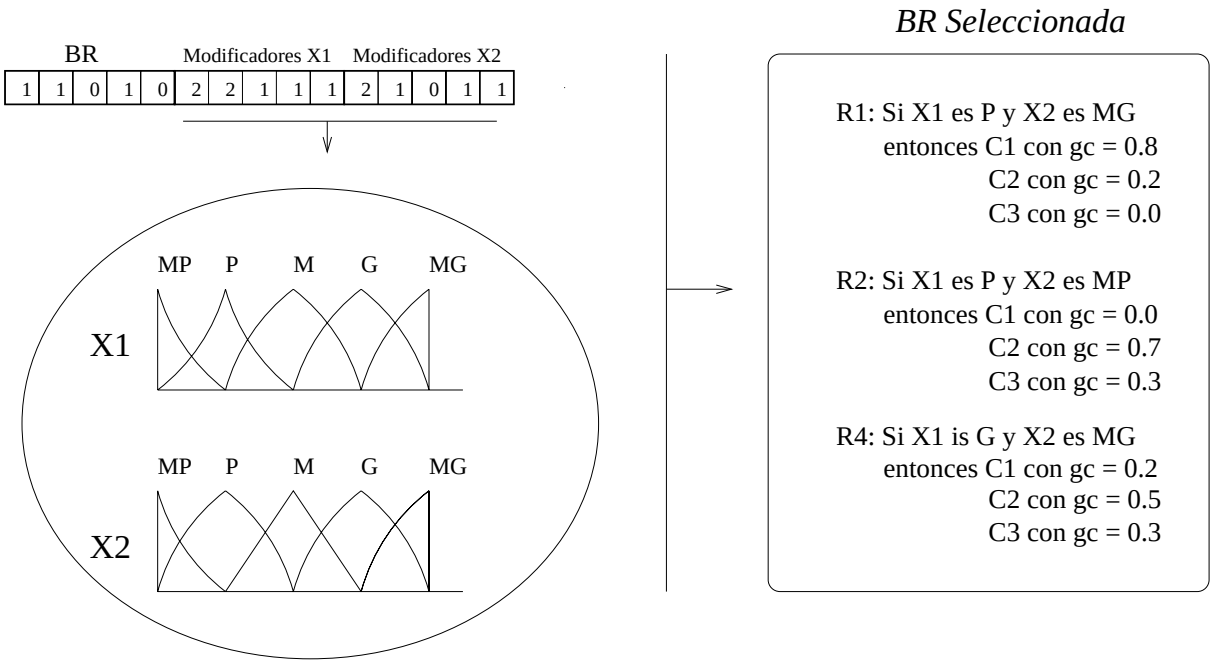

Figura 3.8. Esquema de codificación para modificadores tipo I

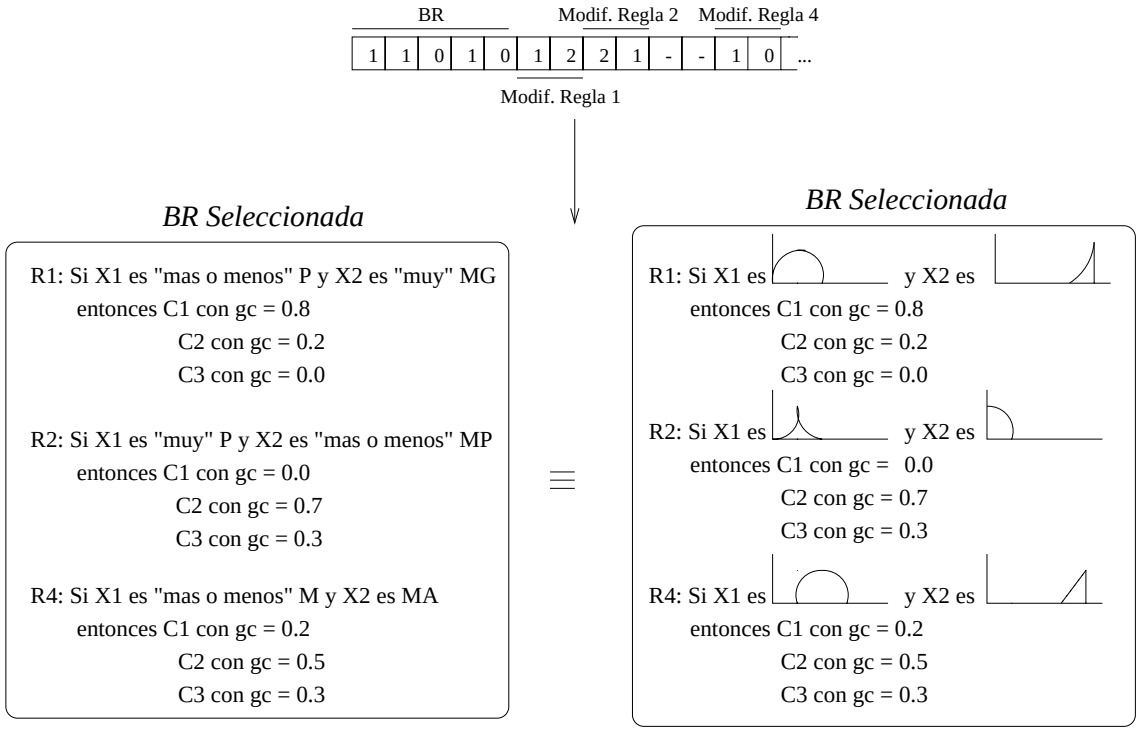

Figura 3.9. Esquema de codificación para modificadores tipo II

La función de adaptación $F(\cdot)$ está basada en el porcentaje de error del SCRBD sobre el conjunto de entrenamiento definido de la siguiente forma:

$$Error(C) = \frac{\text{número de errores}}{\text{número total de casos}}$$

Como se comentó al principio de la sección, el proceso de multiselección desarrollado tiene como objetivo la obtención de un SCBRD con máxima precisión que verifique las propiedades de completitud y k-consistencia. Aseguramos la verificación de la propiedad de completitud forzando que cada ejemplo de entrenamiento $e^l = (e^l_1, \ldots, e^l_N, C_j)$ sea cubierto por la BC codificada en el cromosoma C, $R(C)$, en un grado mayor o igual que un parámetro τ especificado por el diseñador del sistema:

$$C_{R(C)}(e^l) = \bigcup_{k=1\ldots L} b^k_j(e^l) \geq \tau \quad \forall e^l \in E, \quad R_k \in R(C) \quad \text{y} \quad j = 1, \ldots, M$$

donde τ es el grado de completitud mínimo aceptado en el proceso de selección y L es el número total de reglas.

De esta forma, el *grado de completitud del conjunto de entrenamiento* E, al que denominamos GCCE en la expresión, viene dado por

$$GCCE(R(C), E) = \bigcap_{e^l \in E} C_{R(C)}(e^l)$$

y la función de adaptación final que penaliza la no verificación de la propiedad de completitud es

$$F(C_j) = \begin{cases} Error(C), & \text{si } GCCE(R(C), E) \geq \tau \\ 1, & \text{en otro caso} \end{cases}$$

3.4.3 Proceso Genético de Multiselección

El proceso genético de multiselección utiliza la técnica secuencial de nichos [BBM93] para inducir nichos en el espacio de búsqueda y obtener diferentes definiciones de la BC. En cada iteración se utiliza el proceso de selección genética propuesto en la subsección anterior.

Cada vez que el proceso de selección genética obtiene una nueva definición de la BC, el proceso de multiselección penaliza la zona del espacio de búsqueda en la que está localizada para no seleccionarla en futuras ejecuciones. Se utiliza un *esquema de compartición genotípico* [DG89] para penalizar individuos de acuerdo a su proximidad a soluciones encontradas previamente. Para ello, definimos una *distancia métrica* que, dados dos individuos, devuelve un valor determinante de su cercanía. El esquema de codificación

utilizado, que incluye una parte del cromosoma con codificación entera, no permite el uso de funciones como la distancia de Hamming, por lo que proponemos la siguiente función de distancia: Si $A = (a_1, \ldots, a_h)$ y $B = (b_1, \ldots, b_h)$ son dos cromosomas, la función de distancia viene dada por la siguiente expresión

$$D(A,B) = \sum_{i=1}^{h} d_i$$
$$d_i = \begin{cases} 1, & \text{si } a_i \neq b_i \\ 0, & \text{en otro caso} \end{cases}$$

La *función de adaptación modificada*, que orienta la búsqueda del proceso de multiselección, está basada en la modificación del valor asociado a un individuo por la función de adaptación básica del algoritmo. Para ello se multiplica la función original por una función de penalización, $G(C_j, S)$ que penaliza la cercanía del individuo a las soluciones obtenidas previamente.

$$F^{'}(C_j) = F(C_j) \cdot G(C_j, S)$$

donde F es la función de adaptación del proceso genético de selección, $S = \{s_1, \ldots, s_k\}$ es el conjunto que contiene las k soluciones (definiciones de BCs) encontradas hasta el momento, y G es un tipo de función de penalización. Nosotros utilizaremos la siguiente función, teniendo en cuenta que tratamos con un problema de minimización:

$$G(C_j, S) = \begin{cases} \infty, & \text{si } d = 0 \\ 2 - (\frac{d}{r})^{\beta}, & \text{si } d < r \text{ y } d \neq 0 \\ 1, & \text{si } d \geq r \end{cases}$$

donde d es la mínima distancia entre el cromosoma C_j y las soluciones s_i incluidas en S, es decir, $d = \min_i\{H(C_j, s_i)\}$, por lo que la penalización se considera para la solución más próxima; r es el radio del nicho; y β es el factor de potencia que determina cómo de cóncava ($\beta > 1$) o convexa ($\beta < 1$) es la curva de penalización. De esta forma, la penalización toma máximo valor cuando el individuo C_j codifica una de las soluciones encontradas, y no hay penalización cuando C_j está a una distancia de cualquier solución perteneciente a S mayor o igual que el radio del nicho r.

En este proceso se incluye un algoritmo de búsqueda local para optimizar localmente cada definición de la BC obtenida, insertando o eliminando una regla y/o cambiando un modificador, cambios que llevarán a una mejora del comportamiento de la BC. Como se puede observar, es un proceso de optimización muy simple y rápido.

Esta búsqueda local se realiza al final de cada iteración del proceso de multiselección y se divide en dos fases: primero se optimiza la selección de reglas mediante una búsqueda en el espacio de BRs con distancia uno a la solución obtenida, es decir, con una regla más o menos que la BR obtenida. Para reducir el espacio de búsqueda, una vez finalizada

la optimización de la selección de reglas, se determina el mejor conjunto de modificadores con distancia uno al conjunto representado en el cromosoma resultado de la etapa correspondiente del proceso de multiselección.

El algoritmo del proceso genético de multiselección es el siguiente:

1. Inicialización. Asignar la función de adaptación del proceso genético básico de selección a la función de adaptación modificada.
2. Ejecutar el proceso genético básico de selección, utilizando la función de adaptación modificada, almacenando el mejor individuo encontrado en la ejecución.
3. Ejecutar el proceso de optimización local para optimizar la definición de la BC obtenida.
4. Actualizar la función de adaptación modificada, penalizando la región cercana al individuo obtenido.
5. Si no se han obtenido todas las definiciones de BCs requeridas, volver al paso 2.

Por tanto, el número de ejecuciones del algoritmo secuencial (etapas del proceso de multiselección) es el número de soluciones a obtener, es decir, el número de BCs a generar. El diseñador del sistema debe determinar, además de este valor, los valores de los parámetros r y β.

3.5 Proceso de Ajuste Genético

En el método de aprendizaje propuesto en esta memoria, las reglas se generan inicialmente a partir de una partición difusa, generalmente uniforme, determinada por el diseñador del sistema. Éste es uno de los aspectos más difíciles de determinar en el diseño de cualquier SCBRD y, al mismo tiempo, uno de los que más determinan su precisión, por lo que es necesario realizar un proceso posterior de ajuste de la funciones de pertenencia para incrementar el rendimiento del SCBRD.

Como hemos mencionado, este proceso de ajuste puede realizarse a dos niveles: a nivel de BD y a nivel de BC. Para mantener el carácter descriptivo del SCBRD a obtener, nuestra etapa de ajuste optimiza los parámetros que definen las funciones de pertenencia de los conjuntos difusos a nivel de BD mediante un AG. Para ello, el proceso genético de ajuste comienza con un conjunto de particiones difusas predefinidas -uniformes y con funciones de pertenencia triangulares como las que se muestran en la figura 3.10 o definidas por un experto, si se dispone de esa información- y determina un nuevo conjunto de

particiones difusas en el que se modifican los conjuntos difusos tanto en posición como en la amplitud del conjunto soporte correspondiente.

Cada cromosoma de la población codifica una definición diferente de la BD, la cual se combina con la BR existente para evaluar la adaptación del individuo. El AG que hemos diseñado para este proceso de ajuste utiliza codificación real [HLV98b] y el muestreo estocástico universal, junto con el esquema de selección elitista como procedimiento de selección.

Los conjuntos difusos considerados en las particiones difusas iniciales tienen función de pertenencia triangular, por lo que cada función de pertenencia tiene asociada una representación paramétrica basada en tuplas de 3 valores reales. Esto hace que una partición difusa se pueda representar mediante un vector de $3 \cdot l$ valores reales, siendo l el número de términos lingüísticos que constituyen el conjunto de términos de la variable. La definición completa de una BD para un problema con N variables se describe en un cromosoma C_r con codificación real de longitud fija, uniendo la representación parcial de la partición difusa de cada variable, de la siguiente forma:

$$C_{ri} = (a_{i1}, b_{i1}, c_{i1}, \ldots, a_{il_i}, b_{il_i}, c_{il_i}) \ ,$$
$$C_r = C_{r1} \ C_{r2} \ ... \ C_{rN}$$

En este proceso de ajuste se puede considerar cualquier otro tipo de función de pertenencia utilizando su representación parámetrica.

Durante la fase de reproducción del AG, se utilizan el operador de *mutación no uniforme* de Michalewicz [Mic96], descrito en el Capítulo 1, y el operador de *cruce max-min aritmético* [HLV95] que emplea herramientas difusas para mejorar el comportamiento de AG. Para dos cromosomas C_v y C_w, este último operador genera cuatro descendientes de la siguiente forma:

$$C_1 = aC_w + (1-a)C_v$$
$$C_2 = aC_v + (1-a)C_w$$
$$C_3 \text{ con } c_{3k} = \min\{c_k, c'_k\}$$
$$C_4 \text{ con } c_{4k} = \max\{c_k, c'_k\}$$

Este operador utiliza un parámetro a que puede tener un valor constante o dependiente de la edad de la población. La descendencia resultante estará formada por los dos mejores individuos de los cuatro hijos generados.

La población inicial se genera utilizando la información de la BD original del SCBRD a ajustar. De hecho, codificamos los parámetros de la BD del SCBRD inicial en un cromosoma, al que denominaremos C_1. Para generar los individuos restantes, se asocia un intervalo de rendimiento $[c_h^l, c_h^r]$ a cada gen c_h de C_1, $h = 1 \ldots \sum_{i=1}^{N} l_i \cdot 3$, que será el intervalo de ajuste para el gen correspondiente, es decir, $c_h \in [c_h^l, c_h^r]$.

Para definir los intervalos de ajuste de cada gen, y así generar posteriormente de forma

aleatoria los distintos individuos, se sigue el siguiente proceso: Si (t mod 3) = 1 entonces el gen c_t representa el valor de la parte izquierda del conjunto soporte de un número difuso. El número difuso se define con los tres parámetros (c_t, c_{t+1}, c_{t+2}) y los intervalos de rendimiento son los siguientes:

$$
\begin{aligned}
c_t &\in [c_t^l, c_t^r] = [c_t - \tfrac{c_{t+1}-c_t}{2}, c_t + \tfrac{c_{t+1}-c_t}{2}], \\
c_{t+1} &\in [c_{t+1}^l, c_{t+1}^r] = [c_{t+1} - \tfrac{c_{t+1}-c_t}{2}, c_{t+1} + \tfrac{c_{t+2}-c_{t+1}}{2}], \\
c_{t+2} &\in [c_{t+2}^l, c_{t+2}^r] = [c_{t+2} - \tfrac{c_{t+2}-c_{t+1}}{2}, c_{t+2} + \tfrac{c_{t+2}-c_{t+1}}{2}]
\end{aligned}
$$

En la figura 3.10 se pueden ver estos intervalos.

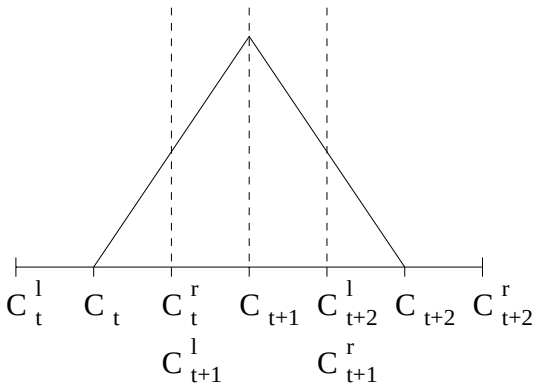

Figura 3.10. Intervalos de ajuste

Con este proceso de generación de la población inicial se crea una población de cromosomas que contiene a C_1 como primer individuo y los restantes se generan aleatoriamente con cada gen definido dentro su intervalo de ajuste correspondiente.

La función de adaptación es la misma utilizada en el proceso de multiselección descrito en la sección anterior: si la BC es completa a un grado τ, el valor de adaptación será igual al porcentaje de error cometido en la clasificación. En otro caso, será igual a 1. En este aspecto, debemos recordar que los modificadores lingüísticos seleccionados en la etapa de multiselección están involucrados en este cálculo de la función de adaptación.

El esquema de codificación y el algoritmo de ajuste que lo incluye no consideran variaciones en el nivel de granularidad establecido para cada variable, en contraste con el método genético propuesto por Ishibuchi y Murata en [IM97]. Esto se debe al hecho de que el objetivo del proceso genético de ajuste propuesto es la optimización de una partición difusa definida a priori para mejorar el rendimiento de un SCBRD para el que se ha obtenido previamente la BR. Esta BC se puede haber obtenido bien mediante un proceso de generación / simplificación como el propuesto o a través de un experto que normalmente no puede definir con precisión las funciones de pertenencia correspondientes a las etiquetas lingüísticas que él/ella utiliza habitualmente.

f	p	Ent.	Pr.	f	p	Ent.	Pr.	f	p	Ent.	Pr.
f_0	—	95.49	94.26	f_0	—	97.31	95.21	f_0	—	97.31	95.21
f_1	—	98.58	**95.80**	f_1	—	98.57	**96.22**	f_1 P	—	97.68	**95.73**
f_1 P	—	97.47	94.36	f_2 P	—	97.14	95.80	f_2 P	—	97.50	**95.73**
f_3	20	95.30	94.26	f_3	20	97.51	95.30	f_3	20	97.51	95.21
f_8	0,0.3,10	93.99	94.26	f_7	0,0.8	97.33	95.37	f_7 P	0,0.5	97.51	**95.73**
f_8	0,0.3,20	93.99	94.26	f_8	0,0.3,10	96.78	95.21	f_7 P	0,0.8	96.69	**95.73**
f_8	0,0.8,20	94.90	94.26	f_8	0,0.8,20	97.51	95.21	f_7	0,0.5	97.49	95.31
BR tipo (a) (61.8 reglas)				BR tipo (b) (70 reglas)				BR tipo (c) (64.4 reglas)			

Tabla 3.2. Resultados del proceso de generación para Iris con distintos MRDs y tipos de reglas difusas

3.6 Experimentación y Análisis de Resultados

Hemos utilizado los conjuntos de ejemplos Iris y Pima, descritos en Capítulo 2, para mostrar el comportamiento del proceso de aprendizaje evolutivo que presentamos. El método de estimación de error empleado es el remuestreo aleatorio y los resultados presentados son medias de los obtenidos para las cinco particiones de entrenamiento y prueba.

La experimentación realizada en el Capítulo 2, con distintos MRDs, tres conjuntos de ejemplos y seis métodos de generación de reglas difusas distintos, mostró que, en la mayoría de los casos, se obtenían los mejores resultados con los SCBRDs formados por BRs tipo (b). No obstante, no se puede señalar una superioridad general de esta estructura de representación del conocimiento, ya que éste es un aspecto determinado por las características del problema y por el proceso de aprendizaje inductivo utilizado. Por este motivo, hemos ejecutado la primera etapa del método, la etapa de generación, para los dos problemas con los tres tipos de reglas y distintos MRDs (los MRDs con los parámetros para los que se sabe que se obtiene mejor comportamiento). Los resultados obtenidos, junto el conocimiento previo de las propiedades de los distintos MRDs, nos permitirán determinar el mejor tipo de regla y MRD, para los problemas considerados y el proceso de aprendizaje propuesto. Los resultados completos de esta experimentación preliminar se muestran en el Apéndice 1 (tablas 3.15, 3.16, 3.17 y 3.18). Las tablas 3.2 y 3.3 son un resumen de los mejores resultados.

Como se puede observar, para nuestro proceso de generación de reglas difusas y los problemas considerados, el tipo de reglas que permite una mejor representación del conocimiento extraído es el (b), por lo que las etapas de postprocesamiento se realizarán sobre BRs tipo (b).

Además, hay que señalar en este punto la superioridad de los MRDs alternativos con BRs generadas por un séptimo método de generación, el proceso iterativo de generación

f	p	Ent.	Pr.	f	p	Ent.	Pr.	f	p	Ent.	Pr.
f_0	—	70.42	66.95	f_0	—	74.87	73.67	f_0	—	70.30	72.21
f_1	—	73.69	**71.68**	f_1 P	—	76.16	**74.49**	f_3 P	10	70.34	**72.52**
f_1 P	—	74.91	70.88	f_1	—	76.75	73.96	f_3	20	70.34	**72.52**
f_2 P	—	66.42	63.01	f_4	0.7	74.04	73.87	f_3	10	70.09	72.42
f_7	0,0.5	68.08	65.81	f_8	0,0.8,20	74.66	73.46	f_8	0,0.3,10	70.34	72.41
f_7	0,0.8	68.15	65.20	f_7	0,0.3	72.71	72.13	f_8	0,0.8,20	70.37	72.31
f_7	0,0.3	67.32	63.34	f_7 P	0,0.3	72.71	72.13	f_8	0,0.8,10	70.13	72.31
BR tipo (a) (153.4 reglas)				BR tipo (b) (279.6 reglas)				BR tipo (c) (187.8 reglas)			

Tabla 3.3. Resultados del proceso de generación para Pima con distintos MRDs y tipos de reglas difusas

incluido en el método de aprendizaje multietapa. De nuevo, y sin intervenir en el proceso de generación de reglas, la incorporación de un MRD que integre la información de las reglas disparadas, incrementa el poder de predicción del SCBRD resultante.

El resto de la experimentación se organizará de la siguiente forma:

- Se ejecutarán las etapas de multiselección y ajuste para Iris y Pima con BRs tipo (b). En la tabla 3.4 se muestran los parámetros utilizados en las tres etapas del proceso de aprendizaje.

- Seleccionaremos, a partir de la información proporcionada por el proceso de generación (tablas 3.15, 3.16, 3.17 y 3.18) y el conocimiento previo de los operadores, los mejores MRDs para realizar con ellos la experimentación de las etapas de postprocesamiento, y obtener distintos SCBRDs formados por

  - una BR simplificada,
  - un conjunto de modificadores lingüísticos, y
  - un conjunto de parámetros de las funciones de pertenencia ajustadas

  que cooperen entre sí y con el MRD.

- Como mencionamos en la descripción del proceso de multiselección, éste puede realizarse desde dos puntos de vista, buscando el mejor conjunto de modificadores comunes para todas las reglas difusas de la BR o determinando los mejores modificadores para cada regla. Para analizar la orientación introducida por el uso de modificadores, ejecutaremos la etapa de postprocesamiento con la misma BR y MRD, con aprendizaje de modificadores tipo I y II y sin modificadores. Estos resultados se muestran en las tablas con la siguiente estructura:

Generación	
N. etiquetas / variable	5 (Iris) y 3 (Pima)
ω (grado cobertura ejs. positivos)	0.05
k (k-consistencia)	0.1
ϵ (grado máximo de cobertura)	1.5
Multiselección	
Número de generaciones	500
Número de individuos	61
τ (τ-completitud)	0.1
% de reglas para el radio del nicho	0.025
β (factor de potencia)	0.5
Número de soluciones	3
Probabilidad de cruce	0.6
Probabilidad de mutación	0.1
Ajuste	
Número de generaciones	500
Número de individuos	61
τ	0.1
a (cruce max-min aritmético)	0.35
b (mutación no uniforme)	5
Probabilidad de cruce	0.6
Probabilidad de mutación	0.1

Tabla 3.4. Parámetros del método genético de aprendizaje

  - Las filas notadas con MS_i (con $i \in \{1,2,3\}$) corresponden a los resultados que se obtienen con la BC seleccionada en la iteración i del proceso de multiselección.
  - Las filas notadas con A_i (con $i \in \{1,2,3\}$) muestran los resultados de clasificación correcta por la BC obtenida tras el proceso de ajuste de la BC seleccionada en la iteración i.

- Compararemos los resultados con los obtenidos por otros algoritmos de aprendizaje basados en estructuras de representación del conocimiento distintas: árboles de decisión con C4.5 [Qui93], redes neuronales con LVQ [Koh97], reglas inductivas con CN2 [CN86] y reglas difusas obtenidas con los seis métodos de generación descritos en el Capítulo 1.

3.6.1 Iris

Sin Modif.			
	Entr.	Pr.	NR
MS_1	98.57	94.72	40.6
MS_2	98.57	93.61	39
MS_3	98.57	95.21	43.8
A_1	99.11	93.28	40.6
A_2	99.11	93.28	39
A_3	99.46	94.78	43.8

Modif. I			
	Ent.	Pr.	NR
MS_1	99.29	94.36	45.4
MS_2	99.48	93.67	44.8
MS_3	99.65	93.18	44.8
A_1	99.47	93.44	45.4
A_2	99.48	93.34	44.8
A_3	99.65	93.28	44.8

Modif. II			
	Entr.	Pr.	NR
MS_1	99.64	94.48	47.4
MS_2	99.82	91.11	46.2
MS_3	100	94.03	51.2
A_1	99.64	95.66	47.4
A_2	99.82	91.60	46.2
A_3	100	94.26	51.2

Tabla 3.5. Resultados para Iris con BR tipo (b) y MRD Clásico (f_0)

Sin Modif.			
	Entr.	Pr.	NR
MS_1	99.47	94.62	45.6
MS_2	99.65	96.71	48
MS_3	99.65	95.21	47.4
A_1	99.47	94.26	45.6
A_2	99.65	96.22	48
A_3	99.65	95.21	47.7

Modif. I			
	Entr.	Pr.	NR
MS_1	99.83	93.28	49.8
MS_2	99.46	95.31	47.4
MS_3	99.65	94.29	48.6
A_1	99.83	92.85	49.8
A_2	99.46	95.31	47.4
A_3	99.65	93.84	48.6

Modif. II			
	Ent.	Pr.	NR
MS_1	99.83	94.74	53.2
MS_2	100	93.28	58.6
MS_3	100	94.16	60.8
A_1	99.83	93.73	53.2
A_2	100	94.36	58.6
A_3	100	94.26	60.8

Tabla 3.6. Resultados para Iris con BR tipo (b) y MRD basado en la Suma (f_1)

En las tablas 3.5, 3.6, 3.7, 3.8, 3.9 y 3.10 se muestran los resultados completos de las etapas multiselección y ajuste, con la intervención de los MRDs clásico, suma, media aritmética ponderada, media quasiaritmética, OWA y quasiOWA respectivamente.

Los resultados indican que los SCBRDs obtenidos para este conjunto de ejemplos, con el método evolutivo de aprendizaje multietapa propuesto en el que interviene el MRD, tienen un alto poder de generalización. Además el proceso es totalmente parametrizable, por lo que podemos diseñar el tipo de proceso de aprendizaje en función del tipo de SCBRDs que queramos: sin modificadores, con aprendizaje de modificadores para toda la BD, para cada regla, con particiones ajustadas o predefinidas.

Un análisis más detallado de los resultados de nuestro modelo nos permite hacer las siguientes observaciones:

- El uso de modificadores lingüísticos provoca un incremento de la cardinalidad de la BR que se obtiene tras el proceso de multiselección. El modificador lingüístico *muy*

Sin Modif.			
	Entr.	Pr.	NR
MS_1	99.47	94.72	43.2
MS_2	99.47	94.09	41
MS_3	99.64	94.78	40
A_1	99.65	94.36	43.2
A_2	99.83	95.76	41
A_3	99.82	93.87	40

Modif. I			
	Entr.	Pr.	NR
MS_1	99.65	94.84	43.6
MS_2	99.82	93.24	44.6
MS_3	99.82	93.73	43.6
A_1	99.65	94.23	43.6
A_2	99.82	92.92	44.6
A_3	99.82	93.83	43.6

Sin Modif. II			
	Entr.	Pr.	NR
MS_1	100	96.18	49.2
MS_2	100	93.95	44.6
MS_3	100	93.67	52.6
A_1	100	95.21	49.2
A_2	100	93.57	44.6
A_3	100	93.67	52.6

Tabla 3.7. Resultados para Iris BR Tipo (b) y MRD basado en la media aritmética (f_2) con ponderación g_2

Sin Modif.			
	Entr.	Pr.	NR
MS_1	98.76	93.71	38.4
MS_2	98.76	94.19	41
MS_3	98.76	94.68	43.8
A_1	99.29	92.37	38.4
A_2	99.29	93.71	41
A_3	99.29	94.27	43.8

Modif. I			
	Entr.	Pr.	NR
MS_1	99.29	94.68	44
MS_2	99.29	96.18	44.4
MS_3	99.46	93.73	49
A_1	99.29	94.24	44
A_2	99.82	92.92	44.4
A_3	99.82	93.83	49

Modif. II			
	Entr.	Pr.	NR
MS_1	99.65	94.34	52.8
MS_2	100	94.60	45.2
MS_3	99.65	96.18	48.2
A_1	99.65	94.84	52.8
A_2	100	93.57	45.2
A_3	100	93.67	48.2

Tabla 3.8. Resultados para Iris con BR tipo (b) y MRD basado en la media quasiaritmética $f_3,\ p = 20$

produce una reducción del grado de pertenencia del valor del atributo al conjunto difuso correspondiente, lo que disminuye el grado de compatibilidad de la regla con el ejemplo y hace que sea necesario incluir un número mayor de reglas en la BR para que ésta verifique la propiedad de completitud.

- El proceso de aprendizaje de modificadores lingüísticos asociados a cada regla (modificadores tipo II) suele dar como resultado un SCBRD con mayor rendimiento. En ocasiones se puede producir un sobreajuste de la BC a los ejemplos de entrenamiento, lo que puede conllevar una pérdida en la capacidad de generalización del sistema en los casos en que no se disponga de un conjunto de ejemplos de entrenamiento suficientemente representativo.

- El mecanismo de selección utilizado en las etapas de multiselección y ajuste puede provocar que, entre varios individuos con el mismo porcentaje de clasificación correcta en ejemplos de entrenamiento (y distinta en ejemplos de prueba), se seleccione el que tiene un peor comportamiento sobre ejemplos de prueba. Esto es debido a

Sin Modif.			
	Entr.	Pr.	NR
MS_1	99.47	93.28	41
MS_2	99.47	94.36	43.8
MS_3	99.64	93.24	41.8
A_1	100	93.87	41
A_2	99.82	95.33	43.8
A_3	100	92.27	41.8

Modif. I			
	Ent.	Pr.	NR
MS_1	99.83	94.84	44.8
MS_2	99.82	94.36	43.8
MS_3	99.82	93.51	42.6
A_1	100	95.76	44.8
A_2	99.82	93.77	43.8
A_3	99.82	92.65	42.6

Modif. II			
	Entr.	Pr.	NR
MS_1	99.83	94.19	50.8
MS_2	100	95.73	51.6
MS_3	100	94.22	53.6
A_1	99.83	94.68	50.8
A_2	100	95.76	51.6
A_3	100	95.29	53.6

Tabla 3.9. Resultados para Iris con BR tipo (b) y MRD basado en el operador OWA f_7, $(0, 0.8)$

Sin Modif.			
	Entr.	Pr.	NR
MS_1	98.76	93.51	43.8
MS_2	98.76	94.61	39.8
MS_3	98.76	93.61	40.6
A_1	99.11	94.26	43.8
A_2	99.29	94.29	39.8
A_3	99.11	92.69	40.6

Modif. I			
	Ent.	Pr.	NR
MS_1	99.48	94.26	45
MS_2	99.29	92.75	45.8
MS_3	99.29	92.69	43.8
A_1	99.48	94.78	45
A_2	99.64	93.87	45.8
A_3	99.29	92.92	43.8

Modif. II			
	Entr.	Pr.	NR
MS_1	100	95.27	50.8
MS_2	99.65	96.28	50.4
MS_3	99.83	94.64	48.2
A_1	100	94.26	50.8
A_2	99.65	95.76	50.4
A_3	99.83	93.73	48.2

Tabla 3.10. Resultados para Iris con BR tipo (b) y MRD basado en el operador quasiOWA f_8, $(0, 0.8, 20)$

que la selección de individuos con igual valor de la función de adaptación se hace en función de su posición ordenada en la población.

- Se puede observar que, en algunos casos, se obtienen los mejores resultados con los SCBRDs obtenidos antes de la etapa de ajuste. Como se ha mencionado anteriormente, el proceso de generación propuesto con este tipo de reglas extrae el conocimiento de los ejemplos de una forma bastante precisa, por lo que el ajuste de los parámetros de las funciones de pertenencia de una BR tipo (b) en este conjunto de ejemplos no tiene un efecto significativo.

En la tabla 3.11 se muestran los mejores resultados de la etapa de postprocesamiento. En esta tabla la columna indicada con E-M indica si el resultado indicado se ha obtenido por una BC resultado de la etapa de multiselección (con MS) o de la etapa de ajuste (con A), sin utilización de modificadores lingüísticos (indicado con 0), con modificadores tipo I (indicado con I) o con modificadores tipo II (notado con II). Por ejemplo, los resultados que tengan el valor MS-II en la columna E-M corresponden a una BC generada en la

f	p	Ent.	Pr.	E-M
f_0	—	99.64	95.66	A-II
f_1	—	99.65	96.71	MS-0
f_2 P	—	100	96.18	MS-II
f_3	20	99.29	96.18	MS-I
f_8	0,0.8,20	99.65	96.28	MS-II
f_7	0,0.8	100	95.76	MS-I

BR tipo (b)

Tabla 3.11. Resumen de resultados de los procesos de posprocesamiento para IRIS con distintos MRDs y modificadores

etapa de multiselección con aprendizaje de modificadores tipo II.

Algoritmo	Prueba
C4.5	92.7
CN2	94.16
LVQ	95.72
Método 1 (Wang y Mendel)	95.23
Método 2 (Hong y Lee)	94.78
Método 3 (Ishibuchi y otros, iterativo)	95.76
Método 4 (Ishibuchi y otros, genético)	88.54
Método 5 (Yuan y Zhuang)	90.26
Método 6 (Pal y otros)	81.17

Tabla 3.12. Resultados obtenidos con otros procesos de aprendizaje para Iris

Los resultados obtenidos por el SCBRD generado por el proceso genético de aprendizaje se comparan con los conseguidos con las mismas particiones entrenamiento/prueba por los Sistemas de Clasificación diseñados a partir los otros algoritmos: C4.5, CN2 y LVQ. Además,se incluyen los mejores resultados alcanzados con SCBRDs obtenidos mediante los procesos de aprendizaje descritos en el Capítulo 1. Los porcentajes de clasificación correcta de SCBRDs mostrados en la tabla 3.12 y en el gráfico 3.11 son los alcanzados con los MRDs con mejor comportamiento.

Los resultados indican que los SCBRDs obtenidos para esta base de ejemplos con nuestro método de aprendizaje genético en cooperación con los MRDs alternativos al clásico, tienen una mayor capacidad de generalización, es decir, un porcentaje mayor de éxito en la clasificación de ejemplos de prueba, que los Sistemas de Clasificación obtenidos por los otros algoritmos de aprendizaje analizados.

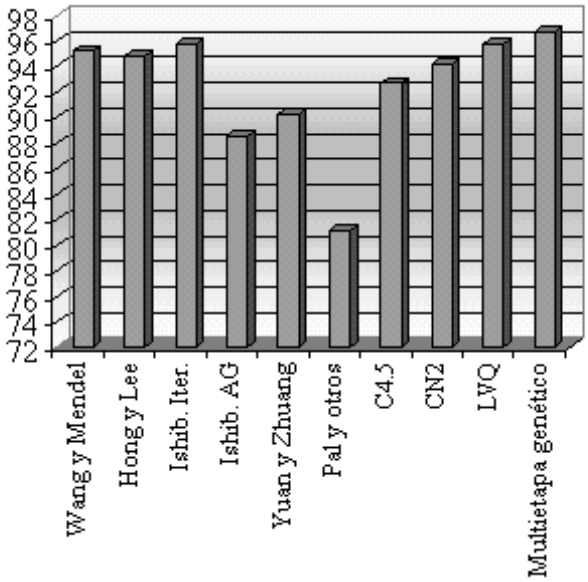

Figura 3.11. Resultados para Iris con distintos métodos de aprendizaje

3.6.2 Pima

En la tabla 3.13 se muestran los mejores resultados obtenidos con el proceso genético de aprendizaje con distintos MRDs.

		Generación			Posprocesamiento			
f	p	Ent.	Pr.	NR	Ent.	Pr.	NR	E-M
f_0	—	74.87	73.67	279	81.81	74.06	147.4	A-0
f_1 P	—	76.16	74.49	279	83.27	75.81	197.6	MS-II
f_4	0.7	74.04	73.87	279	80.87	75.68	148.4	MS-I
f_3 P	20	74.84	73.82	279	78.14	74.17	154.8	MS-0
f_7	0,0.3	72.71	72.13	297	84.21	76.20	187.6	A-II
f_7 P	0,0.3	72.71	72.13	279	83.86	75.82	177	A-II

BR tipo (b)

Tabla 3.13. Resultados obtenidos para Pima con el método genético de aprendizaje

Para el conjunto de ejemplos Pima, los mejores resultados se obtienen tras la etapa de ajuste, con modificadores tipo II y el MRD basado en el operador OWA. Como se observó en la experimentación realizada en el Capítulo 2, las características de este problema hacen que la mayor parte de los procesos de aprendizaje inductivo generen una BR con un elevado número de reglas, por lo que un MRD que selecciona la información a agregar

Algoritmo	Pr.
C4.5	71.4
CN2	74.5
LVQ	67.71
Método 1 (Wang y Mendel)	73.95
Método 2 (Hong y Lee)	65.23
Método 3 (Ish. y otros, iterativo)	74.77
Método 4 (Ish. y otros, genético)	69.74
Método 5 (Yuan y Zhuang)	73.42
Método 6 (Pal y otros)	63.88

Tabla 3.14. Resultados para Pima con los Sistemas de Clasificación obtenidos con otros métodos de aprendizaje

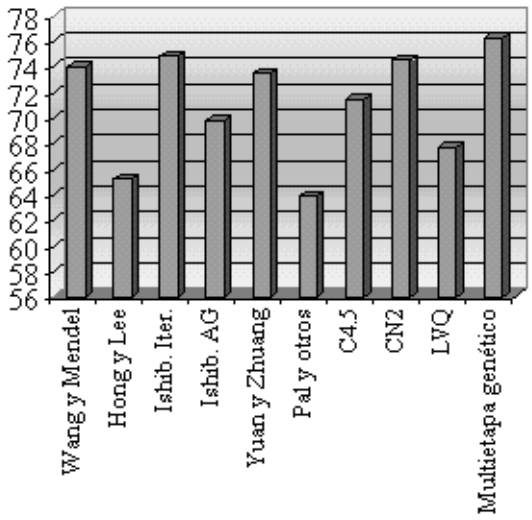

Figura 3.12. Resultados para Pima con distintos métodos de aprendizaje

proporciona el mejor resultado al SCBRD resultante. Además, la integración de este MRD alternativo al clásico en el proceso de aprendizaje facilita la simplificación de la BR, permitiendo la obtención de una BR con menos reglas y mejor porcentaje de clasificación correcta sobre ejemplos de prueba.

De nuevo, la comparación con los resultados obtenidos por el resto de los algoritmos (mostrados en la tabla 3.14 y en la figura 3.12) nos permite observar que el SCBRD generado a partir del proceso genético multietapa tiene un mayor poder de clasificación correcta sobre ejemplos de prueba que los Sistemas de Clasificación obtenidos con C4.5, CN2 y LVQ, y que los SCBRDs generados mediante los métodos de aprendizaje del 1 al 6.

3.7 Comentarios Finales

El proceso de aprendizaje genético multietapa obtiene SCBRDs con carácter descriptivo que alcanzan porcentajes de clasificación correcta sobre ejemplos de prueba superiores a todos los procesos de aprendizaje considerados.

La integración de los MRDs en el proceso de aprendizaje permite la obtención de BRs con buen nivel de cooperación entre las reglas que las componen y con el MRD utilizado, lo que incrementa las posibilidades del proceso de razonamiento aproximado que se realiza en cada clasificación de un nuevo ejemplo. Además, el proceso de aprendizaje determina el mejor conjunto de modificadores lingüísticos para los términos de las variables para mejorar la precisión del sistema mientras se mantiene el carácter descriptivo del mismo. Incluye tambien una etapa de ajuste que, sin disminuir la interpretabilidad del SCBRD, modifica la localización y el ancho de los conjuntos difusos utilizados para incrementar la exactitud del SCBRD.

Los parámetros del SCBRD se determinan de una forma eficiente a través de una pequeña experimentación en la etapa no evolutiva de generación de reglas.

Resumiendo lo expuesto, el método evolutivo multietapa propuesto incluye:

- Un proceso de generación de BRs difusas que representan de forma precisa la información aportada por los ejemplos de entrenamiento y que verifican las propiedades de completitud y consistencia.
- Un proceso genético de multiselección que permite obtener distintas definiciones del SCBRD
  - con una BR compacta,
  - formadas por reglas con alto grado de cooperación entre ellas,
  - con un alto grado de cooperación con el MRD utilizado por el SCBRD, y
  - con un conjunto de modificadores que aporten mayor descripción y precisión.
- Un proceso de ajuste que permite mejorar la definición inicial de las particiones difusas de las variables lingüísticas.

Este método cubre los aspectos de generación, simplificación y ajuste propios del diseño de un SCBRD. El proceso de Selección de Características definido en el Capítulo 4 permite una simplificación del SCBRD previa a su diseño y completa la resolución de todos los aspectos de diseño de un SCBRD.

3.8 Apéndice: Resultados de Clasificación de SCBRDs Obtenidos en la Etapa de Generación con Distintos Métodos de Razonamiento Difusos

f	p	Ent.	Pr.	Ent.	Pr.	Ent.	Pr.	Ent.	Pr.
		g_1		g_2		g_1		g_2	
f_0	—	95.49	94.26	—	—	97.31	95.20	—	—
f_1	—	98.58	95.80	97.47	94.36	98.57	96.22	98.22	95.80
f_2	—	92.16	91.78	94.20	93.28	97.49	95.37	97.14	95.80
f_3	10	94.94	93.28	94.59	94.26	97.33	95.21	97.51	95.21
f_3	20	95.30	94.26	94.59	94.26	97.51	95.21	94.59	94.26
f_5	0.3	93.83	93.28	94.57	93.28	97.50	95.27	97.33	95.21
f_5	0.5	94.56	93.28	94.75	93.28	97.50	95.21	97.32	95.21
f_5	0.7	94.94	94.26	94.75	94.26	97.32	95.21	97.51	95.21
f_6	10	94.76	94.26	94.94	94.26	97.51	95.21	97.31	95.21
f_6	20	95.29	94.26	95.12	94.26	97.31	95.21	97.31	95.21
f_7	0,0.3	93.81	93.28	94.17	93.28	96.77	95.21	96.59	95.21
f_7	0,0.5	93.44	92.37	93.96	92.85	96.77	94.36	96.42	97.78
f_7	0,0.8	92.69	91.78	94.37	93.28	97.33	95.37	96.97	95.27
f_8	0,0.3,10	93.99	94.26	—	—	96.78	95.21	—	—
f_8	0,0.5,10	94.15	93.83	—	—	96.78	95.21	—	—
f_8	0,0.8,10	94.72	93.28	—	—	97.33	95.21	—	—
f_8	0,0.3,20	93.99	94.26	—	—	96.96	95.21	—	—
f_8	0,0.5,20	94.16	93.83	—	—	96.96	95.21	—	—
f_8	0,0.8,20	94.90	94.26	—	—	97.51	95.21	—	—
		BR tipo (a) (61.8 reglas)				BR tipo (b) (70 reglas)			

Tabla 3.15. Resultados para Iris. Proceso de generación (1)

f	p	Ent.	Pr.	Ent.	Pr.
		g_1		g_2	
f_0	—	97.31	95.21	—	—
f_1	—	96.75	94.09	97.68	95.73
f_2	—	96.57	93.67	97.49	95.73
f_3	10	97.51	95.21	97.51	95.21
f_3	20	97.51	95.21	97.51	95.21
f_5	0.3	97.49	95.73	97.69	95.21
f_5	0.5	97.86	95.21	97.51	95.21
f_5	0.7	97.51	95.21	97.51	95.21
f_6	10	97.51	95.21	97.31	95.21
f_6	20	97.31	95.21	97.31	95.21
f_7	0,0.3	97.69	93.93	97.69	94.36
f_7	0,0.5	97.49	95.31	97.51	95.73
f_7	0,0.8	97.12	93.24	97.69	95.73
f_8	0,0.3,10	97.86	94.36	—	—
f_8	0,0.5,10	97.69	95.21	—	—
f_8	0,0.8,10	97.51	95.21	—	—
f_8	0,0.3,20	97.86	94.36	—	—
f_8	0,0.5,20	97.51	95.21	—	—
f_8	0,0.8,20	97.51	95.21	—	—

BR tipo (c) (64.4 reglas)

Tabla 3.16. Resultados para Iris. Proceso de generación (2)

f	p	Ent.	Pr.	Ent.	Pr.	Ent.	Pr.	Ent.	Pr.
		g_1		g_2		g_1		g_2	
f_0	—	70.42	66.94	—	—	74.87	73.67	—	—
f_1	—	73.69	71.68	74.91	70.88	76.75	73.96	76.16	74.49
f_2	—	66.42	63.02	65.26	58.60	71.18	70.06	71.98	70.78
f_3	10	62.13	56.12	61.22	55.11	74.28	72.94	74.63	73.56
f_3	20	61.71	55.62	61.39	55.63	74.63	73.56	74.77	73.56
f_5	0.3	67.78	63.22	65.51	58.70	73.06	71.91	73.65	72.95
f_5	0.5	67.84	62.81	65.43	58.80	73.51	73.04	74.07	73.56
f_5	0.7	68.02	62.71	65.40	58.80	74.04	73.87	74.59	73.46
f_6	10	54.83	49.12	54.41	47.37	75.11	73.56	75.01	73.67
f_6	20	54.34	47.16	60.79	54.62	75.01	73.67	74.84	73.77
f_7	0,0.3	67.32	63.34	66.03	60.47	72.71	72.13	73.24	72.13
f_7	0,0.5	68.08	65.81	67.66	61.80	72.02	71.09	72.75	71.72
f_7	0,0.8	68.15	65.20	67.56	61.38	71.46	70.46	72.54	71.19
f_8	0,0.3,10	63.48	59.13	—	—	74.11	73.06	—	—
f_8	0,0.5,10	64.04	59.33	—	—	74.21	73.16	—	—
f_8	0,0.8,10	63.87	58.30	—	—	74.32	72.94	—	—
f_8	0,0.3,20	62.82	58.73	—	—	74.49	73.06	—	—
f_8	0,0.5,20	63.55	58.73	—	—	74.60	73.16	—	—
f_8	0,0.8,20	63.27	58.00	—	—	74.66	73.46	—	—
		BR tipo (a) (153.4 reglas)				BR tipo (b) (279.6 reglas)			

Tabla 3.17. Resultados para Pima. Proceso de generación (1)

f	p	Ent.	Pr.	Ent.	Pr.
		g_1		g_2	
f_0	—	70.30	72.21	—	—
f_1	—	66.85	68.20	68.35	69.84
f_2		66.68	68.10	68.25	69.84
f_3	10	70.09	72.41	70.34	72.52
f_3	20	70.34	72.52	70.34	72.21
f_5	0.3	68.84	70.26	69.57	71.18
f_5	0.5	69.46	70.97	70.06	71.48
f_5	0.7	70.02	71.27	70.23	72.00
f_6	10	70.61	72.31	70.51	72.21
f_6	20	70.51	72.21	70.48	72.21
f_7	0,0.3	68.87	70.15	69.11	70.77
f_7	0,0.5	67.86	68.91	69.15	69.95
f_7	0,0.8	67.03	68.61	68.42	69.74
f_8	0,0.3,10	70.34	72.41	—	—
f_8	0,0.5,10	70.30	72.21	—	—
f_8	0,0.8,10	70.13	72.31	—	—
f_8	0,0.3,20	70.48	72.31	—	—
f_8	0,0.5,20	70.48	72.21	—	—
f_8	0,0.8,20	70.37	72.31	—	—

BR tipo (c) (187.8 reglas)

Tabla 3.18. Resultados para Pima. Proceso de generación (2)

Capítulo 4

Selección de Características en Problemas de Clasificación

El avance de las tecnologías de la computación y de su uso ha hecho posible la creación de grandes bases de datos y demanda técnicas de aprendizaje automatico adecuadas para su tratamiento. Cualquier agente de aprendizaje tiene que aprender de la experiencia y discriminar entre partes relevantes e irrelevantes de la misma. Éste es un problema universal de todos los agentes inteligentes, con mayor importancia en situaciones en las que la cantidad de información disponible es mayor [KJ97].

El diseño de un Sistema de Clasificación a través de un algoritmo de aprendizaje supervisado parte de conjuntos de instancias "valores de características - clase" y determina una función que generaliza el conocimiento presente en los datos y permite clasificar nuevas instancias.

En cualquier proceso de aprendizaje inductivo, la presencia de características irrelevantes o redundantes puede tener efectos negativos. Cuanto mayor sea el número de características consideradas, mayor será el número de instancias necesarias para asegurar la variabilidad estadística entre patrones de distintas clases. Además, las características irrelevantes o redundantes pueden llevar a la obtención de un Sistema de Clasificación sobreajustado a los datos de entrenamiento o erróneo. La presencia de variables irrelevantes o redundantes tendrá también como consecuencia que el Sistema de Clasificación obtenido sea, en general, más complejo y menos preciso que uno aprendido a partir de datos relevantes.

En problemas reales, se desconocen las características relevantes antes del proceso de obtención de datos, y con frecuencia se introducen muchas variables candidatas para representar mejor el dominio del problema, lo que lleva a considerar características parcial o completamente irrelevantes para el concepto a describir.

Por todas estas razones, este problema debe ser resuelto mediante un *proceso de Selección de Características* (SC) que determine las características relevantes en nuestra

aplicación para alcanzar un Sistema de Clasificación con máximo rendimiento y mínimo esfuerzo de medida.

En este capítulo analizaremos el proceso de SC aplicado a problemas de clasificación aunque, como se ha comentado, el problema es común a cualquier proceso de descripción de conocimiento en cualquiera de sus etapas. Los algoritmos de SC seleccionan las características importantes para una aplicación concreta, eliminan las variables irrelevantes o redundantes, mejoran la calidad de los datos, hacen posible que los algoritmos de extracción de conocimiento trabajen más rápido en conjuntos de datos grandes y mejoran la comprensibilidad de los resultados obtenidos.

El capítulo se organizará como se expone a continuación. En la primera sección analizaremos algunas de las propuestas realizadas hasta el momento en la literatura especializada bajo el enfoque evolutivo. Dedicaremos la siguiente sección a describir nuestro algoritmo de SC, su funcionamiento y componentes. El objetivo de esta propuesta es la determinación de un subconjunto de variables optimal para el diseño de un SCBRD con mayor precisión. Puesto que el proceso de SC desarrollado es independiente del modelo de representación del conocimiento y del proceso de aprendizaje inductivo utilizado para diseñar el Sistema de Clasificación final, en la sección tercera mostraremos los resultados obtenidos en una experimentación en dos etapas:

- En primer lugar describiremos los resultados conseguidos empleando la regla del vecino más cercano con los conjuntos de variables seleccionados por nuestras propuestas, comparándolos con los obtenidos por otros algoritmos de SC.
- A continuación desarrollaremos el proceso de diseño de un Sistema de Clasificación con dos procesos de aprendizaje inductivo distintos (nuestro método, descrito en el Capítulo 3, y C4.5 [Qui93]) a partir de los mejores subconjuntos obtenidos, mostrando así el comportamiento de las variables seleccionadas en procesos inductivos con distinta heurística y orientación.

4.1 Algoritmos Genéticos de Selección de Características

La SC es un problema de optimización con restricciones que se ha abordado en distintas propuestas mediante AGs [SS89, BBM92, PGP+93, LWZ95, BDeJH+97, Lan97, RPG+97, MBV98, YH98].

Los AGs son técnicas de búsqueda robustas que han mostrado, de forma teórica y empírica, un buen comportamiento en espacios de búsqueda complejos, por lo que constituyen

una buena alternativa para una búsqueda heurística eficaz en el espacio de subconjuntos de variables.

Se han desarrollado *algoritmos genéticos de SC* tanto desde el enfoque filtro como envolvente. Sus diferencias residen en la definición de los componentes principales del AG:

- El *esquema de codificación* utilizado depende del objetivo de la SC. Así, si el proceso de SC determina el mejor conjunto de variables con una cardinalidad prefijada, es adecuado el uso de cromosomas con codificación entera en los que el i-ésimo gen representa la i-ésima variable seleccionada [LWZ95]. Por el contrario, si el proceso busca el conjunto mínimo de variables que maximiza una función de adaptación dada, se utilizan cromosomas de longitud igual al número total de variables con codificación binaria, en la que un 1 en el i-ésimo gen indica la presencia de la i-ésima característica y un 0 su ausencia. Este esquema se ha utilizado en distintas propuestas de AGs de SC [SS89, BBM92, PGP+93, Lan97, MBV98, YH98].

  En los procesos de *extracción de características* se busca una transformación, normalmente lineal, de las variables que ayude a eliminar información redundante o irrelevante, y muestre la importancia relativa de las variables seleccionadas. En [JCC98] se desarrolla un proceso de extracción de características que utiliza un algoritmo evolutivo GA-P (método híbrido entre los AGs y la Programación Genética).

  Los procesos de extracción de características engloban a los de SC en el caso en el que se asocie un peso a cada variable igual a 0 ó 1, pero no son considerados en esta memoria ya que ni simplifican la estructura del Sistema de Clasificación, ni reducen el costo de obtención de datos, ni el tiempo de aprendizaje, restando interpretación lingüística a los resultados. Se han propuesto distintos procesos genéticos de extracción de características en los que la codificación utilizada es real, de forma que el valor real del gen i representa el peso asociado a la variable i-ésima [PGP+93]. En [RPG+97] se describen otros esquemas de codificación para este problema.

- La *función de adaptación* depende del enfoque del algoritmo de SC. De esta forma, se puede distinguir entre las funciones de adaptación incluidas en procesos filtro o envolventes de SC.

  1. **Algoritmos evolutivos de SC tipo filtro**. Bajo este enfoque Liu y otros han propuesto una función de evaluación basada en la medida de información mutua entre variables [LWZ95] en el algoritmo de SC descrito brevemente en la siguiente sección. Lanzi utiliza en [Lan97] una función de adaptación basada en el ratio de inconsistencia introducido por la eliminación de variables. Martín-Bautista y Vila proponen en [MBV98] una medida de similitud entre el cromosoma (que representa un conjunto de variables) y un vector de discriminación resultado de la comparación entre ejemplos.

2. **Algoritmos evolutivos de SC envolventes**. La estimación de la precisión proporcionada por la regla del vecino más cercano forma parte de distintas propuestas, fundamentalmente por la rapidez de su cálculo. Brill y otros proponen en [BBM92] un AG de SC orientado por una combinación lineal normalizada de la precisión obtenida con las variables representadas por el cromosoma y de la desviación del individuo respecto a la media. Punch y otros en [PGP+93] utilizan como función de adaptación una combinación lineal del error cometido por el K-NN [PF70] y el ratio de vecinos que clasifican incorrectamente, con el objetivo de conseguir que, para un conjunto dado de variables, la mayoría de los vecinos clasifiquen correctamente. Ray y otros en [RPG+97] utilizan también el ratio de error proporcionado por el K-NN, combinandolo con el número de características utilizadas en la clasificación y con un factor que corrige la orientación introducida por la existencia de distintas proporciones de ejemplos de las distintas clases.

   También se han utilizado otros modelos de aprendizaje distintos al vecino más cercano. En [YH98], Yang y Honavar proponen un AG de SC que optimiza el ratio de clasificación correcta obtenido por una red neuronal.

   Bala y otros en [BDeJH+97] utilizan una función de adaptación que combina tres medidas independientes: una medida de entropía sobre las variables, una estimación del costo de extracción de las variables seleccionadas y el ratio de clasificación obtenido por un árbol de decisión construido mediante C4.5. De esta forma, obtienen un algoritmo evolutivo de SC híbrido filtro-envolvente.

   Habitualmente, el componente del AG que consume el mayor porcentaje del tiempo de cómputo es el cálculo de la función de evaluación. Este hecho se acentúa en algorimos envolventes en los que el cálculo de la función de evaluación conlleva el diseño del Sistema de Clasificación según algún modelo de inducción. Este es el motivo por el que la mayoría de los procesos genéticos envolventes utilizan como parte de su función de adaptación la medida proporcionada por el 1-NN o el K-NN. Además, se han propuesto distintas soluciones para optimizar la evaluación como la técnica del *muestreo de ejemplos de entrenamiento* desarrollada por Brill y otros en [BBM92], en la que cada evaluación se utiliza sólo un subconjunto de ejemplos de entrenamiento generado de forma aleatoria en cada generación. Lin y otros proponen un AG con procesamiento paralelo que distribuye la evaluación individual en distintos nodos [LPG94].

En las siguientes subsecciones describimos brevemente tres procesos de SC basados en AGs que hemos utilizado en esta memoria para su comparación con los de las propuestas realizadas.

4.1.1 AG1: Algoritmo Genético Filtro de Selección de Características de Liu y otros

Liu desarrolla en [LWZ95] un AG generacional filtro de SC. El AG viene definido por sus componentes:

- *Representación entera de los cromosomas.* Cada individuo tiene longitud H (siendo H el número de características deseadas) y en él el gen i representa la i-ésima variable seleccionada.
- *Función de evaluación.* Para determinar la bondad de los subconjuntos de variables, utiliza una combinación lineal de la medida de información entre las variables seleccionadas, y entre cada una de las varibles y la variable de clase.

$$f(S) = \sum_{s \in S} I(C, s) - \beta \cdot \sum_{u,v \in S,\ u \neq v} I(u, v)$$

  siendo β un parámetro real $\beta \in [1, 2]$.
- Operador de *cruce parcialmente complementario* (que se describirá en la siguiente sección).
- Operador de *mutación de inserción - eliminación aleatoria*, que determina un conjunto de indices del cromosoma padre y los sustituye por otros no seleccionados.

4.1.2 AG2: Algoritmo Genético Envolvente de Selección de Características

Para comparar los resultados con los obtenidos por las propuestas genéticas realizadas, se ha extendido el método genético filtro de Liu y otros a un AG envolvente que utiliza la medida de precisión aportada por la regla del vecino más cercano como medida de evaluación de los cromosomas.

El resto de los componentes del AG son iguales a los descritos en la sección anterior (AG1).

4.1.3 AG3: Algoritmo Genético Envolvente de Selección de Características

Hemos desarrollado un AG de estado estacionario con codificación binaria para la selección del *mínimo* número de características que maximicen una función de adaptación.

En el proceso evoluciona una población de cromosomas de longitud fija igual al número total de variables, codificados en binario de forma que un 0 en el gen i indica que la variable i no forma parte del subconjunto representado por el cromosoma y un 1 indica que es seleccionada. De la población así codificada se seleccionan los dos mejores individuos como cromosomas padre que se cruzan con un operador de cruce simple en un punto y se mutan mediante el operador de mutación aleatoria simple.

Se han utilizado dos funciones fitness distintas:

- *Fitness 1.* Es una función que trata de obtener subconjuntos de variables que maximicen el poder de clasificación de la regla del vecino más cercano, por lo que su expresión es igual a la estimación del porcentaje de clasificación correcta obtenida mediante el proceso de 5-validación cruzada.
- *Fitness 2.* Además de buscar los individuos que maximicen la precisión del 1-NN, penaliza subconjuntos con un número de variables elevado para orientar la búsqueda hacia subconjuntos de características mínimos. Su expresión es:

$$\text{Fitness\_2} = \frac{\text{Clasificación\_Correcta}}{100} - u \cdot \frac{q}{\text{Num\_Total\_Variables}}$$

  donde q es la cardinalidad del conjunto de variables representado por el cromosoma y u representa el grado de importancia asignado a la obtención de subconjuntos de variables con cardinalidad mínima. Experimentalmente, hemos observado que un valor adecuado para el mismo es $u = 0.3$ y $u = 0.5$.

4.2 Algoritmos Genéticos de Estado Estacionario para la Selección de Características

En esta sección describiremos dos propuestas de algoritmos genéticos de SC. El carácter evolutivo de la mayor parte de las etapas del proceso de diseño del SCBRD propuesto en esta memoria nos lleva a la elección de un AG como método de búsqueda heurística incluido en el algoritmo de SC, orientado por la precisión proporcionada por un Sistema de Clasificación diseñado sobre las variables seleccionadas. En estas circunstancias, parecería adecuado el diseño de un algoritmo de SC tipo envolvente que incluya la construcción del SCBRD según el proceso descrito en el capítulo anterior, en el cálculo de la función de evaluación. No obstante, aunque en los problemas de clasificación abordados desde el aprendizaje automatico no se manejan Bases de Datos de cardinalidad tan elevada como en Minería de Datos (y por tanto es adecuado el uso de un algoritmo de SC envolvente), el coste del cálculo de la precisión del SCBRD es demasiado elevado y limitaría el uso

del algoritmo de SC a conjuntos de ejemplos con un tamaño limitado. Se pueden conseguir buenos resultados en el proceso de SC utilizando como estimación de la precisión alcanzable con las características seleccionadas, la medida proporcionada por una de las técnicas con mayor sensibilidad a variables irrelevantes, la regla del vecino más cercano [CH69]. Además, el proceso de aprendizaje del K-NN se limita al almacenamiento adecuado de los ejemplos de entrenamiento para el cálculo de distancias en la clasificación de nuevos ejemplos, por lo que es más eficiente que otros procesos de aprendizaje inductivo y representa un equilibrio adecuado entre precisión y eficiencia. Este es el motivo por el que proponemos algoritmos de SC envolventes basados en una búsqueda evolutiva guiada por la estimación proporcionada por el 1-NN.

Los AGs propuestos son dos variantes de un AG basado en la reproducción *estacionaria*. En las siguientes secciones analizaremos las diferencias de estos AGs *estacionarios* frente a los *generacionales* (descritos en el Capítulo 1) y explicaremos cada uno de sus componentes.

4.2.1 El Modelo de Reproducción Estacionario

En el AG clásico descrito por Holland [Hol75], denominado también AG *generacional*, en cada generación se construye una nueva población aplicando los operadores genéticos a la población actual. Los individuos de cada nueva generación pueden ser la descendencia de una operación de recombinación (por ejemplo, el cruce), el producto de una mutación, o simples copias -sin modificaciones- de la población actual (obtenidas a través de la operación de selección). De esta forma, la nueva población está formada complemente por la descendencia obtenida de la generación anterior (aunque muchos de los individuos sean copias exactas de sus antecesores).

En algunos esquemas de AGs generacionales, como en el esquema elitista implementado en las etapas de multiselección y ajuste definidas en el Capítulo 3, las generaciones sucesivas se van solapando en un cierto grado al mantener una proporción de la generación anterior en cada nueva generación. A la fracción de individuos nuevos en cada iteración se le denomina *salto generacional* [DeJ75].

El modelo de reproducción *estacionario* mantiene una población en la que sólo algunos de sus miembros se van modificando en sucesivas generaciones. El salto generacional es menor que en el modelo clásico ya que, en cada generación la única modificación de la población se produce cuando los descendientes obtenidos mediante las operaciones de recombinación y mutación sustituyen a los individuos peor adaptados.

Este tipo de reproducción se ha utilizado en el aprendizaje evolutivo de Sistemas Basados en Reglas, concretamente en los Sistemas Clasificadores [Hol86]. En [Sys89, Whi89, Sys91, DeJS93], se pueden encontrar estudios sobre su comportamiento.

4.2.2 Componentes del Algoritmo Genético Estacionario de Selección de Características

El algoritmo de SC propuesto realiza una búsqueda evolutiva guiada a través de una medida de la precisión alcanzable que le confiere carácter de algoritmo de SC envolvente. El proceso de SC está basado en un Algoritmo Genético Estacionario (AGE) con codificación entera. A continuación describimos sus componentes.

1. **Esquema de codificación.**

   El esquema de codificación propuesto genera cromosomas de longitud fija. Puesto que el objetivo de la SC propuesta es la obtención del subconjunto óptimo de características de un tamaño dado, la codificación entera nos permite representar en un cromosoma de longitud H, un subconjunto candidato de H variables, en el que el i-ésimo gen representa la i-ésima variable seleccionada.

   El AGE propuesto permite la incorporación de conocimiento disponible en la inicialización de la población. Así, es posible la introducción de subconjuntos de variables con buenas propiedades de separabilidad de clases o de precisión, obtenidos por otros algoritmos de SC o a través de conocimiento experto. El resto de la población (o la población al completo si no se dispone de información previa) se genera de forma aleatoria.

2. **Función de evaluación.**

   La función que orienta la búsqueda es una estimación de la precisión alcanzable con las variables codificadas en el cromosoma mediante el uso de la regla del vecino más cercano. Esta estimación se hace mediante el método de *h-validación* cruzada de la forma propuesta por Kohavi [KJ97] y descrito en la figura 4.1 para un valor $h = 5$. El *conjunto de entrenamiento* se divide en h bloques de igual tamaño combinados de h formas distintas en conjuntos de entrenamiento y prueba. Para cada una de esas particiones entrenamiento-prueba, se calcula el porcentaje de acierto sobre el conjunto de prueba correspondiente, y el valor estimado de precisión será la media aritmética de los h porcentajes de acierto, de acuerdo a la siguiente expresión general:

   $$F(C_i) = \sum_{j=1}^{h} \frac{Precision(Particion_j)}{h}$$

   donde h representa la cardinalidad de la validación cruzada empleada (h-validación cruzada). El valor del parámetro h vendrá determinado por el tamaño del conjunto de ejemplos con el que se trabaje. En nuestra experimentación trabajamos con $h = 5$, por representar este valor un equilibrio entre precisión de la estimación y eficiencia del proceso de validación.

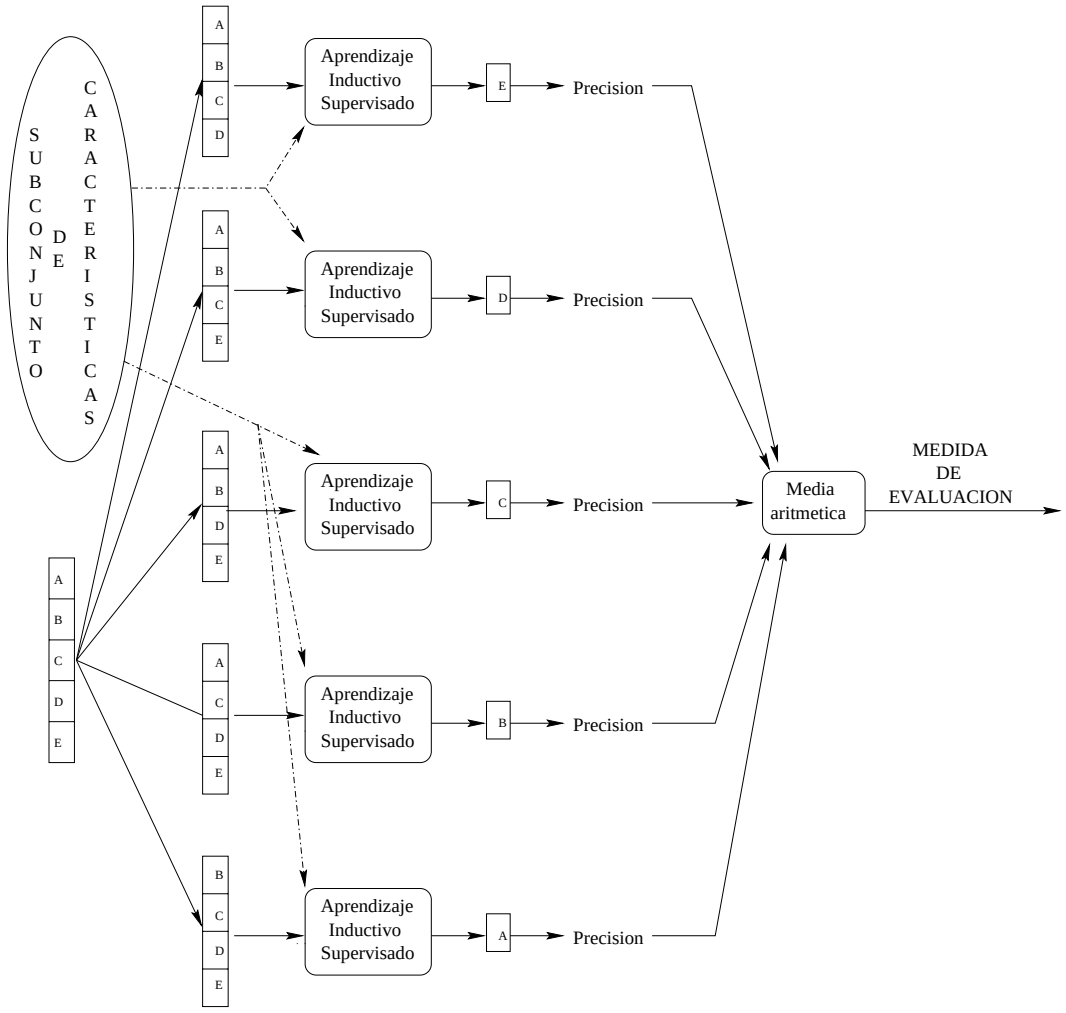

Figura 4.1. Método de estimación de la precisión (5-validación cruzada)

Como se puede observar, este método nos permite estimar la capacidad de generalización sin utilizar el conjunto de prueba empleado para validar el conjunto de variables obtenido.

3. **Esquema de reproducción.**

   El AGE utiliza el esquema de reproducción estacionario que no reemplaza todos los individuos de la población en cada generación sino un determinado número de ellos. El número de nuevos cromosomas a crear es un parámetro de la reproducción estacionaria, y toma el valor 1 ó 2 en la mayor parte de los esquemas de reproducción que siguen esta filosofía. En este punto, hay que destacar que el esquema de reemplazamiento generacional es un caso especial del esquema de reproducción estacionario, en el que el número de individuos a crear (mediante recombinación o selección) es igual al tamaño de la población.

   En nuestra propuesta hemos seguido el siguiente esquema:

- Generamos una población intermedia mediante asignación de probabilidades basada en ordenación lineal y el esquema de selección de muestreo estocástico universal de Baker.
- Aplicamos los operadores de cruce y mutación a algunos individuos de esta población intermedia. El número de cromosomas a crear vendrá determinado, de esta forma, por la probabilidad de cruce y mutación.
- Los nuevos cromosomas creados sustituirán a los cromosomas peor adaptados *de la población original.*

De esta forma, se sigue la filosofía de la reproducción estacionaria, que aplica los operadores de cruce y mutación a los mejores individuos, ya que se utilizan los operadores de variación para un porcentaje de los cromosomas de una población intermedia que selecciona los individuos mejor adaptados según un esquema de ordenación lineal y muestreo estocástico universal. La generación de más de dos cromosomas nuevos introduce más diversidad en la nueva población manteniendo las características del esquema de reproducción estacionario puesto que la nueva población sólo se diferencia de la anterior en estos cromosomas generados que sustituyen a los peor adaptados.

4. **Operador de cruce.**

   El operador de cruce utilizado es el *cruce parcialmente complementario* [LWZ95] que explota el espacio de búsqueda refinando las soluciones obtenidas de la siguiente manera: dados dos cromosomas de la población $P(t)$, $C_v^t = (c_1, \ldots, c_M)$ y $C_w^t = (c'_1, \ldots, c'_M)$, se generan dos hijos

   $$H_1 = (d_1, \ldots, d_k, h_{k+1}, \ldots, h_M)$$
   $$H_2 = (d_1, \ldots, d_k, h'_{k+1}, \ldots, h'_M)$$

   donde $d_1, \ldots, d_k$ son los genes comunes a los dos cromosomas seleccionados para ser cruzados y $h_{k+1}, \ldots, h_M$, y $h'_{k+1}, \ldots, h'_M$ son genes seleccionados aleatoriamente entre los no comunes. De esta forma, los hijos generados mantienen las variables comunes a sus padres y combinan de forma aleatoria el resto de la información contenida en ellos. Son individuos válidos sin variables repetidas, por lo que no es necesario ningún de proceso de reparación de cromosomas.

5. **Operador de mutación.**

   El operador de *mutación aleatoria simple* modifica arbitrariamente uno o más genes de un individuo eliminando la variable correspondiente y sustituyéndola por otra no presente en el cromosoma. Así se introduce diversidad dentro de la población.

Un análisis del AG con reproducción estacionaria nos permite observar que, en determinadas ocasiones, se puede evolucionar hacia una población sin suficiente diversidad. Una solución a este problema viene dada por el uso de un operador de cruce que, además

de explotar la información contenida en los padres, nos permita introducir un cierto nivel de diversidad. Para ello proponemos el operador de *cruce en dos puntos con reparación* que presenta este comportamiento de explotación-exploración. Su funcionamiento es el siguiente: para la obtención de dos descendientes, por cada pareja de cromosomas seleccionados se determinan aleatoriamente dos puntos de cruce y se intercambian los genes entre esos dos puntos. Este proceso puede generar individuos no válidos con variables repetidas a los que se aplica un proceso de reparación que sustituye cada gen repetido por una variable no seleccionada.

Este operador mantiene el caracter de herencia y refinamiento propio de los operadores de cruce incorporando –en los casos en los que los descendientes tengan variables repetidas– la propiedad de exploración muy adecuada en el proceso evolutivo propuesto.

Al AGE con el *cruce parcialmente complementario*, lo denominaremos en esta memoria AGE I y a la propuesta con los mismos componentes salvo el operador de recombinación que será el *cruce multipunto con reparación*, AGE II.

4.3 Experimentación

Para mostrar el comportamiento de los algoritmos de SC propuestos hemos considerado la base de datos Sonar, formada por 208 instancias de un problema de clasificación de objetivos de sonar. Cada uno de estos ejemplos se describe a través de 60 variables con la finalidad de discriminar entre la salida de un sonar correspondiente a un metal cilíndrico o a una roca aproximadamente cilíndrica. Este conjunto de datos se ha utilizado en [Bat94], en un proceso de SC greedy de tipo filtro, y en [GS88], en el entrenamiento de una red neuronal multicapa.

Los algoritmos AGE I y AGE II determinan conjuntos de variables optimales de un tamaño fijo. En esta experimentación se han buscado conjuntos de variables de tamaño 6, 12 y 15, lo que supone una reducción de la dimensionalidad de los datos del 90, 80 y 75 %, respectivamente.

Los conjuntos de variables obtenidos con los métodos de SC propuestos serán validados con la regla del vecino más cercano. Estos resultados se compararán con los conseguidos (también con el 1-NN) con las características seleccionadas por otros algoritmos de SC conocidos:

1. El *algoritmo probabilístico de SC tipo filtro de Liu y Setiono* (LVF) [LS96b], basado el algoritmo de búsqueda probabilística de Las Vegas [BB96] orientada por el ratio de inconsistencia [LM98] del subconjunto de características seleccionado. Determina el conjunto mínimo de características que no genera un grado de inconsistencia mayor que el preestablecido como parámetro de entrada del proceso.

Algoritmo	Objetivo	Tipo	Búsqueda
LVF (Liu)	Subcto. mínimo	Filtro	Aleatoria (Las Vegas)
MIFS (Battiti)	Subcto. tamaño fijo	Filtro	Secuencial hacia delante (Greedy)
AG 1 (Liu)	Subcto. tamaño fijo	Filtro	No determinística (AG)
AG 2	Subcto. tamaño fijo	Envolvente	No determinística (AG)
AG 3	Subcto. mínimo	Envolvente	No determinística (AG)

Tabla 4.1. Algoritmos de Selección de Características utilizados

2. El *algoritmo Greedy de SC tipo filtro de Battiti* (MIFS) [Bat94]. Es un proceso filtro de búsqueda secuencial hacia delante que selecciona las variables más informativas en función de la medida de información mutua [SW49].

3. El *AG generacional tipo filtro de Liu y otros*(AG1) [LWZ95],que utiliza como medida de evaluación la medida de información mutua.

4. Una *extensión de AG1 con la filosofía envolvente* (AG2). Es un AG con los mismos componentes que AG1 pero orientado por una estimación de la precisión alcanzable con la regla del vecino más cercano.

5. Dos *AGs estacionarios binarios con enfoque envolvente*, que determinan el conjunto *mínimo* de características más adecuado en función de dos medidas de evaluación distintas (a los que notamos como AG3a y AG3b, respectivamente). El primero se basa en la precisión proporcionada por la regla del vecino más cercano, incorporando la función de adaptación del segundo una penalización proporcional al número de características seleccionado. Al igual que el algoritmo LVF, no buscan un número fijo de variables.

Como se ha descrito en la sección anterior, los dos algoritmos de SC propuestos son procesos genéticos envolventes. Este es el motivo de la selección de tres algoritmos genéticos de SC para la comparación de resultados, de los cuales dos de ellos siguen el enfoque envolvente.

Por otra parte, el uso de un modelo de aprendizaje inductivo distinto al que posteriormente se utilizará para el diseño del SCBRD para el cálculo de la función de evaluación, hace adecuada la inclusión de algoritmos de SC tipo filtro en esta experimentación. De esta forma, se podrá observar el comportamiento de los conjuntos de variables obtenidos por algoritmos de SC orientados por una medida de separabilidad de clases independiente del modelo de aprendizaje y por aquellos que utilizan una medida de precisión con sesgo distinto al del modelo de aprendizaje final.

La tabla 4.1 muestra un resumen de los algoritmos seleccionados.

El conjunto de ejemplos Sonar contiene una partición entrenamiento-prueba utilizada por algunos autores para comprobar empíricamente el comportamiento de sus sistemas. En esta experimentación, se ha empleado esta división de los datos en 50 % entrenamiento y 50 % prueba.

La experimentación se ha organizado de la siguiente forma: Los métodos de SC tipo filtro trabajan con el conjunto de entrenamiento para calcular la medida de información mutua o de inconsistencia, según el caso. Posteriormente, se valida la solución aportada, mostrando el error de clasificación sobre ejemplos de prueba con la regla del vecino más cercano y éste es el valor mostrado en las tablas, mientras no se indique lo contrario. Para los métodos de SC envolventes, el proceso de selección se ha realizado con las 5 divisiones entrenamiento-prueba (obtenidas del conjunto de entrenamiento) y la fase de validación se ha llevado a cabo con el conjunto de prueba para proporcionar una estimación del error de clasificación también con el 1-NN.

4.3.1 Necesidad de Selección de Características en la Base de Ejemplos Sonar

Como hemos comentado al comienzo de esta sección, para el problema de clasificación del sonar se dispone de un conjunto de instancias descritas mediante 60 variables.

La tabla 4.2 muestra los resultados obtenidos en la etapa de generación descrita en el Capítulo 3 con 5 etiquetas lingüísticas para cada una de las 60 variables lingüísticas y distintos MRDs. Los parámetros utilizados en la ejecución son los siguientes: El proceso de generación utiliza reglas tipo (b), con $\omega = 0.05$, $k = 0.1$ y $\epsilon = 1.5$.

Hay que destacar que las características del proceso de diseño, junto con el modo de funcionamiento propio de los SCBRDs, hace que los SCBRDs resultantes sean especialmente sensibles a la presencia de variables irrelevantes.

Nuestro proceso de diseño no realiza una SC dentro del propio aprendizaje, por lo que utilizará todas las variables suministradas en las etapas de generación, multiselección y ajuste de la BC. El SCBRD resultante calculará el grado de aplicabilidad de una regla en función de una t-norma para considerar el grado de pertenencia de la instancia a los subconjuntos difusos que describen la semántica de las etiquetas lingüísticas presentes en el antecedente de la regla. En los experimentos realizados se ha utilizado la t-norma del mínimo, lo que hace que en el momento en que el valor de la instancia para una variable irrelevante no pertenezca al subconjunto difuso correspondiente, el grado de aplicabilidad de la regla sea cero.

El efecto de la presencia de variables irrelevantes se observa con claridad en la tabla 4.2 donde, independientemente del MRD utilizado, el porcentaje de clasificación obtenido es el mismo. Esto es debido a que los ejemplos mal clasificados, son ejemplos *no clasificados*, es decir, ejemplos para los que el SCBRD no tiene respuesta por no pertenecer a las zonas

GENERACIÓN		Sin pond.		Con pond.	
MRD	NR	Entr.	Pr.	Entr.	Pr.
Clásico (f_0)	311	100	43.27	—	—
Suma (f_1)		100	43.27	100	43.27
M. Arit. (f_2)		100	43.27	100	43.27
M. Quasiarit. (f_3, 10)		100	43.27	100	43.27
M. Quasiarit. (f_3, 20)		100	43.27	100	43.27
SOWA OR (f_5, 0.3)		100	43.27	100	43.27
SOWA OR (f_5, 0.5)		100	43.27	100	43.27
SOWA OR (f_5, 0.7)		100	43.27	100	43.27
SOWA OR (f_5, 0.8)		100	43.27	100	43.27
SOWA OR (f_5, 0.9)		100	43.27	100	43.27
Badd (f_6, 10)		100	43.27	100	43.27
Badd (f_6, 20)		100	43.27	100	43.27
OWA (f_7, 0, 0.3)		100	43.27	100	43.27
OWA (f_7, 0, 0.5)		100	43.27	100	43.27
OWA (f_7, 0, 0.8)		100	43.27	100	43.27
QuasiOWA (f_8, 0, 0.3, 10)		100	43.27	—	—
QuasiOWA (f_8, 0, 0.5, 10)		100	43.27	—	—
QuasiOWA (f_8, 0, 0.8, 10)		100	43.27	—	—
QuasiOWA (f_8, 0, 0.3, 20)		100	43.27	—	—
QuasiOWA (f_8, 0, 0.5, 20)		100	43.27	—	—
QuasiOWA (f_8, 0, 0.8, 20)		100	43.27	—	—

Tabla 4.2. Resultados después de la generación para Sonar con 60 variables y 5 etiquetas lingüísticas por variable

del espacio delimitadas por las reglas. Este efecto se podría atenuar con la utilización de otra t-norma con un comportamiento más compensativo, o considerando una partición difusa con menos etiquetas lingüísticas, como se muestra en la tabla 4.3.

En resumen, podemos afirmar que los resultados obtenidos con el SCBRD generado en la primera etapa de proceso de diseño con distintos MRDs para la base de ejemplos Sonar, muestran que la intervención de todas las variables presentes en el problema lleva al desarrollo de un SCBRD sobreajustado a los datos por la presencia de variables irrelevantes que introducen ruido. Este efecto es disminuido con la utilización de un número menor de etiquetas lingüísticas por variable, pero aún así es evidente el sobreaprendizaje y las limitaciones impuestas a las etapas de postprocesamiento para la mejora del SCBRD.

Este efecto es notable también con otros modelos de aprendizaje. En la tabla 4.4 se muestran los resultados obtenidos con la regla del vecino más cercano y C4.5.

Es necesario por tanto, un proceso de SC previo al proceso de aprendizaje inductivo,

GENERACIÓN		Sin pond.		Con pond.	
MRD	NR	Entr.	Pr.	Entr.	Pr.
Clásico (f_0)	331	99.04	75.00	—	—
Suma (f_1)		98.08	73.08	98.08	73.08
M. Arit. (f_2)		96.15	72.11	98.08	75.00
M. Quasiarit. (f_3, 10)		99.04	75.00	99.04	75.00
M. Quasiarit. (f_3, 20)		99.04	75.00	99.04	75.00
SOWA OR (f_5, 0.3)		98.08	**76.92**	99.04	75.96
SOWA OR (f_5, 0.5)		98.08	75.96	99.04	75.96
SOWA OR (f_5, 0.7)		99.04	75.96	99.04	75.00
SOWA OR (f_5, 0.8)		99.04	75.96	99.04	75.00
SOWA OR (f_5, 0.9)		99.04	75.00	99.04	75.00
Badd (f_6, 10)		99.04	75.00	99.04	75.00
Badd (f_6, 20)		99.04	75.00	99.04	75.00
OWA (f_7, 0, 0.3)		98.08	75.96	99.04	75.96
OWA (f_7, 0, 0.5)		98.08	74.04	98.08	75.00
OWA (f_7, 0, 0.8)		97.11	73.08	98.08	75.96
QuasiOWA (f_8, 0, 0.3, 10)		99.04	75.96	—	—
QuasiOWA (f_8, 0, 0.5, 10)		99.04	75.00	—	—
QuasiOWA (f_8, 0, 0.8, 10)		99.04	75.00	—	—
QuasiOWA (f_8, 0, 0.3, 20)		99.04	75.96	—	—
QuasiOWA (f_8, 0, 0.5, 20)		99.04	75.00	—	—
QuasiOWA (f_8, 0, 0.8, 20)		99.04	75.00	—	—

Tabla 4.3. Resultados después de la generación para Sonar con 60 variables y 3 etiquetas lingüísticas por variable

1-NN	C4.5
90.38	67.3

Tabla 4.4. Resultados con datos de prueba para Sonar con 60 variables y distintos modelos de aprendizaje inductivo

que facilite el proceso de inducción haciéndolo más eficiente y permitiendo la obtención de un SCBRD menos complejo y con mayor poder de generalización.

Algoritmo	Parámetro	Valor
AGE I y II	Long_Población	61
	Num_Generaciones	500
	P_c	0.4
	P_m	0.1
MIFS	β	0, 0.25, 0.5, 0.75, 1
LVF	Inconsistencia	0, 1
	Max_Iteraciones	77*N (4620)
AG1, AG2	Long_Población	61
	Num_Generaciones	500
	P_c	0.6
	P_m	0.1
	β	1, 1.25, 1.5, 1.75, 2
AG3	Long_Población	50
	Num_Generaciones	2000
	P_m	0.1
	u	0.3, 0.5

Tabla 4.5. Parámetros de ejecución de los algoritmos AGE I, AGE II, LVF, MIFS, AG1, AG2 y AG3

4.3.2 Resultados Selección de Características con Validación 1-NN

Hemos ejecutado los algoritmos AGE I, AGE II, MIFS, AG1, AG2 y AG3 para la obtención de conjuntos de variables de tamaño 6, 12 y 15 con los parámetros indicados en la tabla 4.5. Los algoritmos de SC con proceso de búsqueda evolutivo (AGE I, AGE II, AG1, AG2 y AG3) se han ejecutado con una de la población inicialmente formada por un 25 % de individuos que representan las mejores soluciones de los algoritmos LVF (si existían soluciones del tamanõ correspondiente) y MIFS.

En la tabla 4.6 se muestran los mejores resultados de la validación con el 1-NN de los conjuntos de variables obtenidos con los distintos algoritmos de SC. Los resultados completos correspondientes a las ejecuciones con los parámetros empleados se describen en el apéndice de este capítulo.

Hay que destacar que los resultados de LVF y AG3, por ser algoritmos de selección de conjuntos de características de longitud fija, sólo se muestran en el caso de que el conjunto obtenido sea de alguno de los tamaños prefijados. En la tabla 4.7 se describe un resumen de los mejores resultados obtenidos por ambos algoritmos independientemente del tamaño del conjunto de variables.

Como se puede observar, para los tres tamaños estudiados, los conjuntos de variables

	6 Var.	12 Var.	15 Var.
Algoritmo	Prueba	Prueba	Prueba
AGE I	**90.38**	93.27	**93.27**
AGE II	88.46	**95.19**	**93.27**
MIFS	69.23	69.23	69.23
LVF		88.46	91.35
AG 1	69.23	69.23	61.54
AG 2	78.85	93.27	92.31

Tabla 4.6. Resultados obtenidos con el 1-NN para conjuntos de variables seleccionadas con distintos Métodos de Selección de Características

Algoritmo	N. Var.	Prueba
AG3 a	34	94.23
AG3 b	19	92.31
LVF	18	94.23
LVF	15	91.35

Tabla 4.7. Resultados obtenidos con el 1-NN para conjuntos de variables seleccionadas con los Métodos de Selección de Características AG3 y LVF

obtenidos con AGE I y AGE II son los que dan mayor grado de generalización a la regla del vecino más cercano, con una mejora bastante significativa del valor obtenido con las 60 variables, y con una reducción de hasta el 90 % en la dimensionalidad de los datos.

Estos resultados no son mejorados por los algoritmos de selección de un número mínimo, y por tanto variable, de características (LVF y AG3), lo que muestra que la imposición de un tamaño preestablecido no limita la capacidad de obtención de un conjunto de variables optimal. Además, los conjuntos de variables obtenidos con LVF y AG3 tienen un número de características superior.

Es notable, como cabía esperar, la superioridad de las variables seleccionadas con algoritmos envolventes frente a los filtro, por el hecho de haber sido determinadas con el criterio de optimizar la función con la que posteriormente serán validadas. La idoneidad de los conjuntos de variables será valorada con distintos algoritmos de aprendizaje inductivo en la siguiente sección.

Si observamos las figuras 4.2, 4.3 y 4.4, que muestran las variables seleccionadas para distintas ejecuciones de los algoritmos, vemos que en los modelos filtro, la medida de separabilidad de clases utilizada impone una fuerte orientación al algoritmo de búsqueda,

condicionando en un alto grado el conjunto de variables seleccionadas. Esto es especialmente notable con la utilización de la medida de información mutua (algoritmos MIFS y AG1) donde, de forma independiente a los parámetros utilizados, determinadas variables son seleccionadas siempre y se pueden detectar fácilmente las zonas de solución. Esto limita la capacidad de búsqueda de los algoritmos y lleva a la obtención de resultados suboptimales como se verifica en el proceso de validación.

Este efecto no se observa en los modelos envolventes propuestos en esta memoria, en los que en ocasiones existen patrones fijos en las soluciones, pero éstos no condicionan en la misma medida la solución final. Además, el esquema evolutivo utiliza adecuadamente la información suministrada como población inicial, de los algoritmos LVF y MIFS, manteniendo en la herencia los patrones adecuados y eliminando los menos idóneos según la estimación del 1-NN.

4.3.3 Diseño del Sistema de Clasificacion a partir de las Variables Seleccionadas

El objetivo de los algoritmos de SC en esta memoria es la obtención de un conjunto de variables optimal para el diseño de un SCBRD con el proceso propuesto en el Capítulo 3. No obstante, los algoritmos AGE I y AGE II son independientes del modelo de inducción y aplicables como etapa previa a cualquier proceso de aprendizaje.

En esta sección veremos el comportamiento de las variables seleccionadas con AGE I y AGE II para el diseño del SCBRD. Finalmente, se mostrará su comportamiento sobre otro proceso de aprendizaje inductivo distinto, C4.5 [Qui93], para observar de forma empírica la validez de las variables seleccionadas en el aprendizaje del Sistema de Clasificación desde distintos modelos de inducción: 1-NN (mostrado en la validación en la sección anterior), nuestro proceso de diseño de SCBRDs y C4.5.

4.3.3.1 Resultados Obtenidos con un SCBRD

Los conjuntos de características mostrados en la sección anterior, se han seleccionado mediante el uso de un modelo de aprendizaje inductivo distinto a nuestra propuesta de diseño de SCBRD. Esto puede hacer que las variables obtenidas no sean las más adecuadas para el aprendizaje del SCBRD por incorporar el proceso de búsqueda un sesgo distinto al utilizado por el proceso de aprendizaje inductivo en la construcción del SCBRD final.

Una solución a este problema, ya utilizada por Brill y otros en [BBM92], implica la realización de una pequeña experimentación adicional con los conjuntos de variables obtenidos que nos permita determinar los más adecuados para nuestro proceso inductivo. En nuestro caso, esta etapa de prueba se limita a la primera y más eficiente de las fases del proceso de aprendizaje, la etapa de generación. De esta forma, la experimentación se

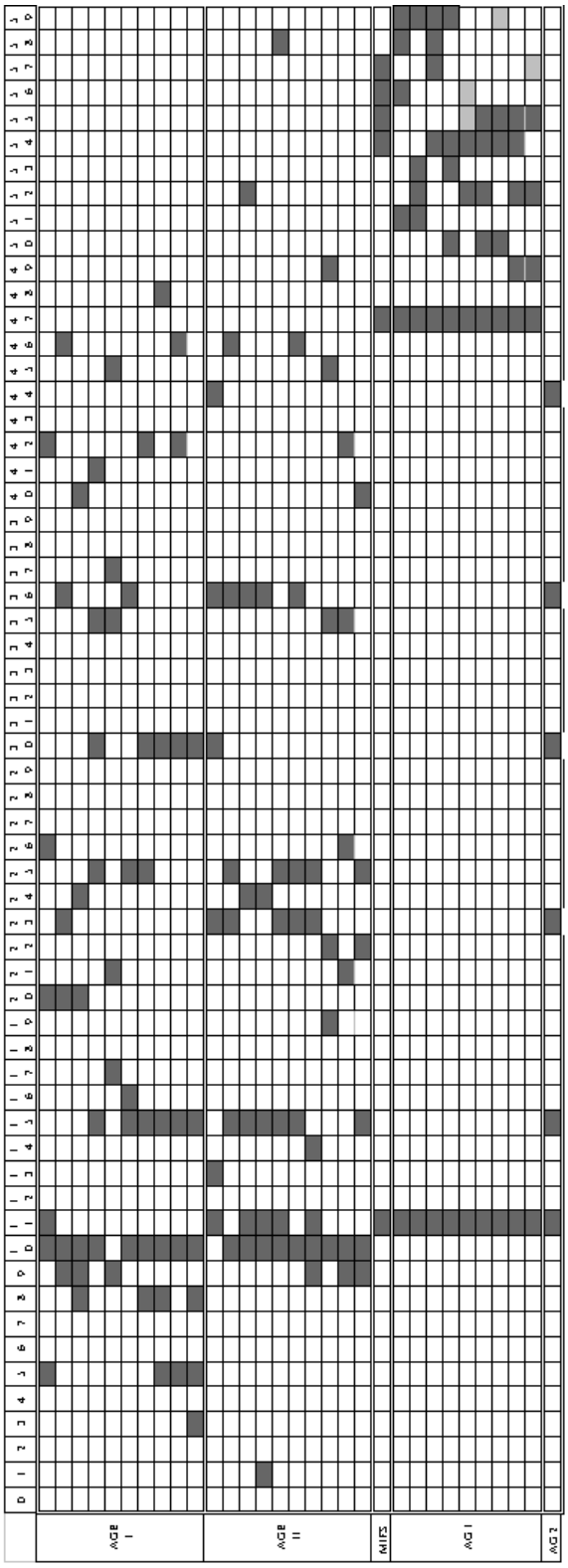

Figura 4.2. Conjuntos de 6 variables seleccionados

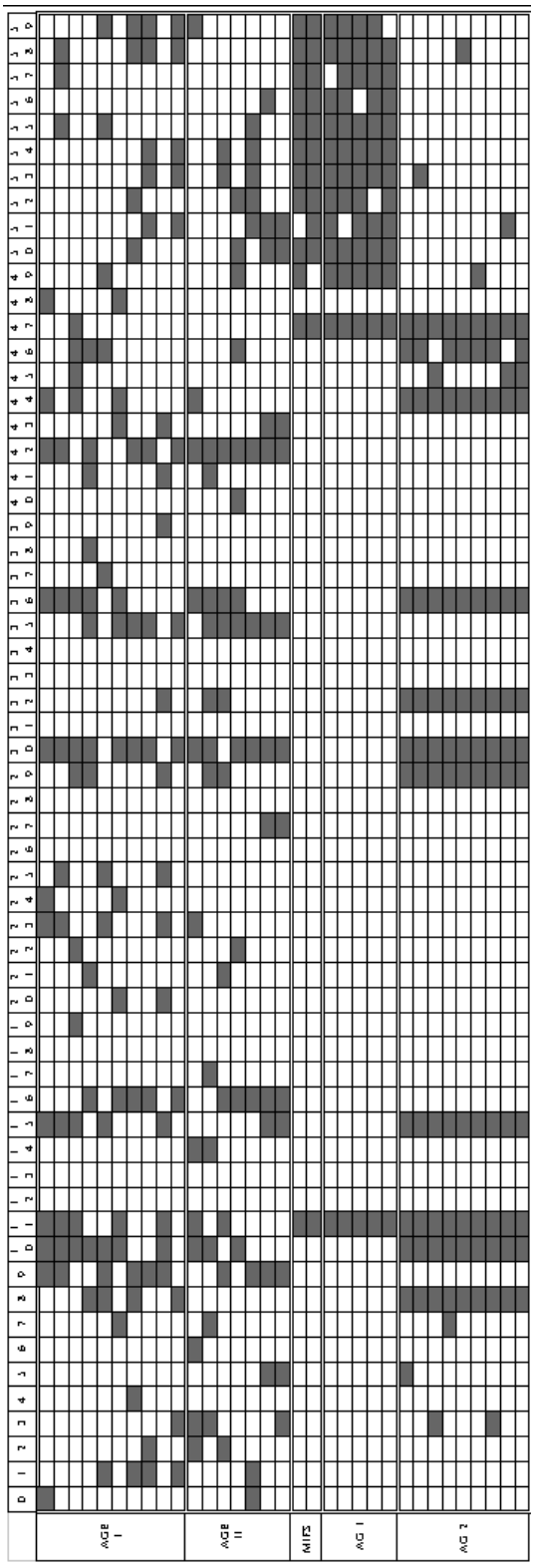

Figura 4.3. Conjuntos de 12 variables seleccionados

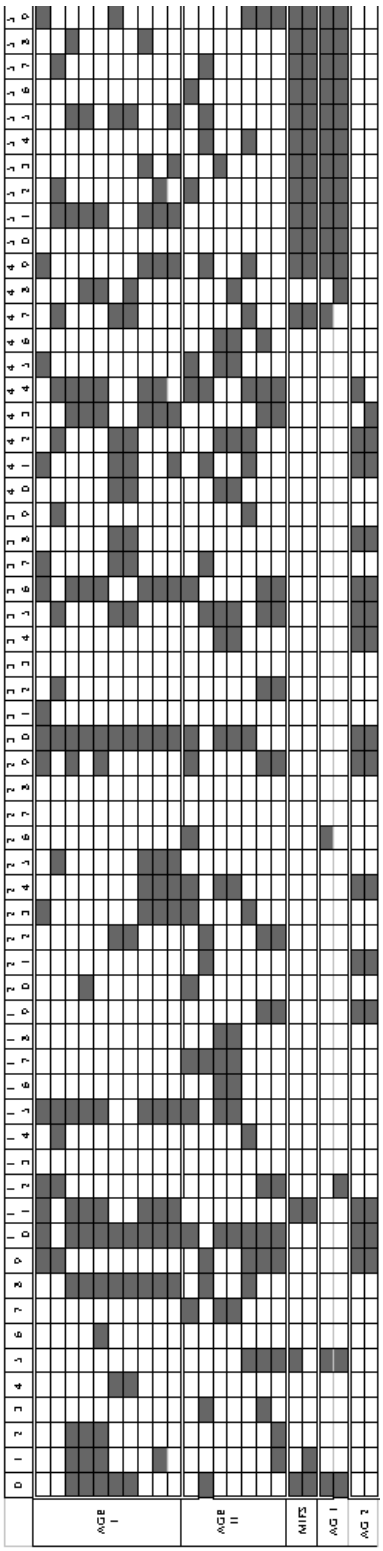

Figura 4.4. Conjuntos de 15 variables seleccionados

Algoritmo	MRD	NR	Entr.	Prueba
AGE I	OWA $(f_7, 0, 0.3)$	55	94.23	89.42
AGE II	M. Arit. P (f_2)	58	93.27	90.38
MIFS (Battiti)	Suma P.(f_1)	27	77.88	76.92
AG 1	Suma (f_1)	42	72.11	76.92
AG 2	SOWA OR P. $(f_5, 0.3)$	113	83.65	86.54

Tabla 4.8. Resultados obtenidos con un SCBRD construido sobre conjuntos de 6 variables seleccionados por distintos métodos de Selección de Características

realizará siguiendo los siguientes pasos:

1. Estudio previo de los subconjuntos seleccionados con la etapa de generación para determinar los dos conjuntos con mejor comportamiento para el diseño del SCBRD.
2. Para cada uno de estos conjuntos de variables, se seleccionan los dos MRDs que proporcionan mejores resultados.
3. Con esos dos conjuntos de variables y los mejores MRDs, se ejecutan las etapas de multiselección y ajuste.

Queremos mostrar la adecuación de las variables seleccionadas por nuestros métodos de SC para nuestro proceso de aprendizaje de SCBRDs, por lo que realizaremos la experimentación detallada anteriormente para los resultados de AGE I y AGE II con subconjuntos de 6, 12 y 15 variables. Además, para observar el rendimiento obtenido por nuestro proceso de aprendizaje a partir de variables seleccionadas con otros algoritmos de SC, desarrollaremos de igual forma SCBRDs para conjuntos de 6 variables con el resto de los algoritmos de SC utilizados en las comparaciones. Los resultados completos, así como las variables seleccionadas, se describen en el apéndice de este capítulo.

En la tabla 4.8 se muestran los resultados obtenidos con un SCBRD para conjuntos de 6 variables seleccionados con los distintos métodos de SC. Como se puede observar, los mejores resultados se alcanzan con los conjuntos generados por los métodos AGE I y AGE II, para los que se consigue un porcentaje de clasificación correcta en ejemplos de prueba del 90.38 con el 10 % de las variables. Tenemos que recordar en este punto que con el conjunto total de variables se conseguía un porcentaje máximo de 76.92. Además, si comparamos los resultados con los de la tabla 4.6, vemos que se mejoran los resultados obtenidos por la regla del vecino más cercano en todos los casos.

Las tablas 4.9 y 4.10 muestran los resultados obtenidos por un SCBRD inducido a partir de los conjuntos de 12 y 15 variables seleccionadas por los algoritmos AGE I y AGE II. Como se puede observar, se aumenta la distancia sobre los resultados conseguidos

Algoritmo	MRD	NR	Entr.	Prueba
AGE I	SOWA OR P. $(f_5, 0.5)$	183	92.31	94.23
AGE II	OWA $(f_7, 0, 0.5)$	45	91.35	90.38

Tabla 4.9. Resultados obtenidos con un SCBRD sobre conjuntos de 12 variables seleccionados con AGE I y II

Algoritmo	MRD	NR	Entr.	Prueba
AGE I	Clásico (f_0)	125	96.15	94.23
AGE II	OWA P. $(f_7, 0, 0.5)$	94	99.04	92.31

Tabla 4.10. Resultados obtenidos con un SCBRD sobre conjuntos de 15 variables seleccionados con AGE I y II

con 60 variables. No obstante, para estos tamaños de variables no se supera en todos los casos el porcentaje de acierto conseguido por la regla del vecino más cercano. Es la consecuencia del desarrollo de un algoritmo de SC envolvente que incorpora un modelo de inducción distinto al utilizado para el diseño del Sistema de Clasificación. Esto hace que las características obtenidas, optimales para el algoritmo de aprendizaje supervisado incluido en la función de evaluación, no lo sean para el proceso inductivo con el que se desarrolla el Sistema de Clasificación. No obstante, la SC realizada consigue, de forma eficiente, un conjunto de 15 variables con el que se obtiene un SCBRD con porcentaje de clasificación sobre datos de prueba de 94.23, lo que supone un incremento del poder de predicción de casi el 20 %, con una reducción de la dimensionalidad de los datos del 75 %.

4.3.3.2 Resultados Obtenidos con Árboles de Decisión Inducidos a partir de C4.5

En las tablas 4.11 y 4.12 se muestran los resultados obtenidos tras la misma experimentación, pero con el proceso de aprendizaje inductivo de árboles de decisión C4.5. Como se puede observar, los algoritmos de SC AGE I y AGE II permiten incrementar el poder de predicción de los árboles de decisión resultantes en más de un 16 %.

4.3.4 Análisis de los Resultados Obtenidos

En este capítulo se ha estudiado el problema de la SC y se ha propuesto un AGE de SC con dos variantes en función del operador de cruce utilizado. Se ha realizado una

Algoritmo	Entr.	Prueba
AGE I	94.2	76.0
AGE II	92.3	83.7
MIFS	96.2	71.2
AG 1	94.2	75.0
AG 2	94.2	67.3

Tabla 4.11. Resultados obtenidos por C4.5 sobre conjuntos de 6 variables seleccionados por distintos métodos de Selección de Características

	12 Variables		15 Variables	
Algoritmo	Entr.	Prueba	Entr.	Prueba
AGE I	95.2	79.8	97.1	83.7
AGE II	96.2	77.9	98.1	81.7

Tabla 4.12. Resultados obtenidos por C4.5 sobre conjuntos de 12 y 15 variables seleccionados por AGE I y II

experimentación con el conjunto de ejemplos Sonar en los siguientes pasos:

1. Ejecución de los algoritmos de SC
   (a) AGE I,
   (b) AGE II,
   (c) AG1,
   (d) AG2, y
   (e) MIFS,

   para la obtención de subconjuntos de 6, 12 y 15 variables.
2. Ejecución de los algoritmos de SC
   (f) LVF y
   (g) AG3

   para generar conjuntos de variables de cardinalidad mínima que optimicen una función de evaluación tipo filtro y envolvente, respectivamente.
3. Proceso de aprendizaje inductivo, mediante el método de aprendizaje supervisado descrito en el Capítulo 3, de SCBRDs

   (a) con los mejores conjuntos de 6 variables obtenidos con todos los métodos de SC descritos en los puntos 1 y 2,

   (b) con los mejores conjuntos con cardinalidad 12 y 15 generados con AGE I y II.

4. Desarrollo de la misma experimentación del punto 3 pero con el método de aprendizaje inductivo de árboles de inducción C4.5.

Esta experimentación nos permite extraer las líneas de comportamiento, sobre el conjunto de ejemplos y los algoritmos de aprendizaje inductivo utilizados, de los algoritmos de SC implementados:

- El proceso de validación a través del mismo algoritmo de aprendizaje inductivo utilizado como función de evaluación en los algoritmos de SC envolventes (1-NN), muestra que los conjuntos de variables seleccionados con AGE I y II tienen mayor capacidad de generalización sobre ejemplos de prueba que el conjunto total de variables.

- Los porcentajes de clasificación correcta sobre ejemplos de prueba con las variables seleccionadas con AGE I y II no son superados por los obtenidos con las variables determinadas por los algoritmos que buscan un conjunto mínimo con mejor comportamiento. Esto muestra que la capacidad de búsqueda de conjuntos optimales de los algoritmos AGE I y II no está limitada por la imposición inicial de una cardinalidad concreta en los conjuntos.

- Como era de esperar, los resultados obtenidos muestran un mejor comportamiento de los algoritmos de SC tipo envolvente frente a los filtro.

- Los algoritmos de SC tipo filtro, especialmente los basados en la medida de información mutua, MIFS y AG1, tienen tendencia a caer en un óptimo local por la fuerte orientación impuesta por la medida de separabilidad de clases utilizada, que les lleva a seleccionar las mismas variables independientemente del esquema de búsqueda utilizado. En este sentido, los algoritmos AGE I y II han mostrado un buen uso de la información proporcionada por este tipo de algoritmos como población inicial: las ejecuciones con población inicial resultado de los algoritmos MIFS y LVF han dado como resultado conjuntos de variables en los que se mantiene parte de la información proporcionada por las medidas de consistencia e información mutua según la estimación proporcionada por la regla del vecino más cercano.

- Los SCBRDs generados con los conjuntos de variables seleccionadas con AGE I y II tienen una capacidad alta de generalización utilizando menos de un 25 % de la información disponible. Esto indica que, para este problema y aún con el uso de un proceso de aprendizaje inductivo en los algoritmos de SC distinto al utilizado para la generación del SCBRD, los algoritmos de SC AGE I y II constituyen una buena etapa de preprocesamiento para la reducción de la dimensionalidad de los datos, adecuada para el diseño de un SCBRD más sencillo y con mayor precisión.

- También con el proceso de aprendizaje de árboles de decisión se observa una reducción importante del número de nodos, con un incremento de la precisión de la clasificación en ejemplos de prueba por encima del 16 % .

En definitiva, podemos afirmar que los algoritmos de SC AGE I y AGE II son una alternativa eficaz y eficiente para el problema de reducción de la dimensionalidad de los datos para el conjunto de ejemplos Sonar y el diseño del SCBRD. Los resultados aportados muestran que es adecuado su uso en el desarrollo de Sistemas de Clasificación a través de otros procesos de aprendizaje inductivo con distinta heurística y orientación.

4.4 Apéndice: Resultados Completos

	Variables						Prueba
AGE I	5	10	11	20	26	42	76.92
	9	10	20	23	36	46	80.77
	8	9	10	20	24	40	74.04
	10	15	25	30	35	41	88.46
	9	17	21	35	37	45	88.46
	10	15	16	25	35	36	81.73
	8	10	15	25	30	42	84.61
	5	8	10	15	30	48	79.81
	5	10	15	30	42	46	75.96
	3	5	8	10	15	30	79.81
AGE II	11	13	23	30	36	44	80.77
	10	15	23	25	36	46	81.73
	10	11	15	24	36	52	79.81
	1	10	11	15	24	36	80.77
	10	11	15	23	25	58	79.81
	10	15	23	25	36	46	81.73
	9	10	11	14	23	25	79.81
	10	19	22	35	45	49	90.38
	9	10	21	26	35	42	86.54
	9	10	15	22	25	40	82.69
MIFS	11	47	54	55	56	57	69.23
AG1	11	47	51	56	58	59	69.23
	11	47	51	52	53	59	69.23
	11	47	54	57	58	59	69.23
	11	47	50	53	54	59	69.23
	11	47	52	54	55	56	69.23
	11	47	50	52	54	55	69.23
	11	47	50	54	55	59	69.23
	11	47	49	52	54	55	69.23
	11	47	49	52	55	57	69.23
AG2	11	15	23	30	36	44	78.85

Tabla 4.13. Conjuntos de 6 variables y porcentaje de acierto sobre datos de prueba con 1-NN

	Variables												Prueba
AGE I	0	9	10	11	15	23	24	30	36	42	44	48	91.35
	9	10	11	15	23	25	30	40	42	55	57	58	92.31
	10	11	15	19	22	29	30	36	44	45	46	47	86.54
	8	10	16	21	29	30	35	36	38	41	42	46	92.31
	1	8	9	10	15	23	25	37	46	49	55	59	84.61
	7	10	11	16	20	24	30	35	36	43	44	48	95.19
	1	4	8	9	16	30	35	42	50	52	58	59	87.50
	1	2	9	16	30	35	42	51	53	54	58	59	84.61
	9	10	11	15	20	23	25	29	32	39	41	43	94.23
	1	3	8	16	30	35	42	51	53	54	58	59	86.54
AGE II	2	3	6	10	11	14	23	30	36	42	44	59	86.54
	4	7	10	14	17	29	30	32	35	36	41	42	92.31
	2	9	11	16	21	29	32	35	36	42	53	54	93.27
	10	16	22	30	35	36	40	42	46	49	50	52	92.31
	0	1	9	16	30	35	42	51	52	53	54	55	84.61
	5	9	15	16	27	30	35	42	43	50	51	56	90.38
	3	5	9	15	16	27	30	35	42	43	50	51	88.46
MIFS	11	47	49	50	52	53	54	55	56	57	58	59	69.23
	11	47	50	51	52	53	54	55	56	57	58	59	69.23
AG1	11	47	49	50	51	52	53	54	55	56	58	59	69.23
	11	47	49	50	52	53	54	55	56	57	58	59	69.23
	11	47	49	50	51	52	53	54	55	57	58	59	69.23
	11	47	49	50	51	53	54	55	56	57	58	59	69.23
	11	47	49	50	51	52	53	54	55	56	57	58	69.23
AG2	5	8	10	11	15	29	30	32	36	44	46	47	93.27
	8	10	11	15	29	30	32	36	44	46	47	53	90.38
	3	8	10	11	15	29	30	32	36	44	45	47	93.27
	7	8	10	11	15	29	30	32	36	44	46	47	92.31
	8	10	11	15	29	30	32	36	44	46	47	58	90.38
	8	10	11	15	29	30	32	36	44	46	47	49	90.38
	3	8	10	11	15	29	30	32	36	44	46	47	89.42
	8	10	11	15	29	30	32	36	44	45	47	51	93.27
	8	10	11	15	29	30	32	36	44	45	46	47	89.42

Tabla 4.14. Conjuntos de 12 variables y porcentaje de acierto sobre datos de prueba con 1-NN

	Variables															Prueba
AGE I	9	10	11	12	15	23	29	30	31	36	37	41	45	49	59	91.35
	9	12	14	15	25	30	32	35	39	42	44	47	51	52	57	93.27
	0	1	2	8	10	11	15	29	30	36	43	44	51	55	58	85.58
	0	1	2	8	10	11	15	20	30	36	43	44	48	51	55	86.54
	0	1	2	6	8	10	11	15	29	30	36	43	44	48	51	87.50
	0	4	8	10	22	30	35	37	38	40	41	42	47	55	59	90.38
	0	4	8	10	22	30	35	37	38	40	41	42	47	48	55	91.35
	8	10	11	15	23	24	25	30	36	43	44	49	51	53	58	90.38
	1	8	10	11	15	23	24	25	30	36	43	44	49	51	52	88.46
	8	10	11	15	23	24	25	30	36	41	43	49	51	53	55	92.31
AGE II	7	10	15	17	20	23	24	26	29	36	30	44	45	52	56	93.27
	0	3	8	9	17	21	22	35	37	41	44	49	54	55	57	90.38
	7	10	15	16	17	18	24	30	34	35	40	42	45	46	53	89.42
	7	10	15	16	17	18	24	30	34	35	40	42	45	46	48	90.38
	5	8	9	10	14	23	30	39	41	42	44	47	49	54	59	86.54
	3	5	9	10	12	19	22	29	32	35	36	43	44	46	59	90.38
	1	2	5	9	10	12	19	22	29	32	35	36	43	44	59	89.42
MIFS	0	5	11	47	49	50	51	52	53	54	55	56	57	58	59	69.23
	0	1	11	47	49	50	51	52	53	54	55	56	57	58	59	57.69
AG1	0	5	26	47	49	50	51	52	53	54	55	56	57	58	59	61.54
	0	5	12	48	49	50	51	52	53	54	55	56	57	58	59	45.19
AG2	9	10	11	19	21	24	29	30	34	35	36	38	41	42	44	91.35
	9	10	11	19	21	24	29	30	34	35	36	38	41	42	43	92.31

Tabla 4.15. Conjuntos de 15 variables y porcentaje de acierto sobre datos de prueba con 1-NN

4.4.1 Selección de Subconjuntos de 6 Variables

4.4.1.1 AGE I (Cruce Parcialmente Complementario)

Variables seleccionadas (Cto. 1): 5, 10, 11, 20, 26, 42

Algoritmo	Prueba
1-NN	76.92
C4.5	68.3

Tabla 4.16. Resultados obtenidos con el conjunto de variables 1

GENERACIÓN		Sin pond.		Con pond.	
MRD	NR	Entr.	Prueba	Entr.	Prueba
Clásico (f_0)	86	75.96	76.92	—	—
Suma (f_1)		72.11	73.08	74.04	75.96
M. Arit. (f_2)		73.08	75.00	73.08	75.96
M. Quasiarit. (f_3, 10)		78.85	78.85	76.92	76.92
M. Quasiarit. (f_3, 20)		76.92	76.92	75.00	76.92
SOWA OR (f_5, 0.3)		75.96	76.92	75.00	78.85
SOWA OR (f_5, 0.5)		75.00	78.85	75.00	76.92
SOWA OR (f_5, 0.9)		77.88	76.92	75.96	76.92
Badd (f_6, 10)		75.96	75.96	76.92	76.92
Badd (f_6, 20)		76.92	77.88	75.96	76.92
OWA (f_7, 0, 0.3)		74.03	80.77	75.96	81.73
OWA (f_7, 0, 0.5)		73.08	78.85	75.96	80.77
OWA (f_7, 0, 0.8)		72.11	75.96	74.04	78.85
QuasiOWA (f_8, 0, 0.3, 10)		77.88	78.85	—	—
QuasiOWA (f_8, 0, 0.5, 10)		78.85	78.85	—	—
QuasiOWA (f_8, 0, 0.8, 10)		78.85	78.85	—	—
QuasiOWA (f_8, 0, 0.3, 20)		75.96	76.92	—	—
QuasiOWA (f_8, 0, 0.5, 20)		75.96	76.92	—	—
QuasiOWA (f_8, 0, 0.8, 20)		75.96	76.92	—	—

Tabla 4.17. Resultados generación con el conjunto de variables 1

MRD OWA $f_7, 0, 0.3$	Sin Modif.			Modif. I			Modif. II		
ETAPA	NR	Entr.	Prueba	NR	Entr.	Prueba	NR	Entr.	Prueba
MULTISELECCIÓN	50	85.58	83.65	50	88.46	77.88	47	93.27	77.88
	41	87.50	**85.58**	64	91.35	76.92	68	94.23	78.85
	51	85.58	83.65	58	87.50	80.77	68	96.15	82.69
AJUSTE	50	88.46	83.65	50	90.38	78.85	47	93.27	76.92
	41	87.50	84.61	64	93.27	76.92	68	94.23	77.88
	51	89.42	83.65	58	88.46	80.77	68	97.11	80.77

Tabla 4.18. Resultados postprocesamiento con el conjunto de variables 1 y el MRD OWA $(f_7, 0, 0.3)$

MRD OWA P. $f_7, 0, 0.3$	Sin Modif.			Modif. I			Modif. II		
ETAPA	NR	Entr.	Prueba	NR	Entr.	Prueba	NR	Entr.	Prueba
MULTISELECCIÓN	46	86.54	78.85	39	89.42	82.69	78	92.31	80.77
	51	85.58	**85.58**	48	89.42	76.92	81	94.23	82.69
	46	86.54	**85.58**	57	86.54	82.69	43	93.27	75.00
AJUSTE	46	88.46	76.92	39	89.42	80.77	78	93.27	82.69
	51	88.46	84.61	48	93.27	78.85	81	94.23	82.69
	46	87.50	82.69	57	91.35	82.69	43	93.27	75.96

Tabla 4.19. Resultados postprocesamiento con el conjunto de variables 1 y el MRD OWA Ponderado (f_7)

Variables seleccionadas (Cto. 2): 9, 17, 21, 35, 37, 45

Algoritmo	Prueba
1-NN	88.46
C4.5	69.20

Tabla 4.20. Resultados obtenidos con el conjunto de variables 2

GENERACIÓN		Sin pond.		Con pond.	
MRD	NR	Entr.	Prueba	Entr.	Prueba
Clásico (f_0)	111	76.92	75.00	—	—
Suma (f_1)		80.77	79.81	80.77	81.73
M. Arit. (f_2)		77.88	80.77	78.85	79.81
M. Quasiarit. (f_3, 10)		77.88	75.96	76.92	75.00
M. Quasiarit. (f_3, 20)		76.92	75.00	76.92	75.00
SOWA OR (f_5, 0.3)		78.85	81.73	78.85	78.85
SOWA OR (f_5, 0.5)		78.85	76.92	77.88	75.00
SOWA OR (f_5, 0.7)		78.85	75.00	77.88	75.00
SOWA OR (f_5, 0.8)		76.92	75.00	77.88	75.00
SOWA OR (f_5, 0.9)		76.92	75.00	76.92	75.00
Badd (f_6, 10)		75.96	75.00	76.92	75.00
Badd (f_6, 20)		76.92	75.00	76.92	75.00
OWA (f_7, 0, 0.3)		79.81	81.73	79.81	80.77
OWA (f_7, 0, 0.5)		79.81	79.81	80.77	80.77
OWA (f_7, 0, 0.8)		76.92	79.81	79.81	81.73
QuasiOWA (f_8, 0, 0.3, 10)		77.88	75.96	—	—
QuasiOWA (f_8, 0, 0.5, 10)		77.88	75.96	—	—
QuasiOWA (f_8, 0, 0.8, 10)		77.88	75.96	—	—
QuasiOWA (f_8, 0, 0.3, 20)		76.92	75.00	—	—
QuasiOWA (f_8, 0, 0.5, 20)		76.92	75.00	—	—
QuasiOWA (f_8, 0, 0.8, 20)		76.92	75.00	—	—

Tabla 4.21. Resultados generación con el conjunto de variables 2

SOWA OR $f_5, 0.3$	Sin Modif.			Modif. I			Modif. II		
ETAPA	NR	Entr.	Prueba	NR	Entr.	Prueba	NR	Entr.	Prueba
MULTISELECCIÓN	59	90.38	85.58	65	93.27	83.65	92	94.23	80.77
	59	88.46	80.77	63	91.35	80.77	100	95.19	88.46
	58	88.46	82.69	75	92.31	79.81	105	97.11	78.85
AJUSTE	59	93.27	83.65	65	94.23	84.61	92	95.19	81.73
	59	92.31	84.61	63	94.23	81.73	100	95.19	**89.42**
	58	92.31	85.58	75	93.27	82.69	105	98.08	79.81

Tabla 4.22. Resultados postprocesamiento con el conjunto de variables 2 y el MRD SOWA Or-Like ($f_5, 0.3$)

MRD OWA $f_7, 0, 0.3$	Sin Modif.			Modif. I			Modif. II		
ETAPA	NR	Entr.	Prueba	NR	Entr.	Prueba	NR	Entr.	Prueba
MULTISELECCIÓN	73	90.38	77.88	70	93.27	78.86	55	94.23	88.46
	71	91.35	82.69	82	92.31	80.77	107	92.31	87.50
	67	91.35	78.85	64	94.23	83.65	91	94.23	82.69
AJUSTE	73	93.27	80.77	70	93.27	82.69	55	94.23	**89.42**
	71	91.35	84.61	82	92.31	80.77	107	92.31	**89.42**
	67	91.35	80.77	64	96.15	86.54	91	94.23	83.65

Tabla 4.23. Resultados postprocesamiento con el conjunto de variables 2 y el MRD OWA (f_7, 0, 0.3)

4.4.1.2 AGE II (Cruce Multipunto con Reparación)

Variables seleccionadas (Cto. 3): 11, 13, 23, 30, 36, 44

Algoritmo	Prueba
1-NN	80.77
C4.5	73.1

Tabla 4.24. Resultados obtenidos con el conjunto de variables 3

GENERACIÓN		Sin pond.		Con pond.	
MRD	NR	Entr.	Prueba	Entr.	Prueba
Clásico (f_0)	112	77.88	83.65	—	—
Suma (f_1)		73.08	77.88	80.77	81.73
M. Arit. (f_2)		79.81	76.92	78.85	80.77
M. Quasiarit. (f_3, 10)		77.88	84.61	77.88	84.61
M. Quasiarit. (f_3, 20)		77.88	84.61	76.92	83.65
SOWA OR (f_5, 0.3)		78.85	83.65	78.85	84.61
SOWA OR (f_5, 0.5)		78.85	84.61	78.85	83.65
SOWA OR (f_5, 0.7)		78.85	84.61	78.85	84.61
SOWA OR (f_5, 0.8)		78.85	84.61	78.85	83.65
SOWA OR (f_5, 0.9)		78.84	83.65	77.88	83.65
Badd (f_6, 10)		75.96	83.65	76.92	82.69
Badd (f_6, 20)		76.92	82.69	77.88	82.69
OWA (f_7, 0, 0.3)		76.92	85.58	78.85	**86.54**
OWA (f_7, 0, 0.5)		77.88	80.77	78.85	82.70
OWA (f_7, 0, 0.8)		76.92	77.88	80.77	80.77
QuasiOWA (f_8, 0, 0.3, 10)		77.88	84.61	—	—
QuasiOWA (f_8, 0, 0.5, 10)		77.88	84.61	—	—
QuasiOWA (f_8, 0, 0.8, 10)		77.88	84.61	—	—
QuasiOWA (f_8, 0, 0.3, 20)		77.88	84.61	—	—
QuasiOWA (f_8, 0, 0.5, 20)		77.88	84.61	—	—
QuasiOWA (f_8, 0, 0.8, 20)		77.88	84.61	—	—

Tabla 4.25. Resultados generación con el conjunto de variables 3

M. Quasiar. $f_3, 10$	Sin Modif.			Modif. I			Modif. II		
ETAPA	NR	Entr.	Prueba	NR	Entr.	Prueba	NR	Entr.	Prueba
MULTISELECCIÓN	70	85.58	83.65	74	88.46	80.77	87	92.31	84.61
	70	83.65	83.65	69	86.54	**86.54**	101	91.35	82.69
	78	84.62	83.65	70	88.46	79.81	106	93.27	77.88
AJUSTE	70	86.54	84.61	74	88.46	80.77	87	92.31	83.65
	70	86.54	81.73	69	90.38	82.69	101	92.31	84.61
	78	88.46	82.69	70	89.42	77.88	106	93.27	80.77

Tabla 4.26. Resultados postprocesamiento con el conjunto de variables 3 y el MRD Media Quasiaritmética (f_3, p=10)

OWA P. $f_7, 0, 0.3$	Sin Modif.			Modif. I			Modif. II		
ETAPA	NR	Entr.	Prueba	NR	Entr.	Prueba	NR	Entr.	Prueba
MULTISELECCIÓN	78	83.65	81.73	66	87.50	76.92	106	92.30	79.81
	66	83.65	80.77	55	89.42	82.70	95	91.35	79.80
	81	83.65	83.65	71	88.46	81.73	109	90.38	77.88
AJUSTE	79	86.54	80.77	66	91.35	77.88	106	93.27	79.81
	66	87.50	78.85	55	91.35	78.85	95	91.35	78.85
	81	89.42	81.73	71	90.38	81.76	109	91.35	79.81

Tabla 4.27. Resultados postprocesamiento con el conjunto de variables 3 y el MRD OWA Ponderado (f_7, 0, 0.3)

Variables seleccionadas (Cto. 4): 10, 19, 22, 35, 45, 49

Algoritmo	Prueba
1-NN	90.38
C4.5	83.7

Tabla 4.28. Resultados obtenidos con el conjunto de variables 4

GENERACIÓN		Sin pond.		Con pond.	
MRD	NR	Entr.	Prueba	Entr.	Prueba
Clásico (f_0)	99	75.00	75.96	—	—
Suma (f_1)		71.15	76.92	71.15	78.85
M. Arit. (f_2)		72.11	83.65	69.23	83.65
M. Quasiarit. (f_3, 10)		75.96	78.84	75.00	78.84
M. Quasiarit. (f_3, 20)		75.00	78.84	75.00	76.92
SOWA OR (f_5, 0.3)		71.15	79.80	74.04	78.85
SOWA OR (f_5, 0.5)		74.03	79.81	73.08	79.80
SOWA OR (f_5, 0.7)		74.04	79.81	75.00	77.88
SOWA OR (f_5, 0.8)		75.00	78.85	75.00	77.88
SOWA OR (f_5, 0.9)		75.00	76.92	75.00	76.92
Badd (f_6, 10)		75.00	75.96	75.00	75.96
Badd (f_6, 20)		75.00	75.96	75.00	75.96
OWA (f_7, 0, 0.3)		76.92	81.73	75.96	80.97
OWA (f_7, 0, 0.5)		73.07	85.58	71.15	82.69
OWA (f_7, 0, 0.8)		71.15	82.69	70.19	82.69
QuasiOWA (f_8, 0, 0.3, 10)		75.96	78.85	—	—
QuasiOWA (f_8, 0, 0.5, 10)		75.96	78.85	—	—
QuasiOWA (f_8, 0, 0.8, 10)		75.96	78.85	—	—
QuasiOWA (f_8, 0, 0.3, 20)		75.00	77.88	—	—
QuasiOWA (f_8, 0, 0.5, 20)		75.00	78.85	—	—
QuasiOWA (f_8, 0, 0.8, 20)		75.00	78.85	—	—

Tabla 4.29. Resultados generación con el conjunto de variables 4

M. Aritm. P. f_2	Sin Modif.			Modif. I			Modif. II		
ETAPA	NR	Tra	Prueba	NR	Entr.	Prueba	NR	Tra	Prueba
MULTISELECCIÓN	43	87.50	81.73	63	93.27	82.69	58	93.27	**90.38**
	51	87.50	79.81	50	93.27	81.73	45	90.38	80.77
	45	88.46	82.69	66	92.31	82.69	52	91.35	84.61
AJUSTE	43	89.42	79.81	63	95.19	82.69	58	93.27	89.42
	51	90.38	81.73	50	93.26	81.73	45	90.38	81.73
	45	90.38	85.58	66	94.23	80.77	52	91.35	86.54

Tabla 4.30. Resultados postprocesamiento con el conjunto de variables 4 y el MRD Media Aritmética Ponderada (f_2)

OWA $f_7, 0, 0.5$	Sin Modif.			Modif. I			Modif. II		
ETAPA	NR	Entr.	Prueba	NR	Entr.	Prueba	NR	Entr.	Prueba
MULTISELECCIÓN	50	86.54	75.96	59	94.23	83.65	54	89.42	89.42
	54	86.54	81.73	58	94.23	85.57	67	93.27	82.69
	50	88.46	84.61	70	94.23	84.61	55	92.31	88.46
	50	88.46	78.85	59	94.23	83.65	54	91.35	85.58
	54	93.27	79.81	58	94.23	83.65	67	93.27	80.77
	50	90.38	81.73	70	94.23	85.58	55	92.31	88.46

Tabla 4.31. Resultados postprocesamiento con el conjunto de variables 4 y el MRD OWA ($f_7, 0, 0.5$)

4.4.1.3 MIFS

Variables seleccionadas (Cto. 5): 11, 47, 54, 55, 56, 57

Algoritmo	Prueba
1-NN	69.23
C4.5	71.2

Tabla 4.32. Resultados obtenidos con el conjunto de variables 5

GENERACIÓN		Sin pond.		Con pond.	
MRD	NR	Entr.	Prueba	Entr.	Prueba
Clásico (f_0)	42	68.27	63.46	—	—
Suma (f_1)		72.11	**76.92**	72.11	75.96
M. Arit. (f_2)		69.23	69.23	69.23	66.35
M. Quasiarit. (f_3, 10)		69.23	63.46	68.27	63.46
M. Quasiarit. (f_3, 20)		68.27	63.46	68.27	63.46
SOWA OR (f_5, 0.3)		70.19	66.35	67.31	64.42
SOWA OR (f_5, 0.5)		66.35	64.42	68.27	64.42
SOWA OR (f_5, 0.7)		68.27	64.42	69.23	63.46
SOWA OR (f_5, 0.8)		68.27	63.46	69.23	63.46
SOWA OR (f_5, 0.9)		69.23	63.46	68.27	63.46
Badd (f_6, 10)		68.27	63.46	68.27	63.46
Badd (f_6, 20)		68.27	63.46	68.27	63.46
OWA (f_7, 0, 0.3)		68.27	63.46	68.27	63.46
OWA (f_7, 0, 0.5)		69.23	62.50	68.27	62.50
OWA (f_7, 0, 0.8)		69.23	68.27	69.23	66.35
QuasiOWA (f_8, 0, 0.3, 10)		68.27	63.46	—	—
QuasiOWA (f_8, 0, 0.5, 10)		69.23	63.46	—	—
QuasiOWA (f_8, 0, 0.8, 10)		69.23	63.46	—	—
QuasiOWA (f_8, 0, 0.3, 20)		68.27	63.46	—	—
QuasiOWA (f_8, 0, 0.5, 20)		68.27	63.46	—	—
QuasiOWA (f_8, 0, 0.8, 20)		68.27	63.46	—	—

Tabla 4.33. Resultados generación con el conjunto de variables 5

Suma f_1	Sin Modif.			Modif. I			Modif. II		
ETAPA	NR	Entr.	Prueba	NR	Entr.	Prueba	NR	Entr.	Prueba
MULTISELECCIÓN	24	75.00	75.96	20	75.96	74.04	36	76.92	72.11
	25	75.00	75.96	27	76.92	68.27	29	78.85	71.15
	28	75.00	75.00	23	76.92	74.04	40	76.92	71.15
AJUSTE	24	75.96	75.96	20	75.96	75.00	36	78.85	71.15
	25	75.00	75.96	27	76.92	70.19	29	78.85	72.11
	28	75.96	75.96	23	76.92	74.04	40	76.92	72.11

Tabla 4.34. Resultados postprocesamiento con el conjunto de variables 5 y el MRD Suma (f_1)

Suma P. f_1	Sin Modif.			Modif. I			Modif. II		
ETAPA	NR	Entr.	Prueba	NR	Entr.	Prueba	NR	Entr.	Prueba
MULTISELECCIÓN	26	76.92	72.11	23	76.92	74.04	35	77.88	73.08
	23	76.92	72.11	31	76.92	71.15	27	77.88	**76.92**
	26	75.96	73.08	28	77.88	72.11	24	78.85	72.11
AJUSTE	26	76.92	71.15	23	76.92	71.15	35	77.88	73.08
	23	77.88	71.15	31	77.88	71.15	27	77.88	75.96
	26	76.92	73.08	28	77.88	72.11	24	79.81	71.15

Tabla 4.35. Resultados postprocesamiento con el conjunto de variables 5 y el MRD Suma Ponderada (f_1)

4.4.1.4 AG1

Variables seleccionadas (Cto. 6): 11, 47, 54, 57, 58, 59

Algoritmo	Prueba
1-NN	69.23
C4.5	67.3

Tabla 4.36. Resultados obtenidos con el conjunto de variables 6

GENERACIÓN		Sin pond.		Con pond.	
MRD	NR	Entr.	Prueba	Entr.	Prueba
Clásico (f_0)	42	68.27	63.46	—	—
Suma (f_1)		72.11	**76.92**	72.11	75.96
M. Arit. (f_2)		69.23	69.23	69.23	66.34
M. Quasiarit. (f_3, 10)		69.23	63.46	68.27	63.46
M. Quasiarit. (f_3, 20)		68.27	63.46	68.27	63.46
SOWA OR (f_5, 0.3)		70.19	66.35	67.31	64.42
SOWA OR (f_5, 0.5)		66.35	64.42	68.27	64.42
SOWA OR (f_5, 0.7)		68.27	64.42	69.23	63.46
SOWA OR (f_5, 0.8)		68.27	63.46	69.23	63.46
SOWA OR (f_5, 0.9)		69.23	63.46	68.27	63.46
Badd (f_6, 10)		68.27	63.46	68.27	63.46
Badd (f_6, 20)		68.27	63.46	68.27	63.46
OWA (f_7, 0, 0.3)		68.27	63.46	68.27	63.46
OWA (f_7, 0, 0.5)		69.23	62.50	68.27	62.50
OWA (f_7, 0, 0.8)		69.23	68.27	69.23	66.35
QuasiOWA (f_8, 0, 0.3, 10)		68.27	63.46	—	—
QuasiOWA (f_8, 0, 0.5, 10)		69.23	63.46	—	—
QuasiOWA (f_8, 0, 0.8, 10)		69.23	63.46	—	—
QuasiOWA (f_8, 0, 0.3, 20)		68.27	63.46	—	—
QuasiOWA (f_8, 0, 0.5, 20)		68.27	63.46	—	—
QuasiOWA (f_8, 0, 0.8, 20)		68.27	63.46	—	—

Tabla 4.37. Resultados generación con el conjunto de variables 6

Suma f_1	Sin Modif.			Modif. I			Modif. II		
ETAPA	NR	Entr.	Prueba	NR	Entr.	Prueba	NR	Entr.	Prueba
MULTISELECCIÓN	24	75.00	75.96	25	75.96	70.19	40	75.96	74.04
	25	75.00	75.96	30	75.96	75.00	41	75.96	75.96
	28	75.00	75.00	33	75.96	74.04	31	75.96	74.04
AJUSTE	24	75.96	75.96	25	77.88	70.19	40	76.92	68.27
	25	75.96	75.96	30	75.96	75.00	41	75.96	75.96
	28	75.96	75.00	33	75.96	74.04	31	75.96	75.96

Tabla 4.38. Resultados postprocesamiento con el conjunto de variables 6 y el MRD Suma (f_1)

M. Aritm. f_2	Sin Modif.			Modif. I			Modif. II		
ETAPA	NR	Entr.	Prueba	NR	Entr.	Prueba	NR	Entr.	Prueba
MULTISELECCIÓN	19	77.88	73.08	25	76.92	71.15	26	77.88	70.19
	24	76.92	73.08	17	76.92	74.04	27	76.92	75.00
	17	77.88	71.15	27	75.96	73.08	26	77.88	72.11
AJUSTE	19	78.85	69.23	25	81.73	69.23	26	77.88	72.11
	24	80.77	73.08	17	77.88	73.08	27	79.81	71.15
	17	77.88	69.23	27	78.85	72.11	26	79.81	74.04

Tabla 4.39. Resultados postprocesamiento con el conjunto de variables 6 y el MRD Media Aritmética (f_2)

4.4.1.5 AG2

Variables seleccionadas (Cto. 7): 11, 15, 23, 30, 36, 44

Algoritmo	Prueba
1-NN	78.85
C4.5	67.3

Tabla 4.40. Resultados obtenidos con el conjunto de variables 7

GENERACIÓN		Sin pond.		Con pond.	
MRD	NR	Entr.	Prueba	Entr.	Prueba
Clásico (f_0)	113	85.58	82.69	—	—
Suma (f_1)		79.81	81.73	83.65	82.69
M. Arit. (f_2)		83.65	79.81	83.65	82.69
M. Quasiarit. (f_3, 10)		84.61	83.65	85.58	83.65
M. Quasiarit. (f_3, 20)		85.58	83.65	85.58	82.69
SOWA OR (f_5, 0.3)		83.65	85.58	83.65	**86.54**
SOWA OR (f_5, 0.5)		83.65	85.58	84.61	85.58
SOWA OR (f_5, 0.7)		84.61	85.58	84.61	84.61
SOWA OR (f_5, 0.8)		84.61	84.61	84.61	83.65
SOWA OR (f_5, 0.9)		84.61	82.69	85.58	82.69
Badd (f_6, 10)		84.61	82.69	85.58	83.65
Badd (f_6, 20)		85.58	83.65	85.58	82.69
OWA (f_7, 0, 0.3)		82.69	84.61	81.73	83.65
OWA (f_7, 0, 0.5)		81.73	85.58	82.69	85.58
OWA (f_7, 0, 0.8)		83.65	77.88	84.61	81.73
QuasiOWA (f_8, 0, 0.3, 10)		85.57	82.69	—	—
QuasiOWA (f_8, 0, 0.5, 10)		84.61	83.65	—	—
QuasiOWA (f_8, 0, 0.8, 10)		84.61	83.65	—	—
QuasiOWA (f_8, 0, 0.3, 20)		85.58	83.65	—	—
QuasiOWA (f_8, 0, 0.5, 20)		85.58	83.65	—	—
QuasiOWA (f_8, 0, 0.8, 20)		85.58	83.65	—	—

Tabla 4.41. Resultados generación con el conjunto de variables 7

SOWA OR P. $f_5, 0.3$	Sin Modif.			Modif. I			Modif. II		
ETAPA	NR	Entr.	Prueba	NR	Entr.	Prueba	NR	Entr.	Prueba
MULTISELECCIÓN	56	92.31	75.96	76	92.31	80.77	66	96.15	76.92
	61	92.31	80.77	66	92.31	85.58	106	95.19	78.85
	64	92.31	77.88	88	92.31	82.69	106	97.11	80.77
AJUSTE	56	92.31	75.00	76	92.31	81.73	66	96.15	75.96
	61	93.27	80.77	66	92.31	84.61	106	95.19	77.88
	64	93.27	78.85	88	93.27	83.65	106	97.11	80.77

Tabla 4.42. Resultados postprocesamiento con el conjunto de variables 7 y el MRD SOWA Or Like Ponderado ($f_5, \alpha = 0.3$)

OWA $f_7, 0, 0.5$	Sin Modif.			Modif. I			Modif. II		
ETAPA	NR	Entr.	Prueba	NR	Entr.	Prueba	NR	Entr.	Prueba
MULTISELECCIÓN	83	90.38	80.77	69	93.27	83.65	109	94.23	85.58
	65	92.31	81.73	78	93.27	80.77	108	92.31	82.69
	63	92.31	83.65	83	91.35	83.65	106	94.23	79.81
AJUSTE	83	92.31	81.73	69	94.23	80.77	109	94.23	83.65
	65	94.23	81.73	78	94.23	80.77	108	93.27	80.77
	63	93.27	83.65	83	93.27	84.61	106	94.23	78.85

Tabla 4.43. Resultados postprocesamiento con el conjunto de variables 7 y el MRD OWA ($f_7, 0, 0.5$)

4.4.2 Selección de Subconjuntos de 12 Variables

4.4.2.1 AGE I (Cruce Parcialmente Complementario)

Variables seleccionadas (Cto. 8): 0, 9, 10, 11, 15, 23, 24, 30, 36, 42, 44, 48

Algoritmo	Prueba
1-NN	91.35
C4.5	78.8

Tabla 4.44. Resultados obtenidos con el conjunto de variables 8

GENERACIÓN		Sin pond.		Con pond.	
FRM	NR	Entr.	Prueba	Entr.	Prueba
Clásico (f_0)	148	86.54	87.50	—	—
Suma (f_1)		81.73	83.65	86.54	89.42
M. Arit. (f_2)		82.69	81.73	85.58	87.50
M. Quasiarit. (f_3, 10)		84.61	88.46	85.58	86.54
M. Quasiarit. (f_3, 20)		85.58	86.54	85.58	88.46
SOWA OR (f_5, 0.3)		89.42	86.54	87.50	86.54
SOWA OR (f_5, 0.5)		86.54	86.54	85.58	88.46
SOWA OR (f_5, 0.7)		84.61	87.50	86.54	87.50
SOWA OR (f_5, 0.8)		86.54	87.50	86.54	87.50
SOWA OR (f_5, 0.9)		86.54	87.50	86.54	87.50
Badd (f_6, 10)		85.58	87.50	85.58	87.50
Badd (f_6, 20)		85.58	87.50	86.54	87.50
OWA (f_7, 0, 0.3)		86.54	88.46	85.58	88.46
OWA (f_7, 0, 0.5)		85.58	86.54	87.50	88.46
OWA (f_7, 0, 0.8)		82.69	80.77	85.58	86.54
QuasiOWA (f_8, 0, 0.3, 10)		84.61	86.54	—	—
QuasiOWA (f_8, 0, 0.5, 10)		84.61	88.46	—	—
QuasiOWA (f_8, 0, 0.8, 10)		84.61	88.46	—	—
QuasiOWA (f_8, 0, 0.3, 20)		85.58	85.58	—	—
QuasiOWA (f_8, 0, 0.5, 20)		85.58	86.54	—	—
QuasiOWA (f_8, 0, 0.8, 20)		85.58	86.54	—	—

Tabla 4.45. Resultados generación con el conjunto de variables 8

Suma P. f_1	Sin Modif.			Modif. I			Modif. II		
ETAPA	NR	Entr.	Prueba	NR	Entr.	Prueba	NR	Entr.	Prueba
MULTISELECCIÓN	104	95.19	89.42	107	96.15	83.65	137	95.19	89.42
	97	96.15	89.42	79	96.15	83.65	124	97.11	88.46
	100	96.15	87.50	113	95.19	86.54	143	98.08	85.58
AJUSTE	104	96.15	89.42	107	96.15	84.61	137	95.19	87.50
	97	96.15	**90.38**	79	98.08	83.65	124	97.11	87.50
	100	97.11	88.46	113	96.15	85.58	143	98.08	86.54

Tabla 4.46. Resultados postprocesamiento con el conjunto de variables 8 y el MRD Suma Ponderada (f_1)

QuasiOWA $f_8, 0, 0.5, 10$	Sin Modif.			Modif. I			Modif. II		
ETAPA	NR	Entr.	Prueba	NR	Entr.	Prueba	NR	Entr.	Prueba
MULTISELECCIÓN	88	93.27	87.50	99	96.15	87.50	143	97.11	84.61
(f_8, 0, 0.5, 10)	83	94.23	83.65	100	95.19	89.42	146	97.11	86.54
	94	93.27	88.46	102	94.23	83.65	145	97.11	86.54
AJUSTE	88	96.15	87.50	99	97.11	86.54	143	97.11	81.73
(f_8, 0, 0.5, 10)	83	96.15	86.54	100	97.11	87.50	146	97.11	87.50
	94	95.19	**90.38**	102	95.19	82.69	145	97.11	88.46

Tabla 4.47. Resultados postprocesamiento con el conjunto de variables 8 y el MRD QuasiOWA ($f_8, 0, 0.5, 10$)

Variables seleccionadas (Cto. 9): 8, 10, 16, 21, 29, 30, 35, 36, 38, 41, 42, 46

Algoritmo	Prueba
1-NN	92.31
C4.5	79.8

Tabla 4.48. Resultados obtenidos con el conjunto de variables 9

GENERACIÓN		Sin pond.		Con pond.	
FRM	NR	Entr.	Prueba	Entr.	Prueba
Clásico (f_0)	183	90.38	92.31	—	—
Suma (f_1)		86.54	82.69	89.42	87.50
M. Arit. (f_2)		88.46	87.50	91.35	88.46
M. Quasiarit. (f_3, 10)		92.31	92.31	92.31	92.31
M. Quasiarit. (f_3, 20)		92.31	92.31	91.35	93.27
SOWA OR (f_5, 0.3)		92.31	91.35	93.27	93.27
SOWA OR (f_5, 0.5)		92.31	92.31	92.31	**94.23**
SOWA OR (f_5, 0.7)		92.31	93.27	91.35	93.27
SOWA OR (f_5, 0.8)		91.35	93.27	91.35	93.27
SOWA OR (f_5, 0.9)		90.38	93.27	90.38	92.31
Badd (f_6, 10)		90.38	91.35	90.38	92.31
Badd (f_6, 20)		90.38	92.31	90.38	92.31
OWA (f_7, 0, 0.3)		91.35	90.38	93.27	91.35
OWA (f_7, 0, 0.5)		90.38	89.42	92.31	90.38
OWA (f_7, 0, 0.8)		89.42	86.54	91.35	88.46
QuasiOWA (f_8, 0, 0.3, 10)		92.31	92.31	—	—
QuasiOWA (f_8, 0, 0.5, 10)		92.31	92.31	—	—
QuasiOWA (f_8, 0, 0.8, 10)		92.31	92.31	—	—
QuasiOWA (f_8, 0, 0.3, 20)		92.31	92.31	—	—
QuasiOWA (f_8, 0, 0.5, 20)		92.31	92.31	—	—
QuasiOWA (f_8, 0, 0.8, 20)		92.31	92.31	—	—

Tabla 4.49. Resultados generación con el conjunto de variables 9

SOWA OR $f_5, \alpha = 0.5$	Sin Modif.			Modif. I			Modif. II		
ETAPA	NR	Entr.	Prueba	NR	Entr.	Prueba	NR	Entr.	Prueba
MULTISELECCIÓN	113	96.15	91.35	120	97.11	90.38	182	98.08	92.31
	126	96.15	91.35	127	98.08	90.38	178	98.08	92.31
	101	97.11	91.35	124	98.08	86.54	178	98.08	93.27
AJUSTE	113	98.08	91.35	120	98.08	83.65	182	98.08	92.31
	126	98.08	93.27	127	98.08	90.38	178	98.08	90.38
	101	98.08	90.38	124	98.08	84.61	178	98.08	92.31

Tabla 4.50. Resultados postprocesamiento con el conjunto de variables 9 y el MRD SOWA Or-Like $(f_5, 0.5)$

M. Quasiar. $f_3, 20$	Sin Modif.			Modif. I			Modif. II		
ETAPA	NR	Entr.	Prueba	NR	Entr.	Prueba	NR	Entr.	Prueba
MULTISELECCIÓN	106	95.19	90.38	116	98.08	87.50	180	98.08	91.35
	114	95.19	91.35	127	98.08	89.42	176	98.08	92.31
	124	95.19	90.38	124	98.08	86.54	178	98.08	88.46
AJUSTE	106	97.11	87.50	116	98.08	83.65	180	98.08	87.50
	114	98.08	91.35	127	98.08	92.31	176	98.08	92.31
	124	98.08	89.42	124	98.08	84.61	178	98.08	88.46

Tabla 4.51. Resultados postprocesamiento con el conjunto de variables 9 y el MRD Media Quasiaritmética $(f_3, 20)$

4.4.2.2 AGE II (Cruce Multipunto con Reparación)

Variables seleccionadas (Cto. 10): 2, 9, 11, 16, 21, 29, 32, 35, 36, 42, 53, 54

Algoritmo	Prueba
1-NN	93.27
C4.5	68.3

Tabla 4.52. Resultados obtenidos con el conjunto de variables 10

GENERACIÓN		Sin pond.		Con pond.	
MRD	NR	Entr.	Prueba	Entr.	Prueba
Clásico (f_0)	175	87.50	80.77	—	—
Suma (f_1)		85.58	80.77	86.54	85.58
M. Arit. (f_2)		85.58	78.85	85.58	84.61
M. Quasiarit. (f_3, 10)		88.46	83.65	88.46	79.81
M. Quasiarit. (f_3, 20)		88.46	79.81	88.46	80.77
SOWA OR (f_5, 0.3)		86.54	83.65	86.54	82.69
SOWA OR (f_5, 0.5)		86.54	81.73	86.54	81.73
SOWA OR (f_5, 0.7)		86.54	81.73	88.46	79.81
SOWA OR (f_5, 0.8)		88.46	80.77	88.46	80.77
SOWA OR (f_5, 0.9)		88.46	80.77	88.46	80.77
Badd (f_6, 10)		87.50	80.77	87.50	81.73
Badd (f_6, 20)		87.50	81.73	87.50	80.77
OWA (f_7, 0, 0.3)		86.54	82.69	85.58	82.69
OWA (f_7, 0, 0.5)		84.61	83.65	85.58	83.65
OWA (f_7, 0, 0.8)		84.61	83.65	85.58	86.54
QuasiOWA (f_8, 0, 0.3, 10)		88.46	82.69	—	—
QuasiOWA (f_8, 0, 0.5, 10)		88.46	82.69	—	—
QuasiOWA (f_8, 0, 0.8, 10)		88.46	83.65	—	—
QuasiOWA (f_8, 0, 0.3, 20)		88.46	79.81	—	—
QuasiOWA (f_8, 0, 0.5, 20)		88.46	79.81	—	—
QuasiOWA (f_8, 0, 0.8, 20)		88.46	79.81	—	—

Tabla 4.53. Resultados generación con el conjunto de variables 10

Suma P. f_1	Sin Modif.			Modif. I			Modif. II		
ETAPA	NR	Entr.	Prueba	NR	Entr.	Prueba	NR	Entr.	Prueba
MULTISELECCIÓN	114	94.23	82.69	120	95.19	84.61	171	95.19	87.50
	110	96.15	82.69	127	96.15	83.65	171	95.19	77.88
	116	95.19	85.58	117	95.19	79.81	164	96.15	80.77
AJUSTE	114	94.23	80.77	120	96.15	82.69	171	95.19	86.54
	110	97.11	79.81	127	96.15	83.65	171	95.19	78.85
	116	96.15	85.58	117	95.19	79.81	164	96.15	80.77

Tabla 4.54. Resultados postprocesamiento con el conjunto de variables 10 y el MRD Suma Ponderada (f_1)

M. Aritm. P. f_2	Sin Modif.			Modif. I			Modif. II		
ETAPA	NR	Entr.	Prueba	NR	Entr.	Prueba	NR	Entr.	Prueba
MULTISELECCIÓN	88	95.19	86.54	133	95.19	82.69	173	96.15	85.58
	93	94.23	80.77	117	94.23	80.77	165	96.15	84.61
	95	96.15	83.65	124	96.15	82.69	147	97.11	82.69
AJUSTE	88	96.15	**89.42**	133	95.19	83.65	173	96.15	87.50
	92	94.23	80.77	117	94.23	79.81	165	96.15	81.73
	95	97.11	80.77	124	97.11	85.58	147	97.11	81.73

Tabla 4.55. Resultados postprocesamiento con el conjunto de variables 10 y el MRD Media Aritmética Ponderada (f_2)

Variables seleccionadas (Cto. 11): 0, 1, 9, 16, 30, 35, 42, 51, 52, 53, 54, 56

Algoritmo	Prueba
1-NN	84.61
C4.5	71.2

Tabla 4.56. Resultados obtenidos con el conjunto de variables 11

GENERACIÓN		Sin pond.		Con pond.	
MRD	NR	Entr.	Prueba	Entr.	Prueba
Clásico (f_0)	85	75.96	78.85	—	—
Suma (f_1)		74.04	77.88	75.96	86.54
M. Arit. (f_2)		79.81	83.65	78.85	88.46
M. Quasiarit. (f_3, 10)		77.88	83.65	76.92	81.73
M. Quasiarit. (f_3, 20)		76.92	81.73	76.92	80.77
SOWA OR (f_5, 0.3)		80.77	82.69	77.88	82.69
SOWA OR (f_5, 0.5)		75.96	82.69	76.92	81.73
SOWA OR (f_5, 0.7)		76.92	79.81	76.92	79.81
SOWA OR (f_5, 0.8)		76.92	79.81	75.96	79.81
SOWA OR (f_5, 0.9)		75.96	79.81	75.96	78.85
Badd (f_6, 10)		76.92	77.88	75.96	78.85
Badd (f_6, 20)		75.96	78.85	75.96	78.85
OWA (f_7, 0, 0.3)		78.85	86.54	79.81	85.58
OWA (f_7, 0, 0.5)		80.77	87.50	79.81	87.50
OWA (f_7, 0, 0.8)		80.77	84.61	77.88	86.54
QuasiOWA (f_8, 0, 0.3, 10)		76.92	83.65	—	—
QuasiOWA (f_8, 0, 0.5, 10)		77.88	83.65	—	—
QuasiOWA (f_8, 0, 0.8, 10)		77.88	83.65	—	—
QuasiOWA (f_8, 0, 0.3, 20)		76.92	81.73	—	—
QuasiOWA (f_8, 0, 0.5, 20)		76.92	81.73	—	—
QuasiOWA (f_8, 0, 0.8, 20)		76.92	81.73	—	—

Tabla 4.57. Resultados generación con el conjunto de variables 11

M. Aritm. P. f_2	Sin Modif.			Modif. I			Modif. II		
ETAPA	NR	Entr.	Prueba	NR	Entr.	Prueba	NR	Entr.	Prueba
MULTISELECCIÓN	47	91.35	85.58	48	91.35	79.81	82	91.35	84.61
	47	92.31	87.50	43	92.31	88.46	81	92.31	86.54
	54	91.35	87.50	56	90.38	89.42	80	92.31	87.50
AJUSTE	47	92.31	87.50	48	91.35	80.77	82	92.31	80.77
	47	94.23	87.50	43	92.31	88.46	81	92.31	84.61
	54	91.35	87.50	56	92.31	84.61	80	93.27	86.54

Tabla 4.58. Resultados postprocesamiento con el conjunto de variables 11 y el MRD Media Aritmética Ponderada (f_2)

OWA $f_7, 0, 0.5$	Sin Modif.			Modif. I			Modif. II		
ETAPA	NR	Entr.	Prueba	NR	Entr.	Prueba	NR	Entr.	Prueba
MULTISELECCIÓN	45	91.35	86.54	49	91.35	87.50	70	94.23	85.58
	42	91.35	85.58	45	91.35	**90.38**	81	91.35	82.69
	51	91.35	87.50	44	93.27	81.73	80	95.19	81.73
AJUSTE	45	92.31	89.42	49	94.23	89.42	70	94.23	86.54
	42	92.31	83.65	45	92.31	87.50	81	92.31	81.73
	51	93.27	86.54	44	93.27	81.73	80	95.19	84.61

Tabla 4.59. Resultados postprocesamiento con el conjunto de variables 11 y el MRD OWA ($f_7, 0, 0.5$)

4.4.3 Selección de Subconjuntos de 15 Variables

4.4.3.1 AGE I (Cruce Parcialmente Complementario)

Variables seleccionadas (Cto. 12):	0, 1, 2, 8, 10, 11, 15, 29, 30, 36, 43, 44, 51, 55, 58

Algoritmo	Prueba
1-NN	85.58
C4.5	76.9

Tabla 4.60. Resultados obtenidos con el conjunto de variables 12

GENERACIÓN		Sin pond.		Con pond.	
MRD	NR	Entr.	Prueba	Entr.	Prueba
Clásico (f_0)	129	82.69	81.73	—	—
Suma (f_1)		79.81	90.38	86.54	**92.31**
M. Arit. (f_2)		82.69	83.65	85.58	84.61
M. Quasiarit. (f_3, 10)		83.65	82.69	82.69	81.73
M. Quasiarit. (f_3, 20)		82.69	81.73	82.69	82.69
SOWA OR (f_5, 0.3)		88.46	82.69	85.58	82.69
SOWA OR (f_5, 0.5)		85.58	80.77	84.61	82.69
SOWA OR (f_5, 0.7)		85.58	81.73	84.61	81.73
SOWA OR (f_5, 0.8)		84.61	81.73	83.65	82.69
SOWA OR (f_5, 0.9)		83.65	81.73	83.65	81.73
Badd (f_6, 10)		81.73	80.77	81.73	80.77
Badd (f_6, 20)		81.73	80.77	82.69	81.73
OWA (f_7, 0, 0.3)		86.54	81.73	87.50	81.73
OWA (f_7, 0, 0.5)		85.58	83.65	84.61	83.65
OWA (f_7, 0, 0.8)		83.65	83.65	85.58	84.61
QuasiOWA (f_8, 0, 0.3, 10)		83.65	82.69	—	—
QuasiOWA (f_8, 0, 0.5, 10)		82.69	82.69	—	—
QuasiOWA (f_8, 0, 0.8, 10)		83.65	82.69	—	—
QuasiOWA (f_8, 0, 0.3, 20)		82.69	82.69	—	—
QuasiOWA (f_8, 0, 0.5, 20)		82.69	81.73	—	—
QuasiOWA (f_8, 0, 0.8, 20)		82.69	81.73	—	—

Tabla 4.61. Resultados generación con el conjunto de variables 12

Suma f_1	Sin Modif.			Modif. I			Modif. II		
ETAPA	NR	Entr.	Prueba	NR	Entr.	Prueba	NR	Entr.	Prueba
MULTISELECCIÓN	94	92.31	85.58	83	92.31	82.69	126	95.19	90.38
	88	94.23	86.54	105	94.23	86.54	126	94.23	91.35
	87	93.27	86.54	103	96.15	83.65	125	95.19	90.38
AJUSTE	94	93.27	85.58	83	93.27	84.61	126	96.15	89.42
	88	94.23	85.58	105	95.19	86.54	126	94.23	91.35
	87	94.23	87.50	103	96.15	84.61	125	96.15	90.38

Tabla 4.62. Resultados postprocesamiento con el conjunto de variables 12 y el MRD Suma (f_1)

Suma P. f_1	Sin Modif.			Modif. I			Modif. II		
ETAPA	NR	Entr.	Prueba	NR	Entr.	Prueba	NR	Entr.	Prueba
MULTISELECCIÓN	94	92.31	83.65	89	93.27	83.65	124	95.19	88.46
	92	92.31	84.61	96	95.19	81.73	125	95.19	83.65
	73	94.23	77.88	99	95.19	86.54	125	97.11	91.35
AJUSTE	94	93.27	82.69	89	93.27	87.50	124	95.19	89.42
	92	92.31	85.58	96	96.15	79.81	125	95.19	87.50
	73	94.23	81.73	99	96.15	85.58	125	97.11	90.38

Tabla 4.63. Resultados postprocesamiento con el conjunto de variables 12 y el MRD Suma Ponderada (f_1)

Variables seleccionadas (Cto. 13): 2, 8, 10, 11, 15, 23, 24, 25, 30, 36, 41, 43, 49, 51, 53, 55

Algoritmo	Prueba
1-NN	92.31
C4.5	72.1

Tabla 4.64. Resultados obtenidos con el conjunto de variables 13

GENERACIÓN		Sin pond.		Con pond.	
MRD	NR	Entr.	Prueba	Entr.	Prueba
Clásico (f_0)	176	87.50	88.46	—	—
Suma (f_1)		78.85	85.58	86.54	88.46
M. Arit. (f_2)		84.61	85.58	89.42	87.50
M. Quasiarit. (f_3, 10)		87.50	89.42	87.50	90.38
M. Quasiarit. (f_3, 20)		87.50	90.38	87.50	89.42
SOWA OR (f_5, 0.3)		87.50	90.38	88.46	90.38
SOWA OR (f_5, 0.5)		88.46	91.35	88.46	90.38
SOWA OR (f_5, 0.7)		89.42	89.42	87.50	89.42
SOWA OR (f_5, 0.8)		88.46	89.42	87.50	88.46
SOWA OR (f_5, 0.9)		87.50	88.46	87.50	88.46
Badd (f_6, 10)		87.50	89.42	87.50	88.46
Badd (f_6, 20)		87.50	88.46	87.50	88.46
OWA (f_7, 0, 0.3)		87.50	88.46	88.46	87.50
OWA (f_7, 0, 0.5)		86.54	88.46	89.42	89.42
OWA (f_7, 0, 0.8)		85.58	86.54	88.46	88.46
QuasiOWA (f_8, 0, 0.3, 10)		88.46	89.42	—	—
QuasiOWA (f_8, 0, 0.5, 10)		88.46	90.38	—	—
QuasiOWA (f_8, 0, 0.8, 10)		87.50	89.42	—	—
QuasiOWA (f_8, 0, 0.3, 20)		87.50	89.42	—	—
QuasiOWA (f_8, 0, 0.5, 20)		87.50	90.38	—	—
QuasiOWA (f_8, 0, 0.8, 20)		87.50	90.38	—	—

Tabla 4.65. Resultados generación con el conjunto de variables 13

Clásico f_0	Sin Modif.			Modif. I			Modif. II		
ETAPA	NR	Entr.	Prueba	NR	Entr.	Prueba	NR	Entr.	Prueba
MULTISELECCIÓN	100	91.35	92.31	111	95.19	90.38	141	100	85.58
	112	91.35	92.31	128	96.15	87.50	171	98.08	87.50
	130	90.38	89.42	125	96.15	**94.23**	172	98.08	87.50
AJUSTE	100	98.08	89.42	111	96.15	88.46	141	100	88.46
	112	97.11	87.50	128	96.15	86.54	171	98.08	84.61
	130	95.19	93.27	125	96.15	87.50	172	99.04	86.54

Tabla 4.66. Resultados postprocesamiento con el conjunto de variables 13 y el MRD Clásico (f_0)

QuasiOWA $f_8, 0, 0.5, 20$	Sin Modif.			Modif. I			Modif. II		
ETAPA	NR	Entr.	Prueba	NR	Entr.	Prueba	NR	Entr.	Prueba
MULTISELECCIÓN	94	92.31	92.31	111	96.15	90.38	172	100	89.42
	103	92.31	93.27	118	96.15	85.58	170	100	85.58
	124	92.31	93.27	130	96.15	93.27	173	99.04	89.42
AJUSTE	94	97.11	90.38	111	97.11	84.61	172	100	87.50
	103	97.11	85.58	118	96.15	86.54	170	100	85.58
	124	98.08	92.31	130	97.11	88.46	173	99.04	87.50

Tabla 4.67. Resultados postprocesamiento con el conjunto de variables 13 y el MRD QuasiOWA ($f_8, 0, 0.5, 20$)

4.4.3.2 AGE II (Cruce Multipunto con Reparación)

Variables seleccionadas (Cto. 14): 7, 10, 15, 17, 20, 23, 24, 26, 29, 36, 39, 44, 45, 52, 56

Algoritmo	Prueba
1-NN	93.27
C4.5	81.7

Tabla 4.68. Resultados obtenidos con el conjunto de variables 14

GENERACIÓN		Sin pond.		Con pond.	
MRD	NR	Entr.	Prueba	Entr.	Prueba
Clásico (f_0)	241	96.15	80.77	—	—
Suma (f_1)		80.77	69.23	93.27	76.92
M. Arit. (f_2)		86.54	79.81	88.46	81.73
M. Quasiarit. (f_3, 10)		95.19	81.73	95.19	80.77
M. Quasiarit. (f_3, 20)		95.19	80.77	96.15	79.81
SOWA OR (f_5, 0.3)		90.38	82.69	94.23	81.73
SOWA OR (f_5, 0.5)		93.27	80.77	95.19	80.77
SOWA OR (f_5, 0.7)		95.19	80.77	95.19	80.77
SOWA OR (f_5, 0.8)		95.19	79.81	95.19	79.81
SOWA OR (f_5, 0.9)		96.15	80.77	96.15	80.77
Badd (f_6, 10)		95.19	81.73	95.19	80.77
Badd (f_6, 20)		95.19	80.77	96.15	80.77
OWA (f_7, 0, 0.3)		91.35	82.69	92.31	83.65
OWA (f_7, 0, 0.5)		88.46	80.77	91.35	83.65
OWA (f_7, 0, 0.8)		87.50	81.73	88.46	82.69
QuasiOWA (f_8, 0, 0.3, 10)		95.19	80.77	—	—
QuasiOWA (f_8, 0, 0.5, 10)		95.19	81.73	—	—
QuasiOWA (f_8, 0, 0.8, 10)		95.19	81.73	—	—
QuasiOWA (f_8, 0, 0.3, 20)		95.19	80.77	—	—
QuasiOWA (f_8, 0, 0.5, 20)		95.19	80.77	—	—
QuasiOWA (f_8, 0, 0.8, 20)		95.19	80.77	—	—

Tabla 4.69. Resultados generación con el conjunto de variables 14

SOWA OR $f_5, 0.3$	Sin Modif.			Modif. I			Modif. II		
ETAPA	NR	Entr.	Prueba	NR	Entr.	Prueba	NR	Entr.	Prueba
MULTISELECCIÓN	129	97.11	80.77	176	100	83.65	237	100	81.73
	129	97.11	80.77	171	99.04	79.81	212	99.04	79.81
	144	97.11	82.69	186	99.04	81.73	234	100	80.77
AJUSTE	129	98.08	79.81	176	100	**88.46**	237	100	80.77
	129	99.04	79.81	171	100	79.81	212	100	77.88
	144	99.04	84.61	186	99.04	82.69	234	100	84.61

Tabla 4.70. Resultados postprocesamiento con el conjunto de variables 14 y el MRD SOWA OR-Like ($f_5, 0.3$)

OWA P. $f_7, 0, 0.5$	Sin Modif.			Modif. I			Modif. II		
ETAPA	NR	Entr.	Prueba	NR	Entr.	Prueba	NR	Entr.	Prueba
MULTISELECCIÓN	153	96.15	83.65	172	99.04	81.73	237	100	84.61
	159	96.15	85.58	168	99.04	86.54	237	100	82.69
	143	96.15	84.61	192	98.08	80.77	238	100	76.92
AJUSTE	153	99.04	82.69	172	99.04	80.77	237	100	80.77
	159	99.04	84.61	168	100	**88.46**	237	100	84.61
	143	100	84.61	192	99.04	80.77	238	100	80.77

Tabla 4.71. Resultados postprocesamiento con el conjunto de variables 14 y el MRD OWA Ponderado ($f_7, 0, 0.5$)

Variables seleccionadas (Cto. 15): 0, 3, 8, 9, 17, 21, 22, 35, 37, 41, 44, 49, 54, 55, 57

Algoritmo	Prueba
1-NN	90.38
C4.5	76.9

Tabla 4.72. Resultados obtenidos con el conjunto de variables 15

GENERACIÓN		Sin pond.		Con pond.	
MRD	NR	Entr.	Prueba	Entr.	Prueba
Clásico (f_0)	144	82.69	82.69	—	—
Suma (f_1)		85.58	77.88	87.50	87.50
M. Arit. (f_2)		86.54	85.58	87.50	89.42
M. Quasiarit. (f_3, 10)		82.69	84.61	82.69	82.69
M. Quasiarit. (f_3, 20)		82.69	82.69	82.69	81.73
SOWA OR (f_5, 0.3)		85.58	83.65	85.58	84.61
SOWA OR (f_5, 0.5)		86.54	83.65	83.65	83.65
SOWA OR (f_5, 0.7)		83.65	84.61	82.69	84.61
SOWA OR (f_5, 0.8)		82.69	84.61	82.69	83.65
SOWA OR (f_5, 0.9)		82.69	83.65	82.69	81.73
Badd (f_6, 10)		82.69	82.69	82.69	82.69
Badd (f_6, 20)		82.69	82.69	82.69	82.69
OWA (f_7, 0, 0.3)		86.54	87.50	87.50	87.50
OWA (f_7, 0, 0.5)		85.58	89.42	87.50	91.35
OWA (f_7, 0, 0.8)		87.50	87.50	87.50	88.46
QuasiOWA (f_8, 0, 0.3, 10)		82.69	85.58	—	—
QuasiOWA (f_8, 0, 0.5, 10)		82.69	84.61	—	—
QuasiOWA (f_8, 0, 0.8, 10)		82.69	84.61	—	—
QuasiOWA (f_8, 0, 0.3, 20)		82.69	83.65	—	—
QuasiOWA (f_8, 0, 0.5, 20)		82.69	82.69	—	—
QuasiOWA (f_8, 0, 0.8, 20)		82.69	82.69	—	—

Tabla 4.73. Resultados generación con el conjunto de variables 15

M. Aritm. P. f_2	Sin Modif.			Modif. I			Modif. II		
ETAPA	NR	Entr.	Prueba	NR	Entr.	Prueba	NR	Entr.	Prueba
MULTISELECCIÓN	91	96.15	87.50	90	95.19	90.38	136	99.04	89.42
	90	94.23	89.42	95	97.11	86.54	143	96.15	90.38
	80	96.15	88.46	106	95.19	84.61	142	98.08	84.61
AJUSTE	91	97.11	87.50	90	98.08	90.38	136	99.04	87.50
	90	97.11	89.42	95	98.08	86.54	143	96.15	90.38
	80	96.15	91.35	106	97.11	89.42	142	98.08	87.50

Tabla 4.74. Resultados postprocesamiento con el conjunto de variables 15 y el MRD Media Aritmética Ponderada (f_2)

OWA P. $f_7, 0, 0.5$	Sin Modif.			Modif. I			Modif. II		
ETAPA	NR	Entr.	Prueba	NR	Entr.	Prueba	NR	Entr.	Prueba
MULTISELECCIÓN	79	96.15	86.54	72	98.08	88.46	126	96.15	89.42
	93	95.19	88.46	95	98.08	89.42	77	98.08	91.35
	86	95.19	85.58	94	97.11	89.42	140	95.19	84.61
AJUSTE	79	97.11	90.38	72	98.08	88.46	126	96.15	88.46
	93	96.15	87.50	95	98.08	86.54	77	98.08	87.50
	86	97.11	90.38	94	99.04	**92.31**	140	95.19	88.46

Tabla 4.75. Resultados postprocesamiento con el conjunto de variables 15 y el MRD OWA Ponderado ($f_7, 0, 0.5$)

Comentarios Finales

Resultados Obtenidos y Conclusiones

En esta memoria hemos presentado un proceso evolutivo de aprendizaje de SCBRDs. Para ello, hemos desarrollado procesos evolutivos para los distintos aspectos de diseño de un SCBRD, capaces de extraer el conocimiento necesario de información numérica, incorporando el conocimiento experto disponible y representándolo y utilizándolo de forma descriptiva.

Los resultados obtenidos y algunas conclusiones sobre los mismos se resumen en los siguientes apartados:

1. Métodos de Razonamiento Difuso en SCBRDs

Los Métodos de Razonamiento Difuso combinan la información presente en las reglas para, frente a un nuevo ejemplo, tomar una decisión sobre la clase a la cual pertenece. Su definición es, por tanto, un aspecto determinante de la precisión del SCBRD.

En esta memoria hemos analizado el MRD clásico y hemos propuesto un modelo general de inferencia dentro del cual se incluye como caso particular. Se han desarrollado propuestas de MRDs alternativos bajo dos modelos de inferencia distintos [CdJH97a, CdJH98a, CdJH98c, CdJH98d, CdJH99b]:

1. El primero utiliza toda la información disponible, de forma que todas las reglas disparadas por un ejemplo participan en la clasificación del mismo.

2. El segundo integra parte de la información, seleccionando las reglas disparadas por el ejemplo que serán utilizadas en el proceso de clasificación. Las reglas con un grado de asociación bajo con el ejemplo no intervienen en el proceso de inferencia. Esta selección se realiza a través de operadores de la familia OWA bajo el concepto de mayoría difusa.

Hemos mostrado de forma experimental, con distintos conjuntos de ejemplos, distintos tipos de reglas difusas y distintos métodos de aprendizaje utilizados en la literatura especializada, que los MRDs alternativos dotan al SCBRD de una mayor capacidad de

generalización. Podemos concluir que los algoritmos de clasificación mejoran con la utilización de mecanismos de combinación de reglas representados en MRDs alternativos.

2. Procesos evolutivos de aprendizaje de SCBRDs

Hemos propuesto un método multietapa de aprendizaje evolutivo de SCBRDs basado en la metodología de aprendizaje genético MOGUL (metodología para la obtención genética de sistemas basados en reglas difusas bajo el enfoque de aprendizaje iterativo de reglas) [CdJHL97, CdJHL98, CdJHL99].

El proceso desarrollado [CdJH97b, CdJH98b] integra el MRD en la obtención de una BC que coopere con el MRD utilizado en la clasificación y, además, determina el mejor conjunto de modificadores lingüísticos para los términos de las variables lingüísticas. De acuerdo con las líneas determinadas por la metodología MOGUL, el proceso contiene los siguientes pasos:

- El primer paso genera una base de reglas de forma independiente al MRD utilizado. El diseñador del sistema determina las particiones difusas para las variables lingüísticas.
- En la segunda etapa, un proceso basado en algoritmos genéticos con nichos selecciona el mejor subconjunto de reglas difusas y aprende el mejor conjunto de modificadores para las variables lingüísticas en cooperación con el MRD.
- Finalmente, se realiza un proceso de ajuste genético de las particiones difusas con los modificadores lingüísticos aprendidos en la etapa anterior.

Hemos mostrado, para distintos conjuntos de ejemplos, que el proceso de aprendizaje propuesto obtiene una BC que, en cooperación con el MRD, presenta mejor comportamiento que las generadas por otros procesos de aprendizaje inductivo conocidos. La comparación de los resultados obtenidos con los alcanzados con otros algoritmos de extracción de reglas que no consideran el MRD en el proceso de aprendizaje nos permite concluir que la integración del MRD en el proceso de generación de reglas difusas permite la obtención de un SCBRD más preciso. Es fundamental, sobre todo, la intervención del MRD en las etapas de selección de reglas y ajuste de particiones, que determinan el grado de solapamiento de las reglas difusas de la BR.

3. Selección de características evolutiva para el diseño de SCBRDs

Hemos propuesto un AGE para la selección de conjuntos de características con un número fijo de variables, con un esquema específico de reproducción y dos operadores de cruce con propiedades distintas.

Para mostrar la validez de las variables seleccionadas, la etapa de validación se ha realizado con tres procesos de aprendizaje inductivo distintos:

1. el vecino más cercano,
2. C4.5, y
3. el método evolutivo de aprendizaje de SCBRDs descrito en el capítulo 3.

Los resultados obtenidos muestran que, con independencia del proceso de aprendizaje inductivo utilizado posteriormente para el diseño del Sistema de Clasificación, los métodos propuestos obtienen conjuntos de variables que permiten disminuir el costo de aprendizaje del sistema, incrementar su simplicidad y descripción lingüística (siempre que el proceso de aprendizaje sea descriptivo) y aumentar la precisión en la predicción. Además, la capacidad de búsqueda de conjuntos optimales de los algoritmos propuestos no está limitada por la imposición inicial de una cardinalidad concreta en los conjuntos ya que sus resultados superan a los obtenidos por métodos de selección de características que determinan subconjuntos con cardinalidad mínima.

Trabajos Futuros

Describimos aquí algunas líneas de trabajo futuro como continuación de la presente memoria, que se pueden agrupar en los siguientes apartados:

- **Métodos de Razonamiento Difuso**. En ese aspecto es necesario realizar algunos estudios adicionales para analizar todas las posibilidades de los nuevos MRDs en la obtención de una mayor capacidad de generalización del SCBRD. Estos estudios estarán asociados con:
  - el uso de distintas funciones de ponderación,
  - el diseño de un criterio de selección para eliminar reglas disparadas con un grado de asociación bajo,
  - el uso de métodos alternativos para la obtención de los pesos de los operadores de la familia OWA, y
  - la optimización de los MRDs basados en el operador SOWA Or-Like y And-Like y de la media quasiaritmética con nuevos métodos de obtención de sus parámetros.
- **Proceso evolutivo de aprendizaje del SCBRD**. Los apartados del proceso de diseño relacionados con la extracción del conocimiento pueden completarse con:
  - La extensión del proceso multietapa de aprendizaje al diseño de un nuevo tipo de SCBRDs, los SCBRDs aproximativos, en los cuales el antecendente de las reglas difusas tiene naturaleza aproximativa, para incrementar el nivel de precisión alcanzable, aunque a costa de disminuir el nivel de descripción lingüística.

  - La extensión del proceso al diseño de SCBRDs con reglas difusas en forma normal disyuntiva en la que cada variable lingüística puede tener distintos valores asociados en la misma regla.
  - La inclusión en la etapa de multiselección de un proceso de selección de modificadores parametrizables que permitan realizar un ajuste local más fino de las funciones de pertenencia utilizadas.
  - El reajuste, tanto en la etapa de multiselección como en la de ajuste, del grado de certeza para cada regla, de forma que se represente de forma más fiel la información sobre la nueva zona del espacio representada por la regla tras la asignación de un modificador lingüístico o el ajuste de los parámetros de la función de pertenencia.

- **Proceso evolutivo de selección de características**. En el campo de la selección de características nuestros trabajos futuros estarán encaminados a:
  - El diseño de algoritmos de Selección de Características que incorporen, además de la medida de precisión proporcionada por un algoritmo de aprendizaje eficiente, una medida de separabilidad de clases o combinación de ellas que asegure que el conjunto de variables obtenido tendrá un buen comportamiento con otros algoritmos de aprendizaje inductivo distintos.
  - La incorporación en los esquemas propuestos de un método de optimización de los mejores individuos con la filosofía de los algoritmos de selección de características filtro.

Bibliografía

[AB96] Aha D. W. y Bankert R. L. (1996) A comparative evaluation of sequential feature selection algorithms. En Fisher D. y Lenz H.-J. (Eds.) *Learning from Data*, Lecture Notes in Statistics, páginas 199–206. Springer, New York.

[AD91] Almuallim H. y Dietterich T. G. (1991) Learning with many irrelevant features. En *Proc. of the Ningh National Conference on Artificial Intelligence (AAAI'91)*, páginas 547–552. MIT Press, Cambridge.

[AL95] Abe S. y Lan M.-S. (1995) A method for fuzzy rules extraction directly from numerical data and its application to pattern classification. *IEEE Transactions on Fuzzy Systems* 3(1): 18–28.

[Alm94] Almuallim H. (Noviembre 1994) Learning boolean concepts in the presence of many irrelevant features. *Artificial Intelligence* 69(1-2): 279–305.

[AT97] Abe S. y Thawonmas R. (1997) A fuzzy classifier with ellipsoidal regions. *IEEE Transactions on fuzzy systems* 5(3): 358–368.

[ATV83] Alsina C., Trillas E., y Valverde L. (1983) On some logical connectives for fuzzy set theory. *J. Math. Anal. Appl.* 93: 149–163.

[Bac96] Back T. (1996) *Evolutionary Algorithms in Theory and Practice.* Oxford University Press.

[Bak87] Baker J. E. (1987) Reducing bias and inefficiency in the selection algorithms. En *Proc. of the Second International Conference on Genetic Algorithms (ICGA'87)*, páginas 14–21. Hillsdale.

[Bat94] Battiti R. (Julio 1994) Using mutual information for selecting features in supervised neural net learning. *IEEE Transactions on Neural Networks* 5(4): 537–550.

[BB82] Ben-Bassat M. (1982) Pattern recognition and reduction of dimensionality. En Krishnaiah P. y Kanal L. (Eds.) *Handbook of Statistics*, volumen II, páginas 773–791. North Holland.

[BB96] Brassard G. y Bratley P. (1996) *Fundamentals of Algorithms.* Prentice Hall, New Jersey.

[BB97] Bonarini A. y Basso F. (1997) Learning to compose fuzzy behaviours for autonomous agents. *International Journal of Approximate Reasoning* 17(4): 433–446.

[BBM92] Brill F. Z., Brown D. E., y Martin W. N. (Marzo 1992) Fast genetic selection of features for neural network classifiers. *IEEE Transactions on Neural Networks* 3(2): 324–328.

[BBM93] Beasly D., Bull D. R., y Martin R. R. (1993) A sequential niche technique for multimodal function optimization. *Evolutionary Computation* 1: 101–125.

[BC94] Battiti R. y Colla A. M. (1994) Democracy in neural nets: Voting schemes for classification. *Neural Networks* 7: 691–703.

[BD95] Bardóssy A. y Duckstein L. (1995) *Modeling with applications to geophysical, biological and engineering systems.* Systems Engineering Series. CRC Press.

[BDeJH+97] Bala J., De Jong K., Huang J., Vafaie H., y Wechsler H. (1997) Using learning to facilitate the evolution of features for recognizing visual concepts. *Evolutionary Computation* 4(3): 297–311.

[BHS97] Bäck T., Hammel U., y Schewefel H. P. (1997) Evolutionary computation: Comments on the history and current state. *IEEE Transactions on Evolutionary Computation* 1(1): 3–17.

[BL97] Blum A. L. y Langley P. (1997) Selection of relevant features and examples in machine learning. *Artificial Intelligence* 97: 245–271.

[BS92] Benediktsson J. A. y Swain P. H. (1992) Consensus theoretic classification methods. *IEEE Transactions on Systems, Man, and Cybernetics* 22: 688–704.

[BS93] Bäck T. y Schewefel H. P. (1993) An overview of evolutionary algorithms for parameter optimization. *Evolutionary Computation* 1(1): 1–23.

[BS95] Bäck T. y Schwefel H. P. (1995) Evolution strategies I: variants and their computational implementations. En Periaux J., Winter G., Galán M., y Cuesta P. (Eds.) *Genetic Algorithms in Engineering and Computer Science*, páginas 111–126. John Wiley.

[Car93] Cardie C. (1993) Using decision trees to improve case-based learning. En *Proc. of the Tenth International Conference on Machine Learning (ICML'93)*, páginas 25–32. Morgan Kaufmann, San Mateo, CA, Amherst, MA.

[CdJH97a] Cordón O., del Jesus M. J., y Herrera F. (1997) Nuevos métodos de razonamiento en sistemas de clasificación basados en reglas difusas. En *Actas del VII Congreso Español Sobre Tecnologías y Lógica Fuzzy (ESTYLF'97)*, páginas 109–114.

[CdJH97b] Cordón O., del Jesus M. J., y Herrera F. (1997) Selecting fuzzy rule based classification systems with specific reasoning methods using genetic algorithms. En *Proc. of the 7th International Fuzzy Systems Association World Congress (IFSA'97)*, páginas 424–429. Praga.

[CdJH98a] Cordón O., del Jesus M. J., y Herrera F. (1998) Analyzing the reasoning mechanisms in fuzzy rule based classification systems. *Mathware & Soft Computing* 5(2-3): 321–332.

[CdJH98b] Cordón O., del Jesus M. J., y Herrera F. (1998) Genetic learning of fuzzy rule-based classification systems co-operating with fuzzy reasoning methods. *International Journal of Intelligent Systems* 13(10/11): 1025–1053.

[CdJH98c] Cordón O., del Jesus M. J., y Herrera F. (1998) Métodos de razonamiento aproximado basados en el concepto de mayoría difusa para sistemas de clasificación. En *Actas del VIII Congreso Español sobre Tecnologías y Lógica Fuzzy (ESTYLF'98)*, páginas 399–404. Pamplona.

[CdJH98d] Cordón O., del Jesus M. J., y Herrera F. (1998) Reasoning methods based on OWA operators under fuzzy majority in fuzzy rule-based classification sytems. Technical Report 98121, Dept. Ciencias de la Computación e Inteligencia Artificial, Universidad de Granada.

[CdJH99a] Cordón O., del Jesus M. J., y Herrera F. (1999) Evolutionary approaches to the learning of fuzzy rule based classification systems. En Jain L. C. (Ed.) *Evolution of Engineering and Information Systems and Their Applications*, International Series on Computational Intelligence, páginas 105–160. CRC Press.

[CdJH99b] Cordón O., del Jesus M. J., y Herrera F. (1999) A proposal on reasoning methods in fuzzy rule-based classification systems. *International Journal of Approximate Reasoning* 20(1): 21–45.

[CdJHL97] Cordón O., del Jesus M. J., Herrera F., y Lozano M. (Septiembre 1997) An evolutionary paradigm for designing fuzzy rule-based systems from examples. En *Proc. of the 2nd IEE/IEEE International Conference on Genetic Algorithms and Engineering Systems: Innovations and Applications (GALESIA'97)*, páginas 139–144. Glasgow.

[CdJHL98] Cordón O., del Jesus M. J., Herrera F., y Lozano M. (1998) Modelado cualitativo utilizando una metodología evolutiva de aprendizaje iterativo de bases de reglas difusas. *Inteligencia Artificial* 5: 56–61.

[CdJHL99] Cordón O., del Jesus M. J., Herrera F., y Lozano M. (1999) MOGUL: A methodology to obtain genetic fuzzy rule-based systems under the iterative rule learning approach. *International Journal of Intelligent Systems* 14(11).

[CF94] Caruana R. A. y Freitag D. (1994) Greedy attribute selection. En Kaufmann M. (Ed.) *Proc. of the Eleventh International Conference on Machine Learning (ICML'94)*, páginas 28–36.

[CFM96] Carse B., Fogarty T. C., y Munro A. (1996) Evolving fuzzy rule based controllers using genetic algorithms. *Fuzzy Sets and Systems* 80: 273–293.

[CGHP99] Cordón O., González A., Herrera F., y Pérez R. (1999) Encouraging cooperation in the genetic iterative rule learning approach for qualitative modeling. En Kacprzyk J. y Zadeh L. (Eds.) *Computing with Words in Information / Intelligent Systems*, páginas 95–117. Physical-Verlag, Heidelberg.

[CH69] Cover T. M. y Hart R. E. (1969) Nearest neighbour pattern classification. *IEEE Transactions on Information Theory* IT-13(1): 21–27.

[CH97] Cordón O. y Herrera F. (1997) A three-stage evolutionary process for learning descriptive and approximative fuzzy logic controller knowledge bases from examples. *International Journal of Approximate Reasoning* 17: 369–407.

[CH99] Cordón O. y Herrera F. (1999) Hybridizing genetic algorithms with sharing scheme and evolution strategies for designing fuzzy rule-based systems. *Fuzzy Sets and Systems* Por aparecer.

[Cho95] Cho S. (1995) Fuzzy aggregation of modular neural networks with ordered weighted averaging operators. *International Journal of Approximate Reasoning* 12: 358–375.

[CHP97] Cordón O., Herrera F., y Peregrín A. (1997) Aplicability of the fuzzy operators in the design of fuzzy logic controllers. *Fuzzy Sets and Systems* 86: 15–41.

[CN86] Clark P. y Niblett T. (1986) Learning if then rules in noisy domains. Technical Report TIRM 86-019, The Turing Institute, Glasgow.

[CS94] Cross V. y Sundkamp R. (1994) Patterns of fuzzy rule-based inference. *International Journal of Approximate Reasoning* 11: 151–183.

[CWY95] Chi Z., Wu J., y Yan H. (1995) Handwritten numeral recognition using self-organizing maps and fuzzy rules. *Pattern Recognition* 28(1): 59–66.

[CY96] Chi Z. y Yan H. (1996) ID3-Derived fuzzy rules and optimized defuzzification for handwritten numeral recognition. *IEEE Transactions on Fuzzy Systems* 4(1): 24–31.

[CYP96] Chi Z., Yan H., y Pham T. (1996) *Fuzzy algorithms with applications to image processing and pattern recognition.* World Scientific.

[DeJ75] De Jong K. A. (1975) *An Analysis of the Behavior of a Class of Genetic Adaptive Systems*. PhD thesis, University of Michigan, Ann Arbor.

[DeJS93] De Jong K. A. y Sarma J. (1993) Generation gaps revisited. En Whitley L. D. (Ed.) *Foundations of Genetic Algorithms 2*, páginas 19–28. Morgan Kaufmann.

[DeJSG93] De Jong K. A., Spears W. M., y Gordon D. F. (1993) Using genetic algorithms for concept learning. *Machine Learning* 13: 161–188.

[DG89] Deb K. y Goldberg D. E. (1989) An investigation of niche and species formation in genetic function optimization. En *Proc. of the Third International Conference on Genetic Algorithms (ICGA'89)*, páginas 42–50. Hillsdale.

[DK82] Devijver P. A. y Kittler J. (1982) *Pattern Recognition: A Statistical Approach*. Prentice-Hall.

[DL97] Dash M. y Liu H. (1997) Feature selection for classification. *Intelligent Data Analysis* 1(3): 1–29.

[Doa92] Doak J. (1992) An evaluation of feature-selection methods and their application to computer security. Technical report, Departament of Computer Science, University of California.

[DP84] Dyckhoff H. y Pedrycz W. (1984) Generalized means as model of compensative connectives. *Fuzzy Sets and Systems* 14: 143–154.

[DP85] Dubois D. y Prade H. (1985) A review of fuzzy set aggregation connectives. *Information Sciences* 36: 85–121.

[DP92] Dubois D. y Prade H. (1992) Upper and lower images of a fuzzy set induced by a fuzzy relation: Applications ot fuzzy inference and diagnosis. *Information Sciences* 64: 203–232.

[ECS89] Eshelman L. J., Caruana R. A., y Schaffer J. D. (1989) Biases on crossover landscape. En *Proc. of the Third International Conference on Genetic Algorithms (ICGA'89)*, páginas 10–19. Morgan Kaufmann, San Mateo, CA.

[ERR94] Eiben A. E., Raué P. E., y Ruttkay Z. (1994) Genetic algorithms with multiparent recombination. En *Proc. of the Third International Conference on Parallel Problem Solving from Nature- PPSN III, L.N.C.S. 866*, páginas 78–87. Springer, Berlin.

[Fal98] Falkenauer E. (1998) *Genetic Algorithms and Grouping Problems*. Wiley and Sons.

[Fis36] Fisher R. A. (1936) The use of multiple measurements in taxonomic problems. *Annual Eugenics* 7(Part II): 179–188.

[FMR95] Fodor J., Marichal J. L., y Roubens M. (1995) Characterization of the ordered weighted averaging operators. *IEEE Transactions on Fuzzy Systems* 3(2): 236–239.

[FN95] Fussell S. J. y Norvig P. (1995) *Artificial Intelligence: A Modern Approach.* Prentice Hall.

[Fog89] Fogarty T. C. (1989) Varying the probability of mutation in the genetic algorithm. En Schaffer J. D. (Ed.) *Proc. of the Third International Conference on Genetic Algorithms (ICGA'89)*, páginas 104–109. Morgan Kaufmann Publishers, San Mateo.

[Fog91] Fogel D. B. (1991) *System Identification trough Simulated Evolution. A Machine Learning Approach.* Ginn Press.

[FOW66] Fogel L. J., Owens A. J., y Walsh M. J. (1966) *Artificial Intelligence through Simulated Evolution.* John Willey and Sons.

[GH97] González A. y Herrera F. (1997) Multi-stage genetic fuzzy systems based on the iterative rule learning approach. *Mathware and Soft Computing* 4(3): 233–249.

[GLF89] Gennari J. H., Langley P., y Fisher D. (1989) Models of incremental concept formation. *Artificial Intelligence* 40: 11–61.

[Gol89] Goldberg D. E. (1989) *Genetic Algorithms in Search, Optimization, and Machine Learning.* Addison-Wesley.

[GP98] González A. y Pérez R. (1998) Completeness and consistency conditions for learning fuzzy rules. *Fuzzy Sets and Systems* 96: 37–51.

[GPV94] González A., Pérez R., y Verdegay J. L. (1994) Learning the structure of a fuzzy rule: A genetic approach. *Fuzzy Systems and Artificial Intelligence* 3: 57–70.

[Gre88] Grefenstette J. J. (1988) Credit assignment in rule discovery systems based on genetic algorithms. *Machine Learning* 3: 225–245.

[GS88] Gorman R. P. y Sejnowski T. J. (1988) Analysis of hidden units in a layered network trained to classify sonar targets. *Neural Networks* 1: 75–89.

[HL96] Hong T.-P. y Lee C.-Y. (1996) Induction of fuzzy rules and membership functions from training examples. *Fuzzy Sets and Systems* 84: 33–47.

[HLV95] Herrera F., Lozano M., y Verdegay J. L. (1995) Tuning fuzzy controllers by genetic algorithms. *International Journal of Approximate Reasoning* 12: 299–315.

[HLV96] Herrera F., Lozano M., y Verdegay J. L. (1996) Dynamic and heuristic fuzzy connectives-based crossover operators for controlling the diversity and convergence of real-coded genetic algorithms. *International Journal of Intelligent Systems* 11: 1013–1040.

[HLV98a] Herrera F., Lozano M., y Verdegay J. L. (1998) A learning process for fuzzy control rules using genetic algorithms. *Fuzzy Sets and Systems* 100: 143–158.

[HLV98b] Herrera F., Lozano M., y Verdegay J. L. (1998) Tackling real-coded genetic algorithms: Operators and tools for the behavioural analysis. *Artificial Intelligence Rev.* 12: 265–319.

[Hol75] Holland J. H. (1975) *Adaptation in Natural and Artificial Systems.* Universtity of Michigan Press, Ann Arbor, MI.

[Hol85] Holland J. H. (1985) Properties of the bucket brigade algorithm. En *Proc. of the First International Conference on Genetic Algorithms (ICGA'85)*, páginas 1–7.

[Hol86] Holland J. H. (1986) Escaping brittleness: The possibilities of general-purpose learning algorithms applied to parrallel rule-based systems. En Michalski R. S., Carbonell J., y Mitchell T. (Eds.) *Machine Learning II.* Morgan Kaufmann.

[HP97] Hoffman F. y Pfister G. (1997) Evolutionary design of a fuzzy knowledge base for a mobile robot. *International Journal of Approximate Reasoning* 17(4): 447–469.

[HR78] Holland J. H. y Reitman J. S. (1978) Cognitive systems based on adaptive algorithms. En Waterman D. y Hayes-Roth F. (Eds.) *Pattern-Directed Inference Systems*, páginas 313–329. Academic Press.

[IM97] Ishibuchi H. y Murata T. (1997) A genetic-algorithm-based fuzzy partition method for pattern classification problems. En Herrera F. y Verdegay J. (Eds.) *Genetic algorithms and soft computing*, páginas 555–578. Physica-Verlag.

[IMN96] Ishibuchi H., Morisawa T., y Nakashima T. (1996) Voting schemes for fuzzy-rule-based classification systems. En *Proc. of the Fith International IEEE Conference on Fuzzy Systems*, páginas 614–620.

[IMT97] Ishibuchi H., Murata T., y Turksen I. B. (1997) Single-objective and two-objective genetic algorithms for selecting linguistic rules for pattern classification problems. *Fuzzy Sets and Systems* 89: 135–150.

[INM95] Ishibuchi H., Nakashima T., y Murata T. (1995) A fuzzy classifier system that generates fuzzy if-then rules for pattern classification problems. En *Proc. of the Second International IEEE Conference on Fuzzy Systems*, páginas 759–764.

[INT92] Ishibuchi H., Nozaki K., y Tanaka H. (1992) Distributed representation of fuzzy rules and its application to pattern classification. *Fuzzy Sets and Systems* 52: 21–32.

[INT93] Ishibuchi H., Nozaki D., y Tanaka H. (1993) Efficient fuzzy partition for pattern space for classification problems. *Fuzzy Sets and Systems* 59: 295–304.

[INT94] Ishibuchi H., Nozaki K., y Tanaka H. (1994) Construction of fuzzy classification systems with rectangular fuzzy rules using genetic algorithms. *Fuzzy Sets and Systems* 65: 237–253.

[INY93] Ishibuchi H., Nozaki K., y Yamamoto N. (1993) Selecting fuzzy rules by genetic algorithm for classification problems. En *Proc. of the International IEEE Conference on Fuzzy Systems*, páginas 1119–1124.

[INYT93] Ishibuchi H., Nozaki K., Yamamoto N., y Tanaka H. (1993) Genetic operations for rule selection in fuzzy classification systems. En *Proc. of the Fifth International Fuzzy Systems Association World Congress (IFSA'97)*, páginas 15–18.

[INYT95] Ishibuchi H., Nozaki K., Yamamoto N., y Tanaka H. (1995) Selecting fuzzy if-then rules for classification problems using genetic algorithms. *IEEE Transactions on Fuzzy Systems* 3(3): 260–270.

[Jan93] Janikow C. Z. (1993) A knowledge intensive genetic algorithm for supervised learning. *Machine Learning* 13: 198–228.

[JCC98] Junco L., Cano F., y Couso I. (1998) Clasificación y selección de características mediante algoritmos ga-p. *Inteligencia Artificial* 5: 50–55.

[JKP94] John G. H., Kohavi R., y Pfleger K. (1994) Irrelevant features and the subset selection problem. En *Proc. of the Eleventh International Conference on Machine Learning (ICML'94)*, páginas 121–129.

[Kac97] Kacprzyk J. (1997) OWA operators in machine learning from imperfect examples. En Yager R. R. y Kacprzyk F. (Eds.) *The Ordered Weighted Averaging Operators. Theory and Applications*, páginas 321–329. Kluwer Academic Publishers.

[Kar91] Karr C. (1991) Genetic algorithms for fuzzy controllers. *AI Expert* 6: 26–33.

[KJ97] Kohavi R. y John G. H. (1997) Wrappers for feature subset selection. *Artificial Intelligence* 97: 273–324.

[Koh97] Kohonen T. (1997) *Self-organizing maps.* 2nd edition. Springer-Verlag, Berlin.

[Kon94] Kononenko I. (1994) Estimating attributes: analysis and extensions of RELIEF. En *Proc. of the Fourth European Conference on Machine Learning (ICML'94)*, páginas 171–182.

[KR92a] Kira K. y Rendell L. A. (1992) The feature selection problem: Traditional methods and a new algorithm. En *Proc. of Ninth National Conference on Artificial Intelligence*, páginas 129–134.

[KR92b] Kira K. y Rendell L. A. (1992) A practical approach to feature selection. En *Proc. of the Ninth International Conference on Machine Learning (ICML'92)*, páginas 249–256. Morgan Kaufmann Publishers, Aberdeen, Scotland.

[Kun96] Kuncheva L. I. (1996) On the equivalence between fuzzy and statistical classifiers. *International Journal of Uncertainty, Fuzzyiness and Knowledge-Based Systems* 4(3): 245–253.

[Kun97] Kuncheva L. I. (1997) An application of WOA operators to the aggregation of multiple classification decisions. En Yager R. y Kacprzyk J. (Eds.) *The Ordered Weighted Averaging Operators. Theory and Applications*, páginas 330–343. Kluwer Academic Plublishers.

[Lan94] Langley P. (1994) Selection of relevant features in machine learning. En *Proc. of the AAAI Fall Symposium on Relevance*, páginas 1–5. AAAI Press, New Orleans.

[Lan97] Lanzi P. L. (1997) Fast feature selection with genetic algorithms: A filter approach. En *Proc. of IEEE International Conference on Evolutionary Computation (ICEC'97)*, páginas 537–540.

[LK89] Lucasius C. B. y Kateman G. (1989) Applications of genetic algorithms in chemometrics. En Shaffer D. (Ed.) *Proc. of the Third International Conference on Genetic Algorithms (ICGA'89)*, páginas 170–176. Morgan Kauffman Publishers, San Mateo.

[LM98] Liu H. y Motoda H. (1998) *Feature selection for knowledge discovery and data mining.* Kluwer Academic Publishers.

[LMD98] Liu H., Motoda H., y Dash M. (1998) A monotonic measure for optimal feature selection. En *Proc. of the European Conference on Machine Learning (ECML'98).* Springer-Verlag.

[LPG94] Lin S. C., Punch W. F., y Goodman E. D. (1994) Coarse-grain parallel genetic algorithms: categorization and new approach. En *Proc. of the International IEEE on Evolutionary Computation (ICEC'94)*, páginas 28–37.

[LS94a] Langley P. y Sage S. (1994) Induction of selective bayesian classifiers. En Morgan Kaufmann San Mateo C. (Ed.) *Proc. of the Tenth Conference on Uncertainty in Artificial Intelligence*, páginas 399–406. Seattle, WA.

[LS94b] Langley P. y Sage S. (1994) Oblivious decision trees and abstract cases. En Press A. (Ed.) *Working Notes of the AAAI-94 Workshop on Case-Based Reasoning*, páginas 113–117. Seattle, WA.

[LS96a] Liu H. y Setiono R. (1996) Feature selection and classification. A probabilistic wrapper approach. En *Proc. of the Ninth International Conference on Industrial and Engineering Applications of AI and ES*, páginas 419–424.

[LS96b] Liu H. y Setiono R. (1996) A probabilistic approach to feature selection. A filter solution. En Saitta L. (Ed.) *Proc. of International Conference on Machine Learning (ICML-96)*, páginas 319–327. Morgan Kaufmann Publishers.

[Lu96] Lu Y. (1996) Knowledge integration in a multiple classifier system. *Applied Intelligence* 6: 75–86.

[LWZ95] Liu W., Wang M., y Zhong Y. (1995) Selecting features with genetic algorithm in handwritten digits recognition. En *Proc. of the International IEEE Conference on Evolutionary Computation (ICEC'95)*, volumen 1, páginas 396–399.

[Mag97] Magdalena L. (1997) A fuzzy logic controller with learning through the evolution of its knowledge base. *International Journal of Approximate Reasoning* 16: 335–358.

[MBV98] Martín-Bautista M. J. y Vila M. A. (1998) Applying genetic algorithms to the feature selection problem in information retrieval. En *Proc. 1998 Workshop on Flexible Query-Answering Systems (FQAS'98)*, páginas 260–269.

[Mic83] Michalski R. S. (1983) A theory and methodology of inductive learning. En *Machine Learning: An artificial intelligence approach I*, páginas 83–134. Morgan Kaufmann.

[Mic86] Michalski R. S. (1986) Understanding the nature of learning. En *Machine Learning: An artificial intelligence approach II*. Morgan Kaufmann.

[Mic96] Michalewicz Z. (1996) *Genetic algorithms + data structures = evolution programs*. 3rd. edition. Springer-Verlag.

[Miz89a] Mizumoto M. (1989) Pictorial representations of fuzzy connectives, part I: Cases of t-norms, t-conorms and averaging operators. *Fuzzy Sets and Systems* 31: 217–242.

[Miz89b] Mizumoto M. (1989) Pictorial representations of fuzzy connectives, part II: Cases of compensatory operators and seft-dual operators. *Fuzzy Sets and Systems* 32: 45–79.

[ML94] Moore A. W. y Lee M. S. (1994) Efficient algorithms fo minimizing cross validation error. En Morgan Kaufmann San Mateo C. (Ed.) *Proc. of the Eleventh International Conference on Machine Learning (ICML'94)*, páginas 190–198. New Brunswick, NJ.

[MMP92] Mandal D. P., Murthy C. A., y Pal S. K. (1992) Formulation of a multivalued recognition system. *IEEE Transactions on Systems, Man, and Cybernetics* 22(4): 607–620.

[MMP94] Mandal D. P., Murthy C. A., y Pal S. K. (1994) Theoretical performance of a multivalued recognition system. *IEEE Transactions on Systems, Man, and Cybernetics* 24(7): 1001–1021.

[MP96] Mitra S. y Pal S. K. (1996) Fuzzy self-organization, inferencing, and rule generation. *IEEE Transactions on Systems, Man and Cybernetics* 26: 608–619.

[MPB95] Mitra S., Pal S. K., y Barnejee S. (1995) Tuning of class membership using genetic algorithms. En *Proc. of the European Conference on Fuzzy and Intelligent Technologies (EUFIT'95)*, páginas 1420–1424.

[MS90] McCallum R. A. y Spackman K. A. (1990) Using gentic algorithms to learn disjunctive rules from examples. En *Proc. of the Seventh International Conference on Machine Learning (ICML'90)*, páginas 149–153.

[MS98] Montseny E. y Sobrevilla P. (1998) Application of fuzzy techniques to the design of algorithms in computer vision. *Mathware & Soft Computing* 5(2-3): 223–230.

[MST94] Michie D., Spiegelhalter D. J., y Taylor C. C. (Eds.) (1994) *Machine Learning, Neural and Statistical Classification.* Ellis Horwood Limited.

[MZ82] Mizumoto M. y Zimmermann H.-J. (1982) Comparison of fuzzy reasoning methods. *Fuzzy Sets and Systems* 8: 253–283.

[NF77] Narendra P. M. y Fukunaga K. (september 1977) A branch and bound algorithm for feature selection. *IEEE Transactions on Computers* C-26(8): 917–922.

[NFUM96] Nakaoka K., Furuhashi T., Uchikawa Y., y Maeda H. (1996) A proposal on payofss and apportionment of credits of fuzzy classifier sytems- finding the knowledge for large scale systems. *Japanese Journal of Fuzzy Theory and Systems* 8(1): 25–34.

[NHW92] Nomura H., Hayashi I., y Wakami N. (1992) A self-tuning method of fuzzy reasoning by genetic algorithm. En *Proc. of the 1992 International Fuzzy Systems and Intelligent Control Conference*, páginas 236–245.

[NIT96] Nozaki K., Ishibuchi H., y Tanaka H. (1996) Adaptive fuzzy rule based classification sytems. *IEEE Transactions on Fuzzy Systems* 4: 238–250.

[NK97] Nauck D. y Kruse R. (1997) A neuro-fuzzy method to learn fuzzy classification rules from data. *Fuzzy Sets and Systems* 89: 277–288.

[PB93] Parodi A. y Bonelli P. (1993) A new approach to fuzzy classifier system. En *Proc. of the Fifth International Conference on Genetic Algorithms (ICGA'93)*, páginas 223–230.

[Ped90] Pedrycz W. (1990) Fuzzy sets in pattern recognition: Methodology and methods. *Pattern Recognition* 23(1/2): 121–146.

[Ped97a] Pedrycz W. (1997) Fuzzy sets in pattern recognition: Accomplishments and challenges. *Fuzzy Sets and Systems* 90: 171–176.

[Ped97b] Pedrycz W. (1997) OWA-Based computing: Learning algorithms. En Yager R. y Kacprzyk J. (Eds.) *The Orderer Weighted Averaging Operators. Theory and Applications*, páginas 309–320. Kluwer Academic Publishers.

[PF70] Patrick E. A. y Ficher III F. P. (1970) A generalized k-nearest neighbor rule. *Information and Control* 16: 128–152.

[PGP+93] Punch W. F., Goodman E. D., Pei M., Chia-Shun L., Hovland P., y Enbody R. (1993) Futher research on feature selection and classification using genetic algorithms. En *Proc. of the Fith International Conference on Genetic Algorithms (ICGA'93)*, páginas 557–564.

[PM92] Pal S. K. y Mandal D. P. (1992) Linguistic recognition system based on approximate reasoning. *Information Sciences* 61: 135–161.

[Qui86] Quinlan J. R. (1986) Induction of decision trees. *Machine Learning* 1(1): 81–106.

[Qui93] Quinlan J. R. (1993) *C4.5: Programs for Machine Learning.* Morgan Kaufmann.

[RG96] Ray K. S. y Ghoshal J. (1996) Approximate reasoning approach to pattern recognition. *Fuzzy Sets and Systems* 77: 125–150.

[RPG+97] Raymer M. L., Punch W. F., Goodman E. D., Sanschagrin P. C., y Kuhn L. A. (1997) Simultaneous feature scaling and selection using a genetic algorithm. En *Proc. of the Seventh International Conference on Genetic Algorithms (ICGA'97)*, páginas 561–567.

[RR97] Ralescu A. L. y Ralescu D. A. (1997) Extension of fuzzy aggregation. *Fuzzy Sets and Systems* 86: 321–330.

[Sch95] Schwefel H. P. (1995) *Evolution and Optimum Seeking. Sixth-Generation Computer Technology Series.* John Wiley and Sons.

[Ska94] Skalak D. B. (1994) Prototype and feature selection by sampling and random mutation hill-climbing algortihms. En Morgan Kaufmann San Mateo C. (Ed.) *Proc. of the Eleventh International Conference on Machine Learning (ICML'94)*, páginas 293–311. New Brunswick, NJ.

[Smi80] Smith S. F. (1980) *A learning system based on genetic algorithms.* PhD thesis, University of Pittsburg.

[SP95] Singh M. y Provan G. M. (1995) A comparision of induction algorithms for selective and non-selective bayesian classifiers. En Morgan Kaufmann San Mateo C. (Ed.) *Proc. of the 12th International Conference on Machine Learning (ICML'95)*, páginas 497–505. Lake Tahoe, CA.

[SS88] Siedlecki W. y Sklansky J. (1988) On automatic feature selection. *International Journal of Pattern Recognition and Artificial Intelligence* 2: 197–220.

[SS89] Siedlecki W. y Sklansky J. (1989) A note on genetic algorithms for large-scale feature selection. *Pattern Recognition Letters* 10: 335–347.

[SW49] Shannon C. E. y Weaver W. (1949) *The Mathematical Theory of Communication.* Urbana IL: Universtity of Illinois Press.

[Sys89] Syswerda G. (1989) Uniform crossover in genetic algorithms. En Schaffer J. D. (Ed.) *Proc. of the Third International Conference on Genetic Algorithms (ICGA'89).* Morgan Kaufmann.

[Sys91] Syswerda G. (1991) A study of reproduction in generational and steady-state genetic algorithms. En Rawlins G. (Ed.) *Foundations of Genetic Algorithms.* Morgan Kaufmann.

[TA96] Thawonmas R. y Abe S. (1996) Extraction of fuzzy rules for classification based on partitioned hyperboxes. *Journal of Intelligent and Fuzzy Systems* 4: 215–226.

[TT93] Turksen I. B. y Tian Y. (1993) Combination of rules or their consequences in fuzzy expert systems. *Fuzzy Sets and Systems* 58: 3–40.

[TWK94] Townsend-Weber T. y Kibler D. (1994) Instance-based prediction of continuous values. En Press A. (Ed.) *Working Notes of the AAAI-94 Workshop on Case-Based Reasoning*, páginas 30–35. Seattle, WA.

[UAL95] Uebele V., Abe S., y Lan M. S. (1995) A neural-network-based fuzzy classifier. *IEEE Transactions on Systems, Man, and Cybernetics* 25(2): 353–361.

[Vel98] Velasco J. R. (1998) Genetic-based on-line learning for fuzzy process control. *International Journal of Intelligent Systems* 13: 891–903.

[Ven93] Venturini G. (1993) SIA: A supervised inductive algorithm with genetic search for learning attributes based concepts. En *Lecture Notes in Artificial Intelligence: Machine Learning (ECML'93)*, páginas 280–296.

[VR91] Valenzuela-Rendón M. (1991) The fuzzy classifier system: motivations and first results. En *Proc. of Parallel Solving from Nature (PPSN II)*, páginas 330–334.

[Whi89] Whitley L. D. (1989) The genitor algorithm and selection pressure: Why rank-based allocation of reproductive trials is best. En Schaffer J. D. (Ed.) *Proc. of the Third International Conference on Genetic Algorithms (ICGA'89)*. Morgan Kaufmann.

[Wil94] Wilson S. W. (1994) ZCS: A zeroth order classifier system. *Evolutionary Computation* 2(1): 1–18.

[Wil95] Wilson S. W. (1995) Classifier fitness based on accuracy. *Evolutionary Computation* 3(2): 149–175.

[WK91] Weiss S. M. y Kulikowski C. A. (1991) *Computer Systems That Learn*. Morgan Kaufmann, San Mateo, CA.

[WM92] Wang L. X. y Mendel J. M. (1992) Generating fuzzy rules by learning from examples. *IEEE Transactions on Systems, Man, and Cybernetics* 25(2): 353–361.

[Yag88] Yager R. R. (1988) On ordered weighted averaging aggregation operators in multicriteria decisionmaking. *IEEE Transactions on Systems Man, and Cybernetics* 18: 183–190.

[Yag93] Yager R. R. (1993) Families of OWA operators. *Fuzzy Sets and Systems* 59: 125–148.

[YF94] Yager R. R. y Filev D. P. (1994) Parametrized and-like and or-like OWA operators. *International Journal General Systems* 22: 297–316.

[YH98] Yang J. y Honavar V. (1998) Feature subset selection using a genetic algorithm. *IEEE Intelligent Systems* 13: 44–49.

[YI96] Yagiura M. y Ibaraki T. (1996) Genetic and local search algorithms as robust and simple optimization tools. En Osman I. y Kelly J. P. (Eds.) *Meta-Heuristics: Theory and Applications*, páginas 63–82. Kluwer Academic Publishers.

[YZ96] Yuan Y. y Zhuang H. (1996) A genetic algorithm for generating fuzzy classification rules. *Fuzzy Sets and Systems* 84: 1–19.

[Zad65] Zadeh L. A. (Febrero 1965) Fuzzy sets. *Information and Control* 8: 338–353.

[Zad73] Zadeh L. A. (1973) Outline of a new approach to the analysis of complex systems and decision processes. *IEEE Transactions on Systems, Man, and Cybernetics* 3: 28–44.

[Zad75] Zadeh L. A. (1975) The concept of a linguistic variable and its applications to approximate reasoning, partes I-III. *Information Sciences* 8-9: 199–249; 301–357; 43–80.

[Zad77] Zadeh L. A. (1977) Fuzzy sets and their applications to classification and clustering. En Ryzin J. V. (Ed.) *Classification and Clustering*, páginas 251–299. Academic Press, New York.

[Zad83] Zadeh L. A. (1983) A computational approach to fuzzy quantifiers in natural languajes. *Comps. and Maths. with Appls.* 9: 149–184.