,

Contenido

Índice de Figuras

Índice de Tablas

Parte I

INTRODUCCIÓN GENERAL

Capítulo 1

Introducción general

1.1 Área de investigación

Los trabajos que han sido llevados a cabo en esta tesis se enmarcan en el área de la *clasificación supervisada* (CS). Con este nombre tan rimbombante se identifica una serie de métodos que tratan de determinar, de forma automática, la clasificación que se debe asignar a un caso dado. La clasificación puede consistir en determinar si una persona padece o no cierta enfermedad, la especie a la que pertenece un tipo de animal o el grado de pureza de un determinado mineral.

Para que el sistema pueda tomar una decisión –acertada o no– sobre la clasificación que va a asignar, es necesario indicarle las características que se han observado en el caso que se está estudiando actualmente. Estas características vienen determinadas por el valor observado en los datos o variables tenidos en consideración, denominadas *variables predictoras*, y el sistema debe discernir entre una serie de categorías a la hora de tomar su decisión sobre la clasificación a asignar. A los distintos valores que se tienen en cuenta se les denomina *clases*, y pueden variar en número y significado dependiendo del problema tratado. Las categorías podrían ser *Si* o *No*, en un problema en el que hay que decidir si una persona está enferma, *Perro*, *Gato* o *Caballo*, en un caso simplificado de determinación de la clase de animal de que se trata, etc.

En algunos casos, la clasificación correspondiente al caso que se está manejando es trivial, y el sistema automático se implementa de forma sencilla mediante un algoritmo determinista. En otros casos, no queda claro cuales son los criterios que se deben de seguir, ya que los entendidos en la materia no coinciden en las clasificaciones, o se equivocan en sus dictámenes. Ejemplos de este tipo de problemas aparecen en muchas situaciones de la vida real: en medicina no todos los médicos aciertan siempre, como no lo hacen los expertos en bolsa.

A la hora de automatizar el proceso de clasificación, existen diversas aproximaciones: existe un experto humano capaz de diseñar un sistema clasificador, o el clasificador se aprende automáticamente de alguna manera, o se hace una mezcla de ambas. Nos centraremos en lo que se ha dado en llamar aprendizaje automático, que consiste en aprender el modelo clasificatorio a utilizar en base a la información que se halla contenida en una base de datos histórica en la que se guardan los valores de las variables predictoras de casos tratados anteriormente, junto con la clase real a la que pertenecían los mismos. A este histórico se le denomina *base de datos de entrenamiento*. Habitualmente se guardan algunos casos para comprobar el funcionamiento del sistema en la llamada *base de datos de test*.

Existen muchas alternativas para construir automáticamente un clasificador dados unos datos recogidos en una base de datos –o en dos–. Nosotros hemos investigado varias alternativas existentes en la literatura, tratando de añadir a los paradigmas existentes alguna modificación que mejore su comportamiento clasificatorio en determinados problemas, o incluso se ha tratado de presentar en el área de la clasificación supervisada un método de clasificación conocido en otros ámbitos, pero novedoso éste, como es la utilización de las redes Bayesianas como clasificadores.

Sin ánimo de formalizar, decir que una red Bayesiana es un formalismo matemático formado por un grafo y una serie de parámetros asociados al mismo, mediante la que se representan las relaciones entre las variables que forman el dominio estudiado. Se puede realizar razonamientos con las mismas asignando una serie de valores a las variables tenidas en consideración –a esto se le denomina instanciación de variables o introducción de la evidencia– y viendo el efecto que estos valores producen sobre los posibles valores del resto de variables mediante lo que se denomina propagación de la evidencia.

El objetivo principal de esta tesis ha sido el estudio de los algoritmos existentes para tratar problemas del mundo de la clasificación supervisada, tratando de incorporar nuevas aproximaciones que puedan ser tenidas en consideración a la hora de abordar un nuevo caso de estudio.

Se han realizado aportaciones metodológicas en las siguientes vertientes:

1. *Redes Bayesianas en clasificación supervisada.* A pesar de que las redes Bayesianas fueron tenidas en cuenta previamente como clasificadores, se presentan en esta tesis algunos trabajos sobre datos reales en los que se aprecia que son una alternativa válida para realizar clasificación supervisada.

2. *Algoritmos de vecindad mutua.* Se ha realizado un estudio de las diferentes alternativas que ofrecen este tipo de algoritmos, y se ha presentado alguna modificación que puede hacer que su comportamiento mejore en algunos casos de estudio.

3. *Multiclasificadores.* En la actualidad se está investigando en la combinación de paradigmas clasificadores con el objetivo de aunar los aciertos y minimizar los errores de cada uno de ellos. Se presenta en esta tesis una nueva alternativa para la combinación de clasificadores basada en utilizar las redes Bayesianas para decidir la clase real a asignar a un nuevo caso, siendo las variables de entrada a esta red Bayesiana los resultados obtenidos por otros paradigmas clasificatorios.

Dichas aportaciones metodológicas se complementan con experimentaciones realizadas en casos reales del mundo médico y del mundo empresarial, así como en problemas estándares utilizados para realizar comparaciones con otras alternativas.

En los problemas estudiados, la metodología llevada a cabo ha sido eminentemente experimental. La elección de llevar a cabo un estudio experimental se ha debido a la dificultad derivada de acometer estudios de carácter analítico. Con objeto de propiciar la comparabilidad con respecto de métodos propuestos por otros autores, los métodos desarrollados se han aplicado a distintos ejemplos encontrados en la literatura.

1.2 Organización de la tesis

La tesis se divide en 7 partes bien diferenciadas:

- Parte I: INTRODUCCIÓN GENERAL
- Parte II: PARADIGMAS DE CLASIFICACIÓN SUPERVISADA
- Parte III: REDES BAYESIANAS
- Parte IV: ALGORITMOS DE CLASIFICACIÓN POR VECINDAD
- Parte V: COMBINACIÓN DE CLASIFICADORES
- Parte VI: RESULTADOS EXPERIMENTALES
- Parte VII: CONCLUSIONES Y LÍNEAS DE TRABAJO FUTURAS.

La Parte I, consta del Capítulo actual. La Parte II, a la que hemos denominado PARADIGMAS DE CLASIFICACIÓN SUPERVISADA, consta de un único Capítulo, el segundo, en el que se presentan los métodos más extendidos para abordar problemas enmarcados en el área de clasificación supervisada.

La Parte III se titula REDES BAYESIANAS, y consta de dos Capítulos. En el primero de ellos, el tercero, se tratan aspectos relacionados con la teoría de las redes Bayesianas. Se

introducen algunos conceptos de la teoría de grafos y la teoría de la probabilidad necesarios para los desarrollos posteriores, enunciándose el concepto de red Bayesiana. Se enumeran algunos de los métodos existentes para realizar la propagación de la evidencia y el aprendizaje estructural de las redes Bayesianas. El siguiente capítulo se centra en las aportaciones realizadas en la utilización de redes Bayesianas como paradigmas clasificadores, mostrándose los diversos modelos utilizados.

La Parte IV, titulada ALGORITMOS DE CLASIFICACIÓN POR VECINDAD, consta de dos Capítulos. En el Capítulo 5 se amplían las nociones que sobre este paradigma de clasificación se han esbozado en el Capítulo 2, presentando diversas alternativas que se pueden tener en cuenta a la hora de tratar un problema de clasificación mediante este tipo de aproximaciones. Se presentan también algunas alternativas especiales, en las que el objetivo no es obtener otro clasificador de la familia que se pueda comportar de forma más adecuada en determinados problemas, sino el reducir el conjunto de entrenamiento con el objeto de que la clasificación se realice de forma más rápida, sin perder, al menos en demasía, potencia clasificadora. En el Capítulo 6 se presentan las aportaciones que en esta familia de algoritmos se han realizado a lo largo de esta tesis. Estas consisten en tres nuevos algoritmos basados en el concepto de vecindad, y dos nuevos métodos de pesado del voto que aporta cada uno de los vecinos a la hora de decidir la clasificación a asignar a un caso dado.

En la Parte V, y bajo el título genérico de COMBINACIÓN DE CLASIFICADORES, se presentan en el primero de los dos Capítulos que la componen, el séptimo, diferentes técnicas que se utilizan en la literatura para la combinación de clasificadores con el objeto de crear un multiclasificador. En el Capítulo siguiente, el 8, se muestran las aportaciones realizadas en este campo, consistentes en un nuevo modelo de combinación, en el que, dadas las clasificaciones que a un caso a estudio le han sido asignadas por diferentes paradigmas, y dada una red Bayesiana inducida a partir de los resultados obtenidos en casos anteriores, se realiza la clasificación de un nuevo caso considerando las clasificaciones como evidencias y estableciendo a partir de ellas la clase que se asigna al caso com0o la más probable a posteriori dada la evidencia.

En la Parte VI se presentan, en tres Capítulos, los RESULTADOS EXPERIMENTALES logrados con las diferentes técnicas desarrolladas, para diferentes tipos de ficheros. Así, en el Capítulo 9, se presentan los resultados obtenidos en casos reales del mundo médico por las redes Bayesianas –consideradas como paradigma clasificatorio–, algoritmos basados en el concepto de vecindad y el nuevo multiclasificador aquí presentado. El Capítulo 10 presenta los resultados obtenidos por técnicas clásicas de combinación de clasificadores en un problema real enmarcado en el mundo de la empresa, mientras que en el Capítulo 11 se muestran los resultados obtenidos por diversos clasificadores en problemas estándares, bases de datos que se hallan a disposición de quien quiera utilizarlas para llevar a cabo comparaciones con trabajos anteriores realizados sobre las mismas.

La Parte VII, titulada CONCLUSIONES Y LÍNEAS DE TRABAJO FUTURAS, consta de un único Capítulo, el decimosegundo, en el cual establecemos las conclusiones, comentándose asimismo las vías abiertas o posibles trabajos a desarrollar en un futuro.

Parte II

PARADIGMAS DE CLASIFICACIÓN SUPERVISADA

Capítulo 2

Paradigmas de clasificación supervisada

2.1 Introducción

En los últimos años, las diferentes técnicas de reconocimiento de formas han adquirido una gran popularidad debido a su probada eficacia en la resolución de problemas de muy diversos ámbitos de la vida real. En Mitchell, (1997) se presenta una lista de las distintas áreas en las que se aplican con éxito este tipo de técnicas.

Desde el punto de vista de la clasificación, el objetivo principal de las técnicas de reconocimiento de formas, aplicadas a un problema general de clasificación, consiste en asignar a un objeto o fenómeno físico una de las diversas categorías o clases previamente especificadas (Sánchez, 1997).

Los problemas de clasificación supervisada serán abordados a lo largo de esta memoria mediante técnicas de aprendizaje automático. Suponemos que disponemos de una serie de variables predictoras $\{X_1, X_2, \ldots, X_n\}$, y una variable C representando a la clase real. Suponemos que, en una base de datos D, se hallan recogidos N casos del problema, en los cuales se conoce el valor de la clase, de la forma $D = \{(\mathbf{x_1}, \theta_1), (\mathbf{x_2}, \theta_2), \ldots, (\mathbf{x_N}, \theta_N)\}$, en un problema con M clases distintas, $\theta_i \in \{c_1, c_2, \ldots, c_M\} \forall i = 1, \ldots, N$. En la Tabla 2.1 se puede ver un ejemplo de una base de datos de este tipo, en el que los valores X_{i_j} indican que la variable X_i toma su j-ésimo valor.

En el campo del *reconocimiento de formas*, se suele hacer la distinción entre las aproximaciones paramétricas y las no paramétricas. En el primer caso, se asume un conocimiento a priori acerca de la forma funcional de las distribuciones de probabilidad de cada clase sobre el espacio de representación, las cuales vendrán determinadas por un conjunto finito de parámetros. Por lo tanto en este caso, las fronteras de decisión estarán definidas a partir de dichas distribuciones de clases.

Tabla 2.1: Forma general de las bases de datos.

N^{o} Caso	X_1	X_2	...	X_n	C
1	x_{1_2}	x_{2_1}	...	x_{n_2}	c_2
2	x_{1_1}	x_{2_3}	...	x_{n_1}	c_M
3	x_{1_3}	x_{2_1}	...	x_{n_3}	c_1
...	...	...	...	...	...
N	x_{1_1}	x_{2_2}	...	x_{n_2}	c_2

Por el contrario, la aproximación no paramétrica no supone ninguna forma de las distribuciones de probabilidad sobre el espacio de representación, de modo que el único conocimiento a priori será el correspondiente a la información inducida a partir de un conjunto de muestras controladas (conjunto de entrenamiento), de las cuales se conoce su clase verdadera. En consecuencia, en este caso las fronteras de decisión estarán determinadas por las muestras del conjunto de entrenamiento.

Dentro del denominado mundo del *Reconocimiento de Patrones* (*Pattern Recognition*) hay dos grandes grupos de familias que enfocan de manera distinta el problema de la clasificación.

Por un lado está la *Clasificación No Supervisada* (*Unsupervised Pattern Recognition*). También conocida como 'Clustering', enfoca la clasificación como el descubrimiento de las clases del problema. Los objetos únicamente vienen descritos por un vector de características, sin que sepamos a la clase a la que pertenece cada uno de ellos. Así, el objetivo de la 'Clasificación No Supervisada' será el descubrimiento de grupos de objetos que, afines en sus características, separen las diferentes clases del problema clasificatorio.

Por otro lado, la *Clasificación Supervisada* (*Supervised Pattern Recognition*) enfoca el problema clasificatorio de forma distinta. Parte de un conjunto de objetos descritos por un vector de características y la clase a la que pertenecen cada uno de ellos: a este conjunto de objetos del que conocemos la clase a la que pertenecen cada uno de ellos se le denomina 'Conjunto de Entrenamiento' o 'Conjunto de Aprendizaje'. Al venir recogidos en una base de datos, se suele utilizar también el término 'Base de Datos de Entrenamiento'. Así, basándose en este 'Conjunto de Entrenamiento', la Clasificación Supervisada construye un 'Modelo' o 'Regla General' que se utilizará para clasificar objetos nuevos de los cuáles no sepamos su clase. Para la creación de esta 'Regla General' la Clasificación Supervisada propone distintos paradigmas, los principales de los cuales serán explicados en puntos posteriores.

La Clasificación Supervisada ha sido utilizada en numerosos problemas de distinta índole tales como el diagnóstico de enfermedades, la concesión o rechazo de créditos en banca, predicción de

quiebra o bancarrota en empresas, reconocimiento de caracteres escritos a mano, detección de anormalidades en cromosomas, desambigüedades de calles mal escritas, etc.

Dentro del problema clasificatorio pueden producirse dos situaciones bien distintas, las cuáles denotan diferentes grados de complejidad del problema. Por un lado, puede producirse una situación en la que las clases que definen el problema sean *separables*. Esta situación viene dada, por ejemplo, cuando todos los objetos con las mismas características pertenecen a la misma clase. Cuando esto ocurre se dice que estamos ante un 'problema degenerado' o 'sin ruido'. Así, la Inteligencia Artificial en sus inicios únicamente era capaz de afrontar problemas en los que se daba esta condición. Por otro lado, con un grado mayor de complejidad aparecen problemas en los que las diferentes clases *no son separables*. Esta situación se produce cuando dos o más objetos con las mismas características pertenecen a diferentes clases. Esta es la situación habitual en problemas reales y se denominan problemas 'con ruido'. Así, para afrontar este último tipo de problemas tenemos diferentes Métodos Estadísticos y de Aprendizaje Automático que explicaremos posteriormente.

Otro concepto capital en el mundo de la clasificación es el de los diferentes *criterios para la evaluación de los clasificadores*. A la hora de evaluar y comparar la bondad de diferentes clasificadores se pueden adoptar distintos criterios:

- La *tasa de error* nos da una idea del porcentaje de objetos nuevos, de los cuáles no sabemos su clase, que no es capaz de clasificar bien el clasificador en cuestión. Este concepto será tratado posteriormente en un punto dedicado a la evaluación de los clasificadores.
- La *rapidez* con la que el clasificador construye el modelo o con la que clasifica objetos nuevos puede ser otro criterio a tener en cuenta.
- La *interpretabilidad del modelo* puede ser otra de las características fundamentales.
- La *simplicidad del modelo* obtenido también puede utilizarse como criterio de comparación entre clasificadores. Así, dentro de este concepto de la simplicidad de los modelos de clasificación tenemos dentro de la Estadística el 'Criterio de Occam' ('*Occam's razor*') y dentro del Aprendizaje Automático el criterio 'KISS' ('*Keep It as Simple as possible Stupid*'), los cuáles hacen referencia a la preferencia por el modelo más simple cuando estamos comparando distintos modelos con la misma *tasa de error*.

En el proceso de construcción de los modelos de clasificación, es necesario tener en cuenta la dirección que se sigue a la hora de construir el modelo o *dirección de la modelización*. Las diferentes direcciones que se pueden seguir en la construcción de modelos clasificatorios son las siguientes:

- La *modelización hacia adelante* (forward) comienza la construcción del modelo clasificatorio empezando desde el modelo más simple posible, y aumentando paso a paso la complejidad del modelo hasta el cumplimiento de algún criterio preestablecido.

- La *modelización hacia atrás* (backward) comienza la construcción del modelo clasificatorio empezando desde el modelo más complejo posible, y disminuyendo paso a paso la complejidad del modelo hasta el cumplimiento de algún criterio preestablecido.

- La *modelización paso a paso* (stepwise) comienza la construcción del modelo clasificatorio empezando desde el modelo más complejo o más simple posible, planteando en cada paso de la construcción del modelo tanto el aumento como la disminución de la complejidad del modelo.

- Por otro lado, la obtención del modelo clasificatorio puede enfocarse mediante una búsqueda en el espacio de modelos. El problema se plantea en los parámetros de una búsqueda en el espacio de posibles modelos. Esta cuestión será estudiada en la sección 2.5 de esta tesis.

2.2 Paradigmas estadísticos

El problema de la clasificación también ha sido largamente estudiado en el mundo de la Estadística. Los métodos estadísticos de clasificación tienen una fuerte base teórica y han servido para el desarrollo de nuevos métodos de clasificación que aprovechan en mayor profundidad las capacidades de cómputo y búsqueda de los ordenadores. A continuación presentaremos dos conocidos métodos estadísticos de clasificación: el análisis discriminante y la regresión logística.

2.2.1 Análisis discriminante

El análisis discriminante (Fisher, 1936) es uno de los métodos más sencillos de clasificación. Su nombre nos indica que una combinación lineal de la evidencia será utilizada para separar o discriminar entre las clases y para así predecir la clase de un nuevo caso. Para un problema con n variables –atributos o características–, esto supone geométricamente la creación de un hiperplano de dimensión $n-1$ para discriminar las diferentes clases del problema. Así, con tres características o atributos se utilizará un plano como separador y una recta sería suficiente en el caso de dos características. Este último caso lo podemos visualizar de una manera sencilla en el caso de la Figura 2.1, en el cual la recta r nos sirve para separar entre las clases c_1 y c_2 en un problema de dos características, X_1 y X_2.

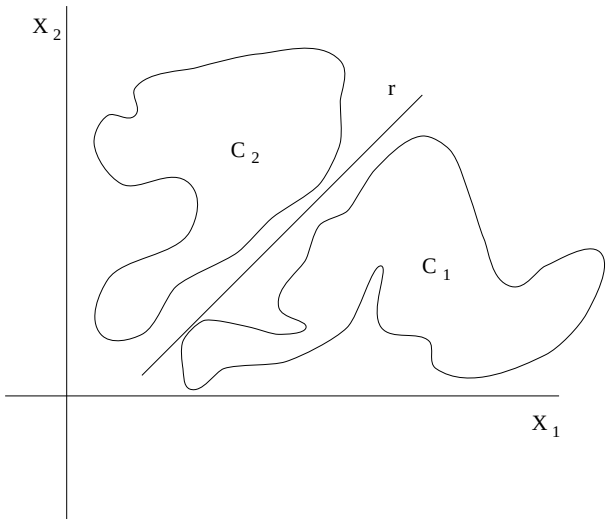


Figura 2.1: Separador lineal.

Es bien sabido que gran número de problemas reales no son separables por estructuras tan sencillas. Aún así, esto no quiere decir que la capacidad predictiva del análisis discriminante no sea competitiva, en según qué problemas, con otros clasificadores más evolucionados.

La forma general del análisis discriminante para un problema con n atributos $\{X_1, X_2, ..., X_n\}$ y dos clases a separar es la siguiente:

$$w_1 X_1 + w_2 X_2 + ... + w_n X_n - w_0$$

en la que las constantes w_i nos indican el peso discriminatorio de cada atributo. En un problema con más de dos clases debemos combinar los discriminadores que crearemos para cada par de clases o modificar el problema de tal forma que esté expuesto como una secuencia de decisiones dicotómicas: una clase versus todas las demás clases.

A la hora de clasificar un nuevo individuo, se calcula una puntuación o 'score' para cada clase, que servirá para asignar dicho individuo a una u otra clase. Podemos pensar en el análisis discriminante como una función que suma o resta puntuaciones o 'scores' en cada atributo o dimensión del problema, ponderando cada observación en el contexto de todas ellas y otorgándonos finalmente una puntuación o 'score' para cada clase. La clase seleccionada, C_i, será la que tenga la puntuación más alta. Se puede observar que todo lo que necesita el análisis discriminante es el cálculo de $n + 1$ valores. La forma en la que estos son calculados queda fuera del ámbito de este trabajo, pudiendo no obstante comentar que está íntimamente ligada con la asunción de una distribución normal multivariante de los datos para cada clase.

2.2.2 Regresión logística

La asunción de normalidad de los datos en la que se basa el análisis discriminante no es cierta en muchas situaciones. Así, aunque la variable edad pueda verse bajo una asunción de normalidad, una variable dicotómica, como el sexo, difícilmente puede ser vista bajo tal asunción. Aún y todo, es muy amplio el espectro de aplicaciones en los que, sin darse condiciones de normalidad por parte de los datos, asumir que las distribuciones son normales reporta buenos resultados.

Así, como un alternativa estadística que no estuviese basada en esta asunción de normalidad para separar las clases, surge la regresión logística (Hosmer y Lemesov, 1989). Esta se fundamenta en la siguiente fórmula para el caso de dos clases (M=2):

$$P(C = 1|X_1 = x_1, ..., X_n = x_n) = \frac{1}{1+e^{-b_0+\sum_{i=1}^{n} b_i x_i}}$$

que estima la probabilidad de que el caso $X_1 = x_1, ..., X_n = x_n$ pertenezca a la clase 1 en un problema con n atributos $\{X_1, X_2, ..., X_n\}$. Basándonos en esta fórmula, la regresión logística finalmente utiliza la siguiente expresión para obtener una puntuación o 'score' para cada clase, en clara similitud con el análisis discriminante:

$$-b_0 + b_1 X_1 + b_2 X_2 + ... + b_n X_n$$

clasificando en este caso cada individuo con la clase que ha obtenido una menor puntuación, ya que la clase asociada será la que obtenga mayor probabilidad de pertenencia. Aunque no entraremos a detallar la forma en la que se obtienen los pesos b_i, comentar que este cálculo está basado en la estimación máximo verosímil de los mismos.

2.3 Paradigmas de aprendizaje automático

2.3.1 Introducción

También conocidos bajo el término anglosajón de *machine learning*, los paradigmas del Aprendizaje Automático se han popularizado en su uso en la década de los 90. Inspirados en los clasificadores estadísticos ya expuestos, los clasificadores del aprendizaje automático realizan un proceso de creación de la regla de clasificación 'dejándose guiar por los datos', en el sentido de que ponen menos énfasis que los métodos clásicos ya expuestos en hipótesis a priori acerca de la distribución de los datos. En vez de testar hipótesis acerca de la distribución de los datos para crear el modelo, los métodos de aprendizaje automático realizan una búsqueda del conocimiento subyacente a los datos para crear la regla clasificatoria. Estos métodos han demostrado una mayor capacidad para trabajar en dominios con ruido, además de una mayor eficacia a la hora de tratar

con datos perdidos o 'missing'. Así, para realizar esta creación de la 'regla clasificatoria' han aparecido numerosos paradigmas dentro del Aprendizaje Automático, los más relevantes de los cuáles serán brevemente explicados en los siguientes apartados.

2.3.2 Ejemplo

Vamos a seguir el siguiente ejemplo para tratar de explicar con mayor claridad los paradigmas de aprendizaje automático. Se trata de un problema muy simplificado en el que se pretende descubrir cual es la lengua materna de una persona. Para ello utilizamos las siguientes variables predictoras:

1. *Faja*: Indica si la persona utiliza faja o no. Sus valores pueden ser *Si* o *No*, indicando si la utiliza o no respectivamente.

2. *SílabasApellido*: Indica el número de sílabas que tiene el apellido de la persona de la cual queremos averiguar la lengua materna. Sus valores pueden ser $1, 2, \ldots$

3. *Boina*: Indica si la persona utiliza boina o no. Sus valores pueden ser *Si* o *No*, indicando si la utiliza o no respectivamente.

4. *Rh*: Indica el Rh de la sangre de la persona en cuestión. Sus dos posibles valores son *Positivo* o *Negativo*.

Las clases que se tienen en consideración son las siguientes: *Andaluza*, *Catalana*, *Gallega* y *Euskera*.

Se dispone de la base de datos de entrenamiento que se muestra en la Tabla 2.2.

Se pretende obtener el modelo clasificatorio para este ejemplo de juguete mediante la aplicación de los paradigmas de clasificación supervisada que se presentan. De este modo, si luego se presenta un nuevo caso, como por ejemplo el mostrado en la Tabla 2.3, –donde el hecho de que el campo clase tenga asociado como valor '?' indica que se desconoce el valor de la clase para esta persona–, se aplicará el modelo clasificatorio que se ha obtenido a partir de la base de datos de entrenamiento para asignarle su clasificación. No podemos obviar el hecho de que la clasificación que se asigna a un caso nuevo no tiene porque ser siempre la correcta, ya que estamos en un mundo sujeto a errores. Es por ello que no se puede tratar un nuevo caso clasificado como nuevo miembro de la base de datos de entrenamiento, utilizando el conocimiento que éste añade, hasta que no se le haya asignado su clase verdadera, esto es, en el ejemplo que nos ocupa, hasta que no sepamos realmente cual es la lengua materna de la persona en concreto.

Tabla 2.2: Base de datos de entrenamiento para el ejemplo de la lengua materna.

Nº Caso	Faja	SílabasApellido	Boina	Rh	Clase
1	No	3	No	Positivo	Andaluza
2	No	4	No	Negativo	Andaluza
3	No	3	Si	Positivo	Andaluza
4	Si	3	No	Negativo	Catalana
5	No	2	Si	Positivo	Catalana
6	Si	3	No	Negativo	Catalana
7	Si	4	No	Positivo	Catalana
8	Si	3	No	Negativo	Catalana
9	No	2	Si	Positivo	Gallega
10	No	3	No	Negativo	Gallega
11	No	4	Si	Positivo	Gallega
12	No	3	Si	Negativo	Euskera
13	No	6	Si	Positivo	Euskera
14	No	5	No	Negativo	Euskera
15	No	7	Si	Positivo	Euskera
16	No	10	Si	Negativo	Euskera

2.3.3 Inducción de reglas

Uno de los paradigmas más fácilmente interpretables del Aprendizaje Automático son los conjuntos de reglas si-entonces (IF-THEN). El objetivo de este paradigma es la inducción de un conjunto de reglas cortas, simples y comprensibles en dominios con ruido, que nos sirvan para discriminar entre las categorías del problema.

Dentro de las diferentes aproximaciones de este paradigma tenemos algoritmos como el *CN2* (Clark y Nibblet, 1989), RIPPER (Cohen, 1995) y OneR (Holte, 1994). Ya que guardan numerosas similitudes entre sí, basándonos en el algoritmo *CN2*, explicaremos a continuación algunas de las características básicas de este paradigma. Véase el libro de Hand (1997) para una más amplia introducción al tema.

Las reglas inducidas están formadas por los siguientes componentes:

Tabla 2.3: Ejemplo de un caso a clasificar.

Faja	SílabasApellido	Boina	Rh	Clase
Si	5	No	Negativo	?

- El *Selector* relaciona un atributo a un valor de atributo o conjunto de valores disjuntos.
- El *Complex*, formado por uno o varios selectores, define la parte de la condición de la regla.
- El *Rule set* es la lista ordenada de reglas para identificar clases, prediciendo cada regla una única clase.

A la hora de clasificar un nuevo caso, *CN2* realiza un emparejamiento estricto. Esto es, cada regla del *Rule set* ordenado se intenta emparejar con el nuevo caso hasta que se encuentra una que satisface todos los atributos del caso. La predicción de esta regla será con la que se clasificará nuestro ejemplo. Si ninguna regla es satisfecha por el nuevo ejemplo, se tiene una regla por defecto mediante la que se asigna la clase más frecuente del conjunto de entrenamiento al nuevo caso.

A continuación enumeraremos algunas de las cuestiones básicas sobre la construcción del conjunto de reglas o *Rule set*:

- El conjunto de reglas se construye gradualmente generando las reglas una a una. Inicialmente el conjunto de reglas esta vacío.
- El conjunto de entrenamiento utilizado para generar cada regla está formado por aquellos ejemplos que no satisfacen las condiciones de ninguna regla del conjunto de reglas existentes hasta dicho momento.
- La búsqueda de un *complex* se realiza prefiriendo los que son satisfechos por un alto número de ejemplos de una única clase, y unos pocos o ninguno de las demás clases. El *complex* seleccionado deberá pasar algún test estadístico para demostrar su significatividad.
- El mejor *complex* encontrado se convierte en la parte de la condición de la regla, prediciendo dicha regla la clase mayoritaria de los ejemplos del conjunto de entrenamiento que abarca. En cada paso de la búsqueda, se mantiene el conjunto de los mejores *complex* encontrados hasta el momento.

A la hora de testar la significatividad de un *complex*, se debe comparar la distribución observada entre las clases de ejemplos satisfaciendo el *complex*, con la distribución esperada bajo la hipótesis nula de que el *complex* selecciona ejemplos al azar. Para testearlo *CN2* utiliza el estadístico basado en el test de la razón de verosimilitud:

$$2 \cdot \sum_{i=1}^{M} f_i \cdot \log(f_i/e_i)$$

donde
$F = (f_1, ..., f_M)$ distribución de frecuencias observadas en el conjunto de ejemplos entre las clases satisfaciendo el *complex*. $f_i = N \cdot p_i$.
$E = (e_1, ..., e_M)$ distribución de frecuencias esperadas bajo la hipótesis nula (selección de ejemplos al azar), es decir $e_i = N \cdot q_i$.
$Q = (q_1, .., q_M)$ probabilidad de las clases en el conjunto de entrenamiento global.

$$2 \cdot \sum_{i=1}^{M} f_i \cdot \log(f_i/e_i) = 2 \cdot N \sum_{i=1}^{M} p_i \cdot \log(p_i/q_i).$$

Veamos como se comporta el paradigma de inducción de reglas para el ejemplo planteado.

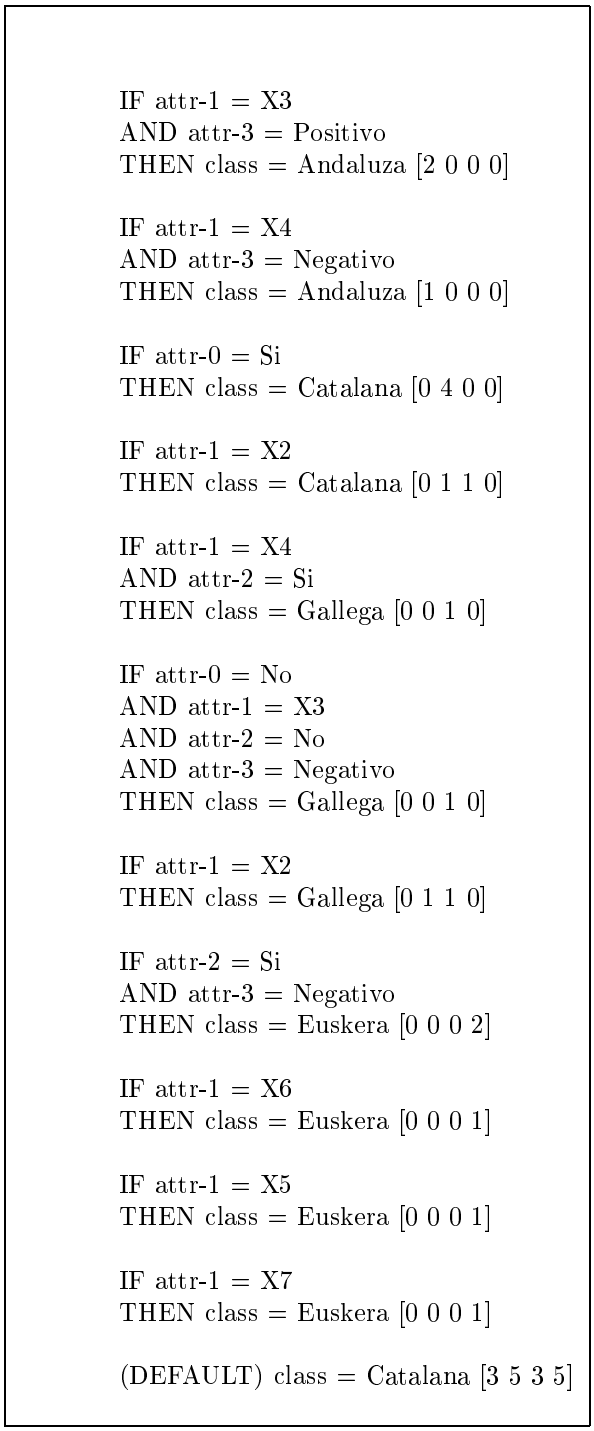
En este apartado se ha utilizado el algoritmo de inducción de reglas *CN2*. Las ejecuciones se han realizado utilizando la librería de funciones MLC++ (Kohavi y col., 1997). Mediante este software se obtienen automáticamente una serie de reglas que son fácilmente aplicables para realizar la asignación a los casos a estudio. En la Figura 2.2 se puede ver un subconjunto de las reglas inducidas para el ejemplo que nos ocupa.

La interpretación de las reglas es bastante sencilla. Por ejemplo, la primera de ellas indica lo siguiente:

```
IF    attr_1 = X3
  AND attr_3 = Positivo
THEN  class = Andaluza  [2 0 0 0]
```

Si el atributo 1 (*SílabasApellido*) vale su tercer valor (3 en este caso), y el atributo 3 (*Rh*) es igual a *Positivo*, entonces clasifica el nuevo caso como de lengua *Andaluza*, ya que entre los casos que se han tratado aparece la distribución [2 0 0 0], esto es, con estas características, en la base de datos se han encontrado dos personas de lengua *Andaluza*, cero de lengua *Catalana*, cero de lengua *Gallega* y cero de lengua *Euskera*.

La última regla,



IF attr-1 = X3
AND attr-3 = Positivo
THEN class = Andaluza [2 0 0 0]

IF attr-1 = X4
AND attr-3 = Negativo
THEN class = Andaluza [1 0 0 0]

IF attr-0 = Si
THEN class = Catalana [0 4 0 0]

IF attr-1 = X2
THEN class = Catalana [0 1 1 0]

IF attr-1 = X4
AND attr-2 = Si
THEN class = Gallega [0 0 1 0]

IF attr-0 = No
AND attr-1 = X3
AND attr-2 = No
AND attr-3 = Negativo
THEN class = Gallega [0 0 1 0]

IF attr-1 = X2
THEN class = Gallega [0 1 1 0]

IF attr-2 = Si
AND attr-3 = Negativo
THEN class = Euskera [0 0 0 2]

IF attr-1 = X6
THEN class = Euskera [0 0 0 1]

IF attr-1 = X5
THEN class = Euskera [0 0 0 1]

IF attr-1 = X7
THEN class = Euskera [0 0 0 1]

(DEFAULT) class = Catalana [3 5 3 5]

Figura 2.2: Subconjunto de las reglas obtenidas por el algoritmo *CN2*.

```
(DEFAULT) class = Catalana  [3 5 3 5]
```

nos indica que si no se cumple ninguna de las reglas anteriores, se debe clasificar el caso como de lengua *Catalana*.

De este conjunto de reglas, y aplicándolas en el orden en que se encuentran, podemos extraer el conocimiento suficiente para clasificar un nuevo caso que se nos presente. En el ejemplo a clasificar mostrado en la Tabla 2.3, la clasificación se realizaría de la siguiente manera: se mira, en orden, si alguna de los valores de las variables predictoras de este nuevo caso cumplen la condición de alguna de las reglas. En caso de que esto ocurra, se clasifica el caso con la clase que aparece en la sentencia asociada a la regla y se detiene el proceso de clasificación. Si los valores de las variables no coinciden con la condición de ninguna regla, se asociará al nuevo caso la clase que indica la regla por defecto, en este caso, *Catalana*.

En el ejemplo, la regla que se aplica es la tercera, clasificándose en consecuencia el caso como de lengua materna *Catalana*.

2.3.4 Árboles de clasificación

Los árboles de clasificación son uno de los paradigmas más clásicos y ampliamente usados del mundo del Aprendizaje Automático (Hunt y Hovland, 1962, Morgan y Sonquist, 1963). En la literatura acerca de este paradigma nos encontramos numerosas implementaciones como CART (Breiman y col., 1984), ID3 (Quinlan, 1986), ASSISTANT (Kononenko y col., 1984), OC1 (Murthy y Salzberg, 1994) o C4.5 (Quinlan, 1993).

Un árbol de clasificación está formado por nodos y ramas. Cada nodo representa un test univariado o decisión sobre los valores de un atributo concreto. El primer nodo del árbol es conocido como el nodo raíz. Finalmente tenemos los nodos terminales u hojas en los que se toma una decisión acerca de la clase a asignar. Así, a la hora de clasificar un nuevo caso, tendrán que compararse los valores de sus atributos con las decisiones o tests que se toman en los nodos, siguiendo la rama que coincida con dichos valores en cada test, para finalmente llegar a un nodo terminal u hoja que prediga una clase para el caso tratado. Un árbol de decisión también puede verse como un conjunto de reglas si-entonces, si bien la diferencia más obvia entre los dos paradigmas es que las reglas de decisión son independientes entre sí, mientras que las reglas extraídas del árbol de decisión no lo son.

Figura 2.3: Árbol de clasificación obtenido para el ejemplo de la lengua materna.

En la Figura 2.3 tenemos un ejemplo de árbol de clasificación. Este árbol es el que se obtiene para el ejemplo de clasificación de la lengua materna de una persona dados los valores de las variables. Se ha utilizado el software MineSet (Kohavi y col., 1997).

En la Figura 2.4 se observa un detalle final del árbol de clasificación obtenido. Se aprecia de manera más clara la parte profunda del mismo, y se ve que los nodos hoja no están constituidos en ella por elementos de una sola clase. Esto se debe a dos razones fundamentales: en primer lugar, si el fichero de casos de entrenamiento tiene ruido, esto es, dos casos con los mismos valores para todas las variables predictoras pertenecen a distintas clases, no es posible separar estos casos en el recorrido del árbol, y en segundo lugar, el algoritmo utilizado en este ejemplo es el ID3 (Quinlan, 1986), y en él se realiza un proceso de poda para evitar sobreajustarse al fichero de entrenamiento.

Existen varios métodos para hallar el modelo asociado en forma de árbol de clasificación. Los que aquí se presentan se basan en el algoritmo general de inducción de árboles de clasificación en

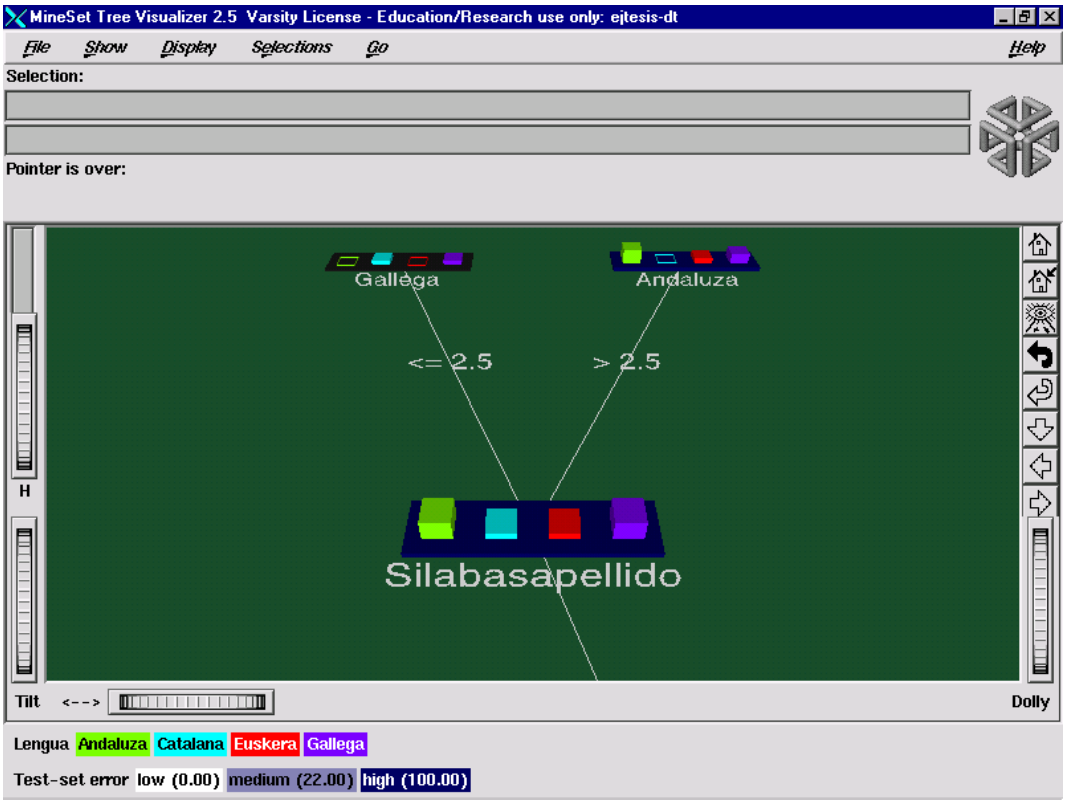


Figura 2.4: Detalle del árbol de clasificación.

modo descendente, conocido como algoritmo T.D.I.D.T. (*Top Down Induction of Decision Trees*). El algoritmo es presentado en la Figura 2.5. La forma de seleccionar el atributo más significativo es la que diferencia los diferentes algoritmos.

La construcción del árbol está basada en un método de particionamiento recursivo del Conjunto de Entrenamiento. Para realizar este particionamiento se han propuesto diferentes medidas para seleccionar el atributo que mejor discrimine las clases en cada nodo del árbol y así ir particionando el Conjunto de Entrenamiento.

Teniendo en cuenta la siguiente notación, correspondiente a la Tabla 2.4, donde,
M número de clases distintas que hay en el conjunto de ejemplos del nodo a particionar
V número de subconjuntos en el nodo a particionar
m_v número de ejemplos en el subconjunto v
f_{cv} número de ejemplos en el subconjunto v de clase c

Algoritmo Top Down Induction of Decission Trees

T. D. I. D. T.

INPUT: Conjunto de ejemplos, S, caracterizados por atributos,
y una clase de pertenencia
OUTPUT: Arbol de clasificación
begin *T. D. I. D. T.*
if todos los ejemplos en S son de la misma clase c
then
resultado nodo simple etiquetado como c
else
begin
Seleccionar el atributo más informativo X_i con valores $x_{i1}, ..., x_{il}$;
Particionar S en $S_1, ..., S_l$ de acorde a los valores de X_i;
Construir subárboles $T_1, ..., T_l$ para $S_1, ..., S_l$;
El resultado final es un árbol con raíz X_i y subárboles $T_1, ..., T_l$.
Las uniones entre X_1 y los subárboles están etiquetadas mediante $x_{i1}, ..., x_{il}$;
end
end

Figura 2.5: Algoritmo general de inducción de árboles de clasificación.

N número de ejemplos en el nodo a particionar
n_c número de ejemplos de la clase c en el nodo a particionar.

Se presentan algunas de las medidas más relevantes para localizar el *atributo más informativo*:

- *Ganancia en información. (ID3, Quinlan, 1986)*
  gain= $(\sum_{i=1}^{M}[-(\frac{n_c}{N})\log_2(\frac{n_c}{N})]) - (\sum_{v=1}^{V}(\frac{m_v}{N})\sum_{c=1}^{C}[-(\frac{f_{cv}}{m_v})\log_2(\frac{f_{cv}}{m_v})])$
  Esta medida está basada en la Teoría de la Información y mide la diferencia entre la entropía en el nodo padre y la entropía media de sus ramas.

- *Gain ratio. C4.5., Quinlam, 1993*
  gain ratio= gain / $\sum_{v=1}^{V}[-(\frac{m_v}{N})\log_2(\frac{m_v}{N})]$
  Compensa el sesgo derivado de las particiones con muchas ramas que aparece en la Ganancia en información.

- $1-d$. *(López de Mántaras, 1991)*
  $1-d$ = gain / $\sum_{v=1}^{V}\sum_{c=1}^{M}[-(\frac{f_{cv}}{N})\log_2(\frac{f_{cv}}{N})]$

Tabla 2.4: Ejemplo de un caso a clasificar.

	sub1	...	subV	Total
clase1	f_{11}	...	f_{1V}	c_1
...	...	...	...	...
...	...	...	...	...
claseM	f_{C1}	...	f_{CV}	c_M
Total	m_1	.....	m_V	N

Compensa el sesgo derivado de las particiones con muchas ramas que aparece en la Ganancia en información.

- *Chi- squared. CHAID. (Kass y col., 1988)*
  $\chi^2 = \sum_{c=1}^{M} \sum_{v=1}^{V} \frac{(f_{cv}-e_{cv})^2}{e_{cv}}$ con $e_{cv} = (n_c m_v/N)$; χ^2 sigue aproximadamente una distribución χ^2 con $(M-1)\times(V-1)$ grados de libertad.
  Basado en el test de la χ^2.

En la mayoría de los algoritmos, la construcción del árbol se efectúa en dos fases. En una primera de *Desarrollo* o 'Growing' se realiza la expansión del árbol hasta que la medida de partición que utilicemos no supere un umbral. Partiendo del árbol que tenemos en este momento se emprende una segunda fase de *Poda* o 'Pruning', que eliminará o podará algunas de las ramas formadas en el árbol, comenzando dicha poda desde el nivel más profundo del árbol, hasta que un test estadístico (específico al programa) nos detenga en esta labor de reducción de la complejidad del árbol.

2.3.5 K-NN

La *Regla de los K vecinos más próximos* o 'K-Nearest Neighbour' (Dasarathy, 1991) es uno de los paradigmas de más sencilla comprensión del Aprendizaje Automático. Para clasificar un nuevo individuo, previamente se guardan en memoria todas las instancias del Conjunto de Entrenamiento; se calculan (mediante el uso de una distancia específica, típicamente la distancia euclídea) las distancias entre la nueva instancia y todas las instancias del Conjunto de Entrenamiento; se tienen en cuenta las K instancias del Conjunto de Entrenamiento (de las cuáles sabemos la clase a la que pertenecen) más cercanas a la recién llegada para predecir la clase de ésta, ponderando el peso

que tiene cada una de las K instancias para tomar esta decisión por medio de la distancia a la que se encuentran de ésta.

Así expuesto, el algoritmo admite varias variantes, sea en el tipo de distancia que se utilice (euclídea, de Mahalanobis, ...), sea en la ponderación de la contribución de cada atributo en el cálculo de las distancias, sea la forma en la que las K instancias más cercanas asignan la clase a la nueva instancia (ponderando cada instancia mediante su distancia, voto por la mayoría,...).

Aunque de sencillo aprendizaje, este algoritmo puede ser inabordable para bases de datos grandes, dada la cantidad de memoria necesaria para almacenar las instancias del conjunto de entrenamiento y la cantidad de cálculos en el cómputo de las distancias.

Para poder aplicar al ejemplo utilizado en esta introducción el paradigma K-NN, en su versión más básica, necesitamos recodificar los valores de las variables para que puedan ser tratados como números en el cálculo de las distancias. Existen versiones del algoritmo que permiten tratar datos cualitativos, pero en esta sección no entramos a valorarlos.

La transformación que se realiza es la siguiente:

1. Variable *Faja*:
   - Valor *No* se recodifica como el valor numérico 1
   - Valor *Si* se recodifica como el valor numérico 2
2. Variable *SílabasApellido*:
   - Se mantienen los valores numéricos tal cual están
3. Variable *Boina*:
   - Valor *No* se recodifica como el valor numérico 1
   - Valor *Si* se recodifica como el valor numérico 2
4. Variable *Rh*:
   - Valor *Negativo* se recodifica como el valor numérico 1
   - Valor *Positivo* se recodifica como el valor numérico 2
5. Variable *Clase*:
   - Valor *Andaluza* se recodifica como el valor numérico 1
   - Valor *Catalana* se recodifica como el valor numérico 2
   - Valor *Gallega* se recodifica como el valor numérico 3

Tabla 2.5: Base de datos de entrenamiento recodificada.

Nº Caso	Faja'	SílabasApellido'	Boina'	Rh'	Lengua'
1	1	3	1	2	1
2	1	4	1	1	1
3	1	3	2	2	1
4	2	3	1	1	2
5	1	2	2	2	2
6	2	3	1	1	2
7	2	4	1	2	2
8	2	3	1	1	2
9	1	2	2	2	3
10	1	3	1	1	3
11	1	4	2	2	3
12	1	3	2	1	4
13	1	6	2	2	4
14	1	5	1	1	4
15	1	7	2	2	4
16	1	10	2	1	4

- Valor *Euskera* se recodifica como el valor numérico 4

Con lo que se obtiene la base de datos recodificada mostrada en la Tabla 2.5.

Obviamente, también es necesario recodificar los nuevos casos a clasificar. El caso que se viene utilizando, una vez recodificado, se muestra en la Tabla 2.6.

A la hora de asignar una clasificación a este nuevo caso, se debe tener en consideración el valor del parámetro K, esto es, el número de vecinos que se van a tener en cuenta para asignar un valor a la clase de este nuevo caso. En la Figura 2.6, se aprecia el resultado de la ejecución[1] del algoritmo K-NN con K=3. La resolución de los empates se realiza por distancia mínima. En este caso, el algoritmo clasifica el nuevo caso como de lengua *Euskera*, ya que los tres vecinos más próximos, que corresponden a los casos 13, 1 y 6 de la base de datos de entrenamiento, son de distintas clases (4=*Euskera*, 1=*Andaluza* y 2=*Catalana*, respectivamente), y de ellos el más cercano es el caso 13, a una distancia de 1. Obviamente, este será también el resultado del algoritmo NN.

[1] El software utilizado es de elaboración propia.

Tabla 2.6: Ejemplo recodificado.

Faja'	SílabasApellido'	Boina'	Rh'	Clase'
2	5	1	1	?

```
   K auzokide hurbilenak: (K = 3)
      hurbilenak[0]=13 distantzia=1.000000 klasea=4.000000
      hurbilenak[1]=1 distantzia=1.414214 klasea=1.000000
      hurbilenak[2]=6 distantzia=1.414214 klasea=2.000000
           klasea=4.000000 kont=0.500000
           klasea=1.000000 kont=0.414214
           klasea=2.000000 kont=1.080880
      Klase maioritarioa=4.000000
Amaiera 3-NN
```

Figura 2.6: Ejecución del algoritmo K-NN para K=3.

En la Figura 2.7 se observa el resultado de la ejecución del mismo algoritmo con K=5. En este caso, la clasificación que se obtiene para el nuevo caso es la clase 2, esto es, lo clasifica como de lengua *Catalana*. Con el objeto de ver la importancia del valor del parámetro K, se presentan en la Tabla 2.6 los resultados que se obtienen para diferentes valores del mismo. Se observa que la clasificación que asigna el algoritmo al nuevo caso varía según el valor del parámetro K. Esto se manifiesta en mayor medida en problemas con mayor complejidad que el ejemplo presentado, aumentando de este modo la importancia de elegir un valor adecuado del número de vecinos a considerar.

2.3.6 Naive-Bayes

En los últimos años ha habido un interés creciente en la utilización de métodos probabilísticos para clasificación. Estos han demostrado acomodarse a la naturaleza flexible de numerosos conceptos y además gozan de una sólida base en la teoría de la probabilidad. El método probabilístico para clasificación más ampliamente utilizado es conocido como el método del 'Simple-Bayes' o *Naive-Bayes* (Cestnik, 1990), el cual se basa en una aplicación del teorema de Bayes.

```
K auzokide hurbilenak: (K = 5)
   hurbilenak[0]=13 distantzia=1.000000 klasea=4.000000
   hurbilenak[1]=1 distantzia=1.414214 klasea=1.000000
   hurbilenak[2]=6 distantzia=1.414214 klasea=2.000000
   hurbilenak[3]=3 distantzia=2.000000 klasea=2.000000
   hurbilenak[4]=5 distantzia=2.000000 klasea=2.000000
       klasea=4.000000 kont=0.500000
       klasea=1.000000 kont=0.414214
       klasea=2.000000 kont=1.080880
       klasea=2.000000 kont=1.080880
       klasea=2.000000 kont=1.080880
  Klase maioritarioa=2.000000
Amaiera 5-NN
```

Figura 2.7: Ejecución del algoritmo K-NN para K=5.

Para clasificar una nueva instancia I, podemos primeramente basarnos en el teorema de Bayes para calcular la probabilidad a posteriori con la que la instancia puede pertenecer a cada una de las clases del problema,

$$p(c_i|I) = \frac{p(c_i)p(I|c_i)}{p(I)}$$

donde $p(c_i)$ es la probabilidad a priori de la clase c_i en las instancias del conjunto de entrenamiento. De todas formas, sabiendo que la instancia I es una conjunción de los n valores de sus atributos descriptivos $X_1 = x_1, X_2 = x_2, \ldots, X_n = x_n$, podemos reescribir la regla expuesta de la siguiente forma:

$$p(C = c_i \mid X_1 = x_1, \ldots, X_n = x_n) \propto p(C = c_i)p(X_1 = x_1, \ldots, X_n = x_n \mid C = c_i),$$

donde $p(X_1 = x_1, X_2 = x_2, \ldots, X_n = x_n \mid C = c_i)$ supone la probabilidad de que ocurra la instancia I cuando la clase es c_i. Estas probabilidades pueden ser estimadas mediante las frecuencias del Conjunto de Entrenamiento. Aún así esta expresión no es operacional, ya que el término $p(X_1 = x_1, X_2 = x_2, \ldots, X_n = x_n \mid C_i)$ será habitualmente cero. Para hacer operativo el teorema de Bayes, el clasificador Naive-Bayes asume que los atributos que definen cada instancia son independientes entre sí dada la clase del problema, lo que nos permite utilizar la siguiente igualdad:

$$p(X_1 = x_1, X_2 = x_2, \ldots, X_n = x_n|C = c_i) = \prod_{l=1}^{n} p(X_l = x_l|C = c_i)$$

Tabla 2.7: Resultados obtenidos con el paradigma K-NN utilizando diferentes valores de K.

K	Casos vecinos	Distancias	Clases vecinos	Clase
1	C_{13}	1	E	Euskera
2	C_{13}, C_1	1, 1.41	E, A	Euskera
3	C_{13}, C_1, C_6	1, 1.41, 1.41	E, A, C	Euskera
4	C_{13}, C_1, C_6, C_3	1, 1.41, 1.41, 2	E, A, C, C	Catalana
5	$C_{13}, C_1, C_6, C_3, C_5$	1, 1.41, 1.41, 2, 2	E, A, C, C, C	Catalana
6	$C_{13}, C_1, C_6, C_3, C_5, C_{10}$	1, 1.41, 1.41, 2, 2, 2	E, A, C, C, C, G	Catalana
7	$C_{13}, C_1, C_6, C_3, C_5, C_{10}, C_{12}$	1, 1.41, 1.41, 2, 2, 2, 2	E, A, C, C, C, G, E	Catalana

donde las probabilidades condicionales del valor de cada atributo X_l dada la clase c_i vienen estimadas de las frecuencias del Conjunto de Entrenamiento. Utilizando la última igualdad, se calculará la probabilidad a posteriori de cada clase dada la instancia, clasificándose la instancia con la clase que computa la mayor de estas probabilidades.

A pesar de su simplicidad y la asunción de independencia entre las variables en la que se basa (la cuál no es cierta para la mayoría de los problemas reales), ha demostrado su eficacia en numerosos problemas de muy distinta naturaleza. La citada aproximación del naive-Bayes será utilizada en esta tesis así como otra aproximación, llamada 'Naive-Bayes-Tree' o 'NBTree' (Kohavi, 1996), en la que, tras desarrollar un árbol de clasificación, se aplica la regla correspondiente al clasificador Naive-Bayes construido sobre las hojas del árbol.

Ejemplo

En la Figura 2.8 se puede apreciar el modelo *naive Bayes* para el problema planteado. En él, se asume que la variable correspondiente a la clase es madre de todas las variables predictoras, siendo estas, a su vez, condicionalmente independientes[2] dada la clase. Se presenta en forma de red Bayesiana (Friedman y col., 1997), que será explicada en la siguiente sección. Para realizar las ejecuciones se utiliza el software HUGIN (Andersen y col. 1989). Los valores que se muestran son los estimadores de las probabilidades a priori de los valores de cada una de las variables.

[2] El concepto de independencia condicional será presentado en el capítulo dedicado a las redes Bayesianas. Por el momento consideraremos que una variable o un conjunto de ellas $\mathbf{x}$ es independiente de otro conjunto de variables disjunto $\mathbf{y}$ dado un tercer conjunto de variables $\mathbf{z}$, disjunto de los dos anteriores, si se cumple que, dados los valores de las variables de $\mathbf{z}$, los valores de las variables de $\mathbf{y}$ no afectan en el cálculo de las probabilidades condicionadas de los valores de las variables de $\mathbf{x}$.

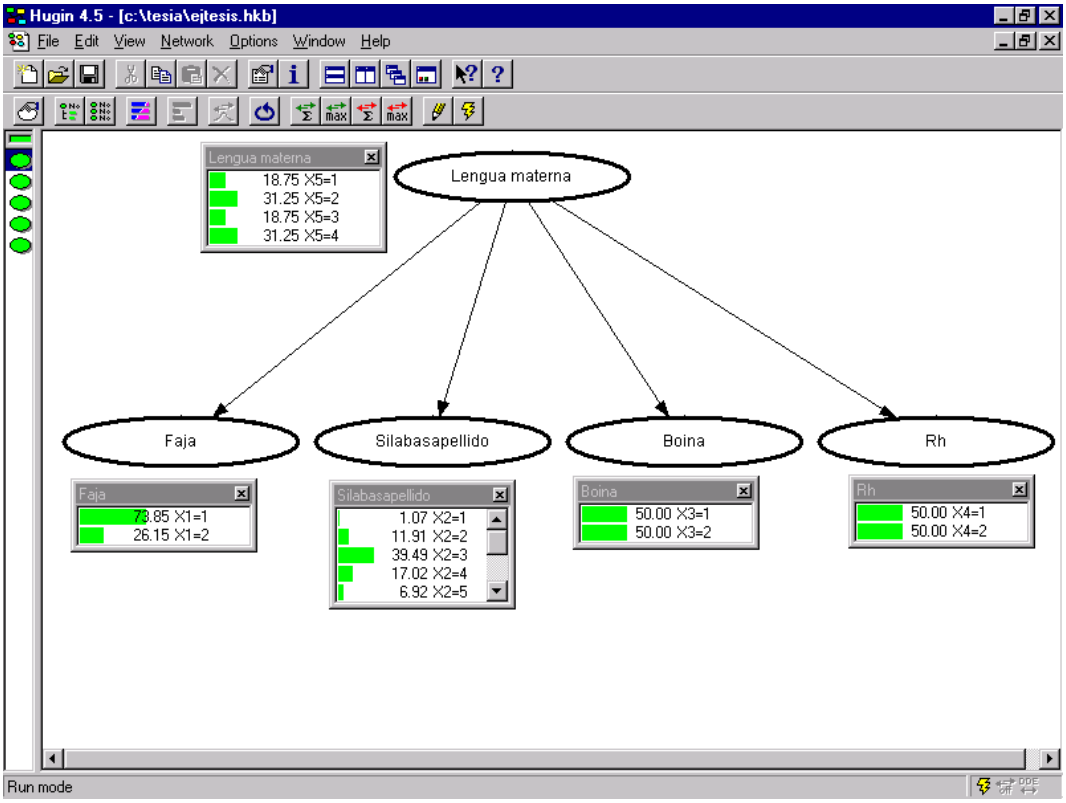


Figura 2.8: Modelo *naive Bayes* para el ejemplo planteado.

En la Figura 2.9 se observa el resultado de la clasificación para el ejemplo que se viene manejando a lo largo de esta presentación de los modelos de aprendizaje automático. Se observa que las variables predictoras han sido instanciadas al valor correspondiente en el ejemplo a clasificar, y como resultado, se obtiene la probabilidad a posteriori de cada una de las clases. En este caso, se le asigna al ejemplo la clase 2 (*Catalana*), ya que resulta ser la clase con probabilidad a posteriori más elevada (0.9259).

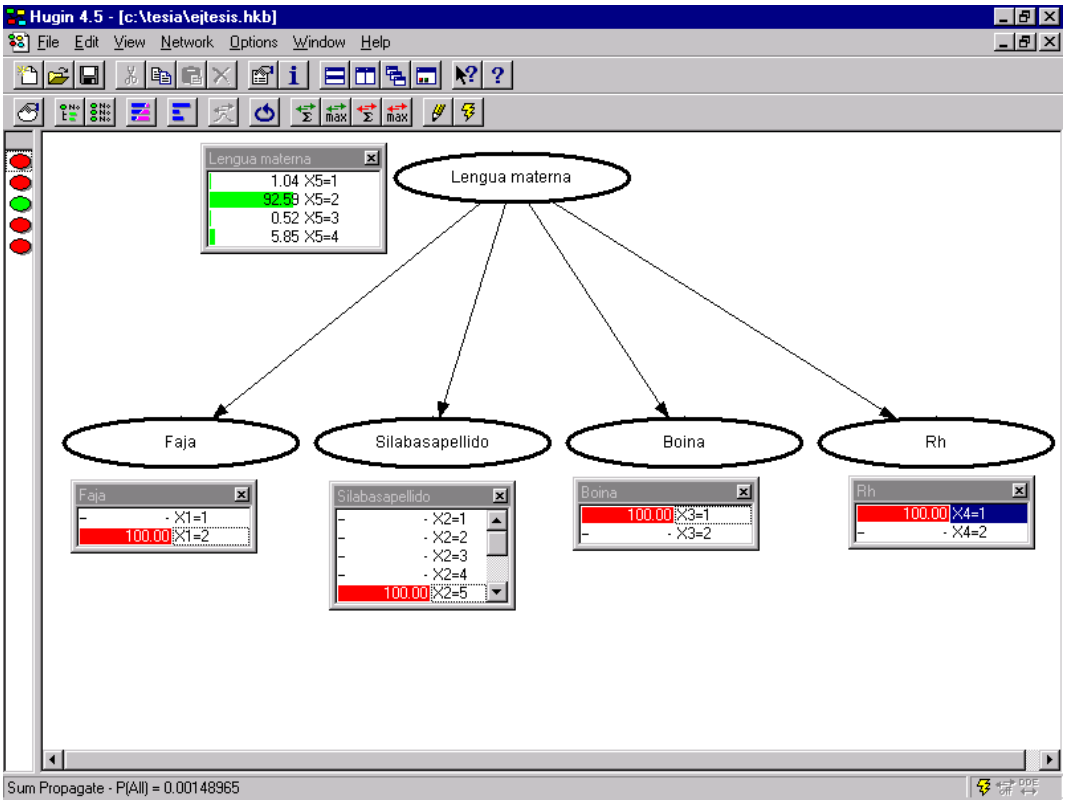


Figura 2.9: Clasificación del nuevo caso.

2.3.7 Redes Bayesianas

Se presentan en esta sección las redes Bayesianas (RRBB) (Pearl, 1988, Jensen, 1996). Su importancia en el desarrollo de esta tesis les hace merecer un capítulo aparte, pero esta sucinta introducción se considera necesaria para poder apreciar su comportamiento cuando son vistas como modelo de clasificación supervisada.

Las redes Bayesianas representan un método de razonamiento basado en la teoría de la probabilidad y están siendo motivo de estudio e investigación –fundamentalmente en la última década–, entre los miembros de la comunidad de Incertidumbre en Inteligencia Artificial. En cierto modo constituyen una introducción de las metodologías probabilística y estadística en la rama de la Inteligencia Artificial conocida bajo el nombre de sistemas expertos. Si bien la probabilidad se consideraba inadecuada para tratar con la incertidumbre, debido fundamentalmente a la necesidad de especificar un gran número de parámetros en las distribuciones de probabilidad conjuntas, es a partir de la explotación del concepto de independencia condicional, cuando el problema representacional se convierte en abordable.

La definición de independencia condicional sirve para introducir el concepto de red Bayesiana. Desde una perspectiva informal las redes Bayesianas son grafosaAcíclicos dirigidos (DAG-s), donde los vértices son variables aleatorias, y los arcos especifican las relaciones de independencia condicional entre las variables.

Para especificar la distribución de probabilidad de una red Bayesiana, se debe proporcionar la distribución de probabilidad a priori de todos los vértices raíz (vértices sin predecesores), así como las probabilidades condicionadas de todos los vértices no raíz, para cada posible combinación de sus padres o predecesores directos. Estos números, en conjunción con el DAG especifican totalmente la red Bayesiana. La probabilidad conjunta de cualquier punto n dimensional $(x_1, ..., x_n)$ correspondiente a una realización de la variable aleatoria n-dimensional $(X_1, ..., X_n)$ puede calcularse como:

$$P(x_1, ..., x_n) = \prod_{i=1}^{n} P(x_i|pa(x_i)$$

donde x_i representa el valor de la variable X_i, y $pa(x_i)$ representa el valor de los padres de X_i.

En la Figura 2.10 se puede observar un típico ejemplo de red Bayesiana, junto con el significado de los nodos y la factorización que representa.

Redes Bayesianas como paradigma clasificador

Las Redes Bayesianas se han incorporado recientemente a tareas de clasificación supervisada, en base a ideas expuestas por Acid y de Campos, (1994), Friedman y Goldszmidt (1996), y ampliadas en Sierra y Larrañaga (1998), según las cuales se pueden utilizar las factorizaciones de probabilidad representadas por las redes Bayesianas para realizar clasificaciones, considerando para ello la existencia de una variable especial (la variable a clasificar), que viene a ser predicha por un grupo de variables (el resto), de forma que la estructura de red obtenida puede ser utilizada para la predicción del valor de la clase de ésta variable especial, mediante la asignación de valores a las predictoras, y la posterior propagación de la evidencia introducida en la red, esto es, mediante

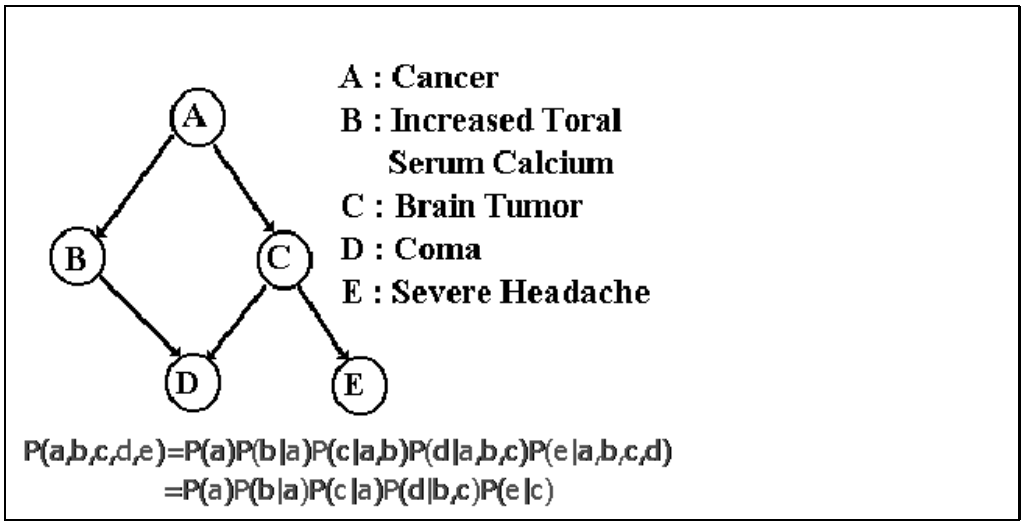


Figura 2.10: Ejemplo de la factorización de la probabilidad que representa una red Bayesiana.

el cálculo de la probabilidad a posteriori del nodo asociado a la variable especial dados los valores del resto.

Es fundamental para realizar esta labor que la estructura de red utilizada sea aprendida con objetivos clasificatorios, esto es, que se haya tenido en cuenta en el proceso de aprendizaje estructural el hecho de que existe una variable cuya probabilidad a posteriori es primordial en el manejo de la red Bayesiana obtenida.

Es de este modo que las redes Bayesianas se han convertido en uno más de entre los denominados Sistemas Expertos Clasificadores, con una utilización clara y sencilla, tal y como se verá en el capítulo siguiente.

Ejemplo

Veamos como se comportan las redes Bayesianas como clasificadores en el ejemplo de la clasificación de la lengua materna de una persona. Existen diferentes modelos que se pueden aplicar. El primero de ellos ya ha sido presentado, ya que el *naive-Bayes* es una red Bayesiana. Otra aproximación consiste en utilizar como base la estructura que ofrece el *naive-Bayes*, pero sin asumir la independencia condicional de las variables predictoras dada la clase, esto es, permitiendo añadir arcos entre los nodos hijos, correspondientes a las variables predictoras. A este modelo se le ha denominado *tree augmented network* (TAN) en la literatura (Friedman y col., 1996). En la Figura 2.11 se presenta un modelo TAN para el ejemplo.

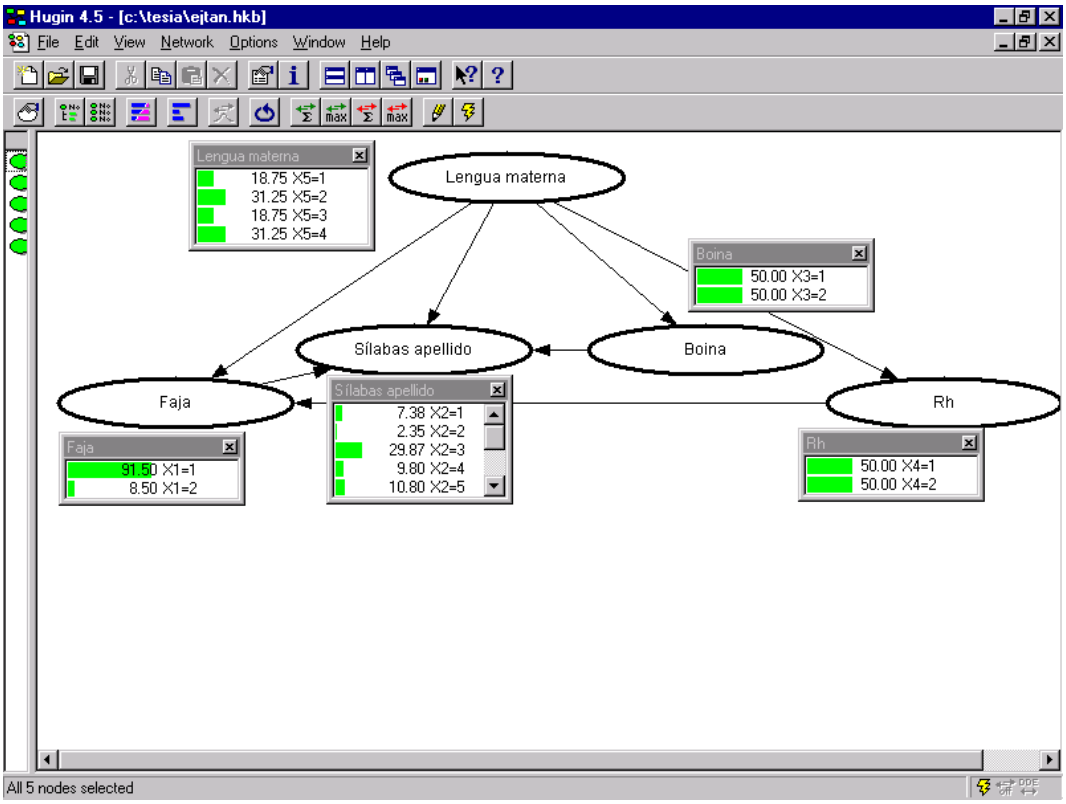


Figura 2.11: Un posible modelo TAN para el ejemplo.

Obviamente este es sólo un ejemplo de un posible modelo TAN para el ejemplo. Existen diferentes alternativas que son todas las que incluyen arcos entre los hijos siempre que no se formen ciclos. Para el ejemplo presentado, como se observa en la Figura 2.12, la clase pronosticada es la primera de ellas, esto es, la *Andaluza*, puesto que tiene una probabilidad a posteriori de 0.5518, siendo la segunda clase más probable la tercera, (*Gallega*), con un valor asociado de 0.2759. Como se puede apreciar, el pronóstico es totalmente diferente al obtenido utilizando el modelo *naive-Bayes*.

Otro modelo de red Bayesiana que se utiliza en aras a obtener un paradigma de clasificación es el denominado *Markov blanquet* (MB) de la variable a clasificar, en el cual los nodos correspondientes a las variables predictoras se disponen de forma que sean:

1. Padres del nodo asociado a la variable a clasificar, o
2. Hijos de éste mismo nodo, o

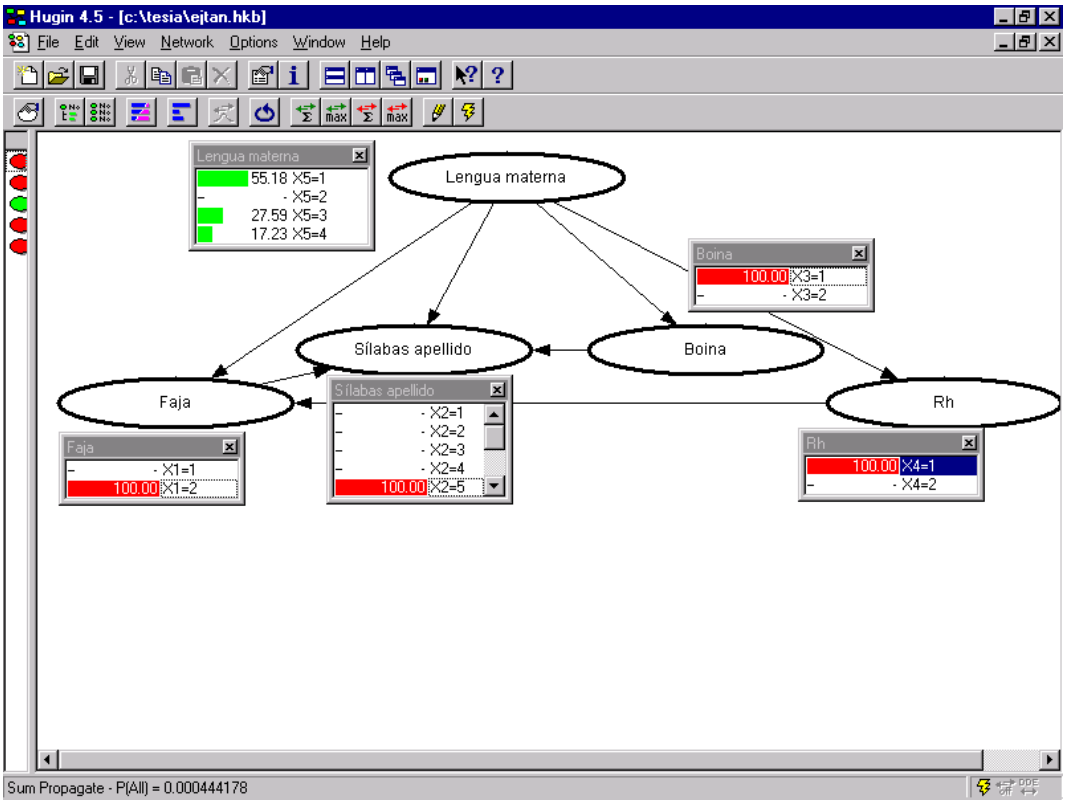


Figura 2.12: Probabilidades a posteriori para el primer caso de ejemplo.

3. Padres de hijos del mismo.

pudiendo adoptar más de una de estas características siempre que no se generen ciclos en la red Bayesiana.

En la Figura 2.13 se aprecia un modelo *Markov blanquet* para el ejemplo de la lengua materna. De nuevo es necesario indicar que son muchos los modelos de red Bayesiana que cumplen las condiciones necesarias para ser MB de la variable a clasificar. En este caso se ha elegido uno de ellos al azar.

Al utilizar este modelo MB para clasificar el caso de ejemplo, se obtienen las probabilidades a posteriori para las clases mostradas en la Figura 2.14. En este caso la clase que se asigna es la segunda, (*Catalana*), ya que su probabilidad a posteriori es la mayor (0.7792).

Existen más aproximaciones (Acid, 1999) para la construcción de modelos de redes Bayesianas

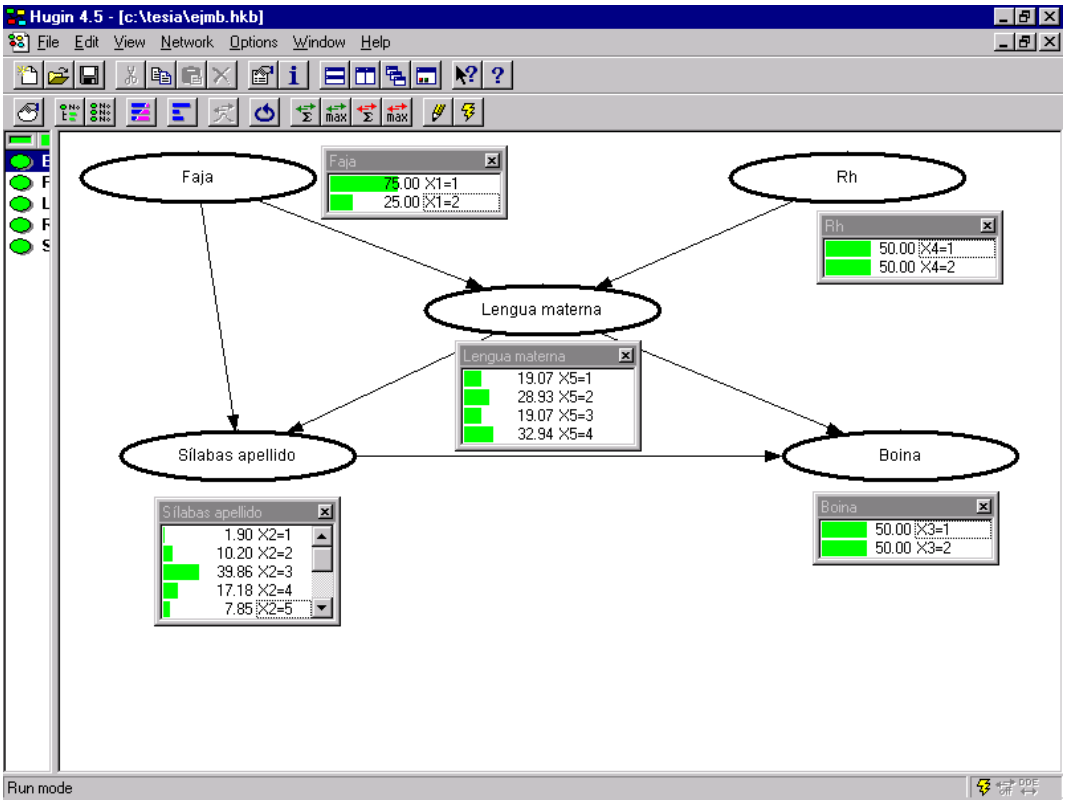


Figura 2.13: Un posible modelo *Markov Blanket* para el ejemplo propuesto.

con objetivos clasificatorios. Algunos de ellos se verán en el capítulo que a las redes Bayesianas se consagra en esta tesis.

2.4 Validación

Estimar la bondad de un clasificador, más conocido como validar, nos sirve para medir su capacidad de predicción sobre nuevas instancias que le lleguen en el futuro para que las clasifique. La validación se realiza habitualmente basándonos en la *tasa de error* del clasificador como representante más habitual de la medida de éxito de éste, entendido el error como la clasificación incorrecta:

$$\text{Tasa de error} = \text{número de errores/número total de casos.}$$

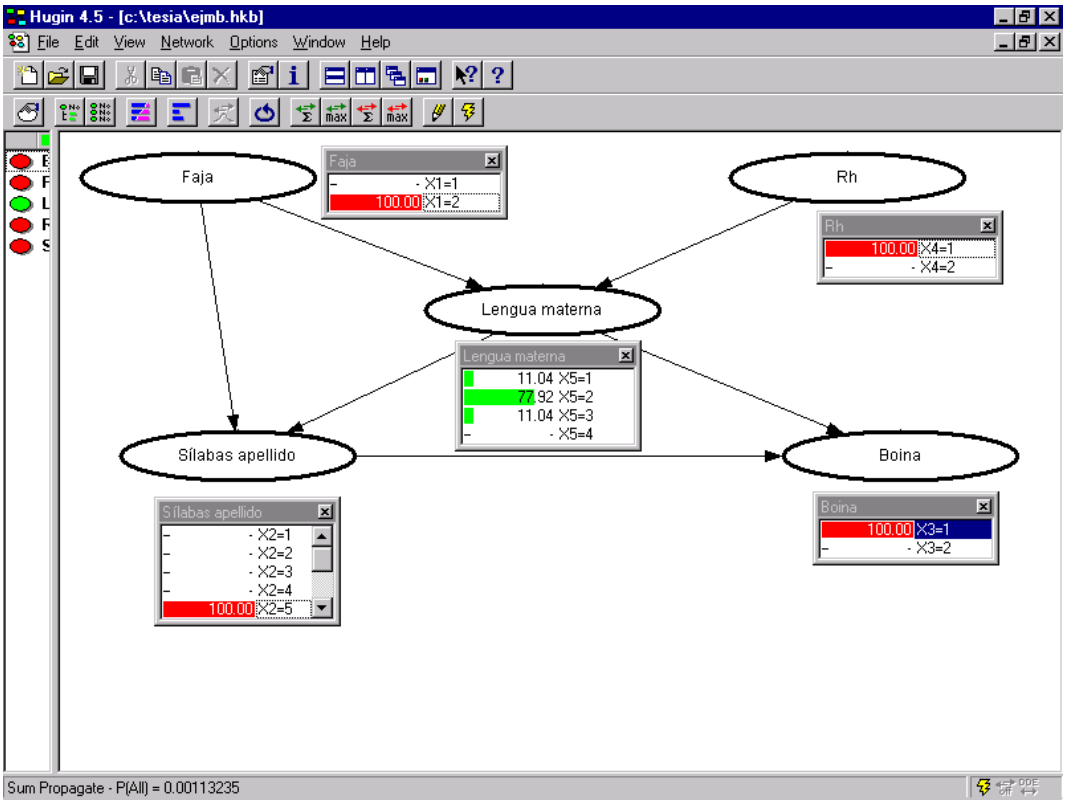


Figura 2.14: Probabilidades a posteriori obtenidas para el caso a clasificar.

Una *matriz de confusión* nos permite ver mediante una tabla de contingencia la distribución de los errores cometidos por un clasificador a lo largo de las distintas categorías del problema. En dicha tabla de contingencia se cruza la variable derivada de la clasificación predecida por el clasificador con la variable que guarda la verdadera clasificación.

Una matriz de confusión para el caso de 2 clases tiene la forma que se puede apreciar en la Tabla 2.8.

En la misma, se tiene que:

1. π_0 denota la probabilidad a priori de la clase 0.
2. π_1 denota la probabilidad a priori de la clase 1; $\pi_1 = 1 - \pi_0$.
3. p_0 indica la proporción de casos que el clasificador predice en la clase 0.

Tabla 2.8: Matriz de confusión para un caso de estudio de dos clases.

		Clase Verdadera		
		0 (+)	1(-)	
Clase predicha	0 (+)	a	b	p_0
	1 (-)	c	d	p_1
		π_0	π_1	N

4. p_1 indica la proporción de casos que el clasificador predice en la clase 1; $p_1 = 1 - p_0$.

5. $N = a + b + c + d$.

De una matriz de confusión también se pueden extraer los siguientes conceptos, enriquecedores a la hora de comprender la distribución y naturaleza de los errores cometidos por nuestro clasificador:

- *Sensibilidad* $Se = a/(a + c)$ proporción de verdaderos positivos
- *Especifidad* $Es = d/(b + d)$ proporción de verdaderos negativos
- Proporción de *falsos positivos* $c/(a + c)$
- Proporción de *falsos negativos* $b/(b + d)$.

Otros dos conceptos importantes en la validación vienen a ser la *tasa de error verdadera* y la *tasa de error aparente*:

- La *tasa de error verdadera* es la probabilidad de que el modelo construido clasifique incorrectamente nuevos casos no utilizados en su construcción. El objetivo de la validación es realizar una estimación lo más honesta posible de esta tasa.
- La *tasa de error aparente* es la tasa de error obtenida por el clasificador a la hora de clasificar las instancias utilizadas en su construcción. Esta tasa es demasiado optimista respecto a la realidad (o tasa de error verdadera), ya que las instancias utilizadas para inducir el modelo suelen adaptarse mejor a él que instancias nuevas no utilizadas en su construcción, conociéndose este fenómeno como 'sobreentrenamiento' u '*overfitting*'.

Teniendo en cuenta los términos expuestos, vayamos ahora a explicar los principales métodos de validación. Esta se realiza utilizando el conjunto de casos que usamos para inducir el clasificador final, esto es, el conjunto de instancias de las cuáles sabemos su verdadera clasificación, bajo el supuesto de que estas provienen de una muestra aleatoria (Dietterich, 1996).

El método *H (Holdout)* es el método más sencillo de validación. Particiona el conjunto de casos en dos grupos: uno de entrenamiento y otro de testeo. El grupo de entrenamiento es usado para inducir un modelo clasificatorio, utilizando el grupo de testeo para estimar la tasa de error verdadera. El grupo de entrenamiento viene a ser habitualmente dos terceras partes del conjunto total de casos, utilizando el resto para el grupo de testeo.

El método de 'Remuestreo' o '*Random Subsampling*' viene a ser una generalización del método H, realizándose éste múltiples veces sobre diferentes particiones independientes del grupo de entrenamiento y grupo de testeo. Así, la estimación de la tasa de error se efectúa a partir de la media de las tasas de error obtenidas en los diferentes experimentos.

El método de la 'Validación cruzada' o '*Cross-Validation*' (Stone, 1974) viene a ser también una generalización del método H. La validación cruzada se basa en la partición de la muestra en K subconjuntos de aproximadamente el mismo tamaño, donde $K-1$ subconjuntos constituyen el grupo de entrenamiento y el restante el grupo de testeo. Al repetir el citado proceso K veces sobre las distintas combinaciones de $K-1$ subconjuntos para entrenar y el restante para testear, la media de las tasas de error de los subconjuntos de testeo se utilizará como estimación de la tasa de error verdadero. Inspirándose en la importancia del parámetro K en la validación cruzada, a menudo se la suele denominar 'validación cruzada de K hojas' o '*K-fold cross validation*'. Un caso particular de la validación cruzada viene a ser el método de validación de 'dejar uno fuera' o '*leave-one-out*', en el cual el parámetro K viene a ser igual al número de instancias que tenemos para inducir el modelo final. De esta forma, los subconjuntos de testeo están formados por una única instancia y los de entrenamiento por la cardinalidad del conjunto total menos esa única instancia llevada al de testeo.

El método de *Bootstrapping* es quizás el de más difícil comprensión. En un conjunto de casos de cardinalidad N, se escoge una muestra aleatoria con reemplazamiento del mismo tamaño como grupo de entrenamiento, dejando los casos no seleccionados como grupo de testeo. Ya que las instancias son escogidas aleatoriamente con reemplazamiento, la probabilidad de que una instancia cualquiera no sea escogida tras N muestras es $(1-\frac{1}{N})^N \approx e^{-1} \approx 0.368$, y el número esperado de instancias distintas en la muestra se aproxima por medio de $0.632N$. Así, denotando por α la tasa de error en el grupo de testeo y si realizásemos b procesos de selección de N muestras aleatorias

con reemplazamiento en cada una de ellas, la estimación de la tasa de error real del Bootstrapping viene a ser:

$$est_{boot} = \frac{1}{b}\sum_{i=1}^{b}(0.632 \cdot \alpha_i + 0.368 \cdot tasa_{tot})$$

donde $tasa_{tot}$ es la tasa de error en el conjunto total de instancias.

2.5 Búsqueda en el espacio de modelos

En el caso de la inducción de reglas de clasificación, los árboles de clasificación y las redes Bayesianas, la construcción[3]del modelo clasificatorio se efectúa habitualmente de una manera voraz y progresiva, también denominada 'greedy'. Esto es, la construcción del modelo clasificatorio se realiza partiendo de una cierta estructura (sea la más simple o la más compleja posible), construyéndose el modelo paso a paso, tomando decisiones acerca del aumento o la disminución de la complejidad del modelo (según sea la dirección en la que éste se construye, como se vio en el apartado 2.1 acerca de la *Dirección de la Modelización*), hasta que se considera que se cumple un criterio preestablecido que lo haga detener.

Esta no es la única forma en la que se puede enfocar la búsqueda del modelo. Otra opción es la de realizar una búsqueda en el espacio de posibles modelos (sea en el espacio de los posibles conjuntos de reglas de clasificación que tratan de resolver el problema planteado, sea en el espacio de los posibles árboles de clasificación, sea el espacio de posibles estructuras de RRBB) que dan una posible solución al problema clasificatorio que se está abordando. Por lo tanto, en vez de construir el modelo paso a paso, se trata de realizar una búsqueda inteligente en el espacio de posibles modelos, siendo cada individuo de nuestra búsqueda, un modelo clasificatorio que trate de resolver el problema que se está estudiando.

Una vez que planteamos el problema de la inducción del modelo en términos de una búsqueda, debemos definir cuatro parámetros implícitos a ésta:

1. *El punto de inicio de la búsqueda.* Puede ser el modelo más simple de todos, el más complejo o cualquier modelo elegido al azar en cualquier punto del espacio de modelos. El punto de partida nos determinará también la dirección en la que se realizará la búsqueda.

2. *La organización de la búsqueda.* Esta viene a determinar la estrategia de búsqueda. Así, podemos visitar todos y cada uno de los puntos del espacio (búsqueda completa), lo cual nos reportará la mejor solución posible para nuestro problema, más conocido como máximo

[3] En los casos del Naive-Bayes y del K-NN no existe una construcción del modelo como tal, sino que únicamente se siguen unas fórmulas específicas para clasificar instancias nuevas.

global (en nuestro caso el mejor modelo para nuestra base de datos, sea un conjunto de reglas de clasificación, sea un árbol de decisión, sea una red Bayesiana). Esta aproximación es inabordable en la realidad, dado el ingente número de modelos posibles que tratan de solventar nuestro problema. Así, surgen los métodos heurísticos de búsqueda, los cuales tratan de utilizar la información que tenemos en cada momento acerca del espacio de búsqueda para no realizar una búsqueda completa e intentar encontrar un punto o solución que, aún no pudiendo garantizarse que sea el máximo global, sea satisfactorio para nuestros intereses (más conocido como máximo local). La literatura relacionada nos enseña numerosos ejemplos de algoritmos de búsqueda que intentan realizar la mejor relación entre el número de soluciones visitadas y la calidad de la mejor solución encontrada por el algoritmo.

3. *Función de evaluación.* La función de evaluación viene a ser una puntuación o 'score' que nos mide la bondad o calidad de la solución que ha visitado nuestro algoritmo de búsqueda. Siendo el objetivo del mismo el encontrar la solución con mayor valor (o máximo global), el algoritmo utilizará el valor de la función de evaluación de cada individuo que haya visitado para guiar la búsqueda. Para nuestro problema concreto de la búsqueda del mejor modelo clasificatorio, la función de evaluación más intuitiva es el porcentaje de bien clasificados obtenido por el modelo.

4. *Criterio para la búsqueda.* Teniendo en cuenta los recursos de máquina que se tienen y las características del algoritmo de búsqueda, habrá que adoptar un criterio para finalizar la búsqueda cuando utilicemos un algoritmo de búsqueda heurístico. Por otro lado, en los algoritmos de búsqueda completa, ésta termina cuando se han visitado todos los posibles puntos del espacio.

Presentamos a continuación las líneas principales de tres algoritmos de búsqueda, dos clásicos en la Inteligencia Artificial: *enfriamiento estadístico* y *algoritmos genéticos* y uno que ha surgido recientemente en el campo de la computación evolutiva: *algoritmos de estimación de distribuciones.*

2.5.1 Enfriamiento estadístico

El Algoritmo de Metrópolis (Metropolis y col., 1953) para problemas de minimización puede ser descrito a partir del pseudocódigo expuesto en la Figura 2.15. Partiendo de un elemento del espacio de búsqueda escogido al azar, el algoritmo salta a otro elemento –perteneciente a un entorno del anterior– siempre y cuando la evaluación en este último punto supere a la del punto anterior. Se trata de un algoritmo de optimización local que presenta serios problemas al tratar de optimizar funciones multimodales.

Algoritmo de Metrópolis

COMIENZO /*Algoritmo de Metrópolis*/
  Seleccionar al azar un elemento i del espacio de búsqueda
  REPETIR
    Generar al azar estado j, vecino de i
    Si $C(j) - C(i) \leq 0$ entonces $i := j$
  HASTA $C(j) \geq C(i)$ para todo vecino de i
FIN /*Algoritmo de Metropólis*/

Figura 2.15: Pseudocódigo del Algoritmo de Metrópolis. $C(i)$ representa el valor de la función de evaluación del estado i.

Una posibilidad de mejorar el Algoritmo de Metrópolis, puede ser ejecutarlo varias veces partiendo de varias soluciones iniciales y tomando el mejor mínimo encontrado.

Kirkpatrick y col. (1983) propusieron un algoritmo de optimización combinatoria, inspirado en el proceso físico de calentamiento y enfriamiento lento en condiciones de cuasiequilibrio, al que llamaron 'enfriamiento estadístico, o *Simulated Annealing*, el cual intenta evitar el quedarse atrapado en óptimos locales por medio de la relajación de la condición de mejora continua impuesta al algoritmo de Metrópolis. En el algoritmo *Simulated Annealing* se permite visitar, bajo criterios probabilísticos, elementos del espacio de búsqueda con peor evaluación que la del punto de donde se parte. Dicha permisividad va reduciéndose en función de un parámetro que –por analogía con el fenómeno físico en el que se inspira el algoritmo– se denomina temperatura.

Buenas introducciones al *Simulated Annealing* pueden encontrarse en Aarts y Korst (1989) y en Aarts y Van Laarhoven (1985).

Dos cuestiones fundamentales para ejecutar el algoritmo *Simulated Annealing* son la definición de una topología en el espacio de búsqueda, así como la determinación del plan de enfriamiento –temperatura inicial, razón de reducción de temperatura, número de iteraciones para cada temperatura, y criterio de parada del algoritmo–.

Un pseudocódigo para el algoritmo *Simulated Annealing*, en problemas de minimización, puede verse en la Figura 2.16.

Algoritmo de enfriamiento estadístico

COMIENZO /* Simulated Annealing */
Seleccionar un estado inicial i del espacio de búsqueda
Seleccionar una temperatura inicial $T > 0$
Posicionar a cero el contador de los cambios de temperatura $t = 0$
REPETIR
Posicionar a cero el contador de las repeticiones $n = 0$
REPETIR
Generar un estado j vecino del estado i
Calcular $\delta = C(j) - C(i)$
Si $\delta < 0$ *entonces* $i := j$
sino si random(0,1) $\leq exp(-\delta/T)$ *entonces* $i := j$
$n := n + 1$
HASTA QUE $n = N(t)$
$t := t + 1$
$T := T(t)$
HASTA QUE se satisfaga la condición de parada
FIN /* Simulated Annealing */

Figura 2.16: Pseudocódigo del Algoritmo de enfriamiento estadístico.

2.5.2 Algoritmos genéticos

Introducción

Los *algoritmos genéticos* (AAGG) (Holland, 1975) son una de las técnicas de resolución de problemas de optimización más conocidas. En realidad son algoritmos adaptativos, que se han utilizado con éxito en problemas combinatorios o de búsqueda. Los AAGG están basados en poblaciones. En primer lugar, se obtiene de algún modo una primera población, compuesta por individuos que son, cada uno de ellos, una solución candidata al problema planteado; a continuación, se seleccionan individuos de ésta población, y a partir de una combinación de los mismos (mediante unos operadores de cruce y mutación) se obtienen los nuevos individuos que pasarán a formar parte de la siguiente población.

Mientras el algoritmo '*Simulated Annealing*' realiza la búsqueda intentando mejorar una única solución, los algoritmos genéticos trabajan con un grupo o población de posibles soluciones. La evolución de dichas soluciones hacia valores óptimos del problema depende en buena medida de una adecuada codificación de las mismas.

Los principios básicos de los algoritmos genéticos fueron establecidos por Holland (1975), y

se encuentran bien descritos en varios textos –Goldberg (1989), Davis (1991), Reeves (1993), Michalewicz (1994)–.

La potencia de los algoritmos genéticos proviene de la robustez de la técnica, lo que les permite abordar un gran número de problemas con garantías de encontrar una solución aceptable. No se garantiza que se encuentre la solución óptima, pero si una *cuasióptima* en un tiempo razonable en comparación con otras técnicas de búsqueda global. Existen muchas posibilidades de hibridar los AAGG con otro tipo de técnicas específicas para el problema a tratar.

El Algoritmo Genético Simple

Algoritmo Genético Simple

```
COMIENZO /* Algoritmo Genético Simple */
   Generar una población inicial.
   Computar la función de evaluación de cada individuo.
   MIENTRAS NO Terminado HACER
   COMIENZO /* Producir nueva generación */
      PARA Tamaño población/2 HACER
      COMIENZO /*Ciclo Reproductivo */
               Seleccionar dos individuos de la anterior generación,
               para el cruce (probabilidad de selección proporcional
               a la función de evaluación del individuo).
               Cruzar con cierta probabilidad los dos
               individuos obteniendo dos descendientes.
               Mutar los dos descendientes con cierta probabilidad.
               Computar la función de evaluación de los dos
               descendientes mutados.
               Insertar los dos descendientes mutados en la nueva generación.
      FIN
      SI       la población ha convergido ENTONCES
               Terminado := TRUE
   FIN
FIN
```

Figura 2.17: Pseudocódigo del Algoritmo Genético Simple.

El Algoritmo Genético Simple, también denominado Canónico, se representa en la Figura 2.17. Se necesita una codificación o representación del problema, que resulte adecuada al mismo. Además se requiere una función de ajuste ó adaptación al problema, la cual asigna un número real a cada posible solución codificada. Durante la ejecución del algoritmo, los padres deben ser seleccionados para la reproducción, a continuación dichos padres seleccionados se cruzarán

generando dos hijos, sobre cada uno de los cuales actuará un operador de mutación. El resultado de la combinación de los anteriores operadores será un conjunto de individuos (posibles soluciones al problema), los cuales en la evolución del algoritmo genético formarán parte de la siguiente población.

Codificación

Las posibles soluciones al problema deben poder codificarse como una cadena elementos, a los que se denomina 'genes' tratando de mantener una nomenclatura relacionada con la biología. La forma más simple que puede adoptar esta cadena es la de cadena binaria o de bits, en la cual cada uno de los posibles genes adopta un valor que puede ser o 0 o 1. En base a la representación elegida, se debe diseñar una función que evalúe la bondad de una cadena data al problema en estudio. A esta función se le denomina *función de adaptación*. Mediante esta función se asocia un valor a cada una de las cadenas o individuos que forman la población.

Una vez generada la población inicial, normalmente al azar, comienza a simularse una evolución en la misma, utilizándose para ello una fase reproductiva, en la que se seleccionan individuos para hacer el papel de progenitores, y se *cruzan* produciendo descendientes que serán los que formen la siguiente población –o la siguiente generación–. Los individuos de la población tienen una probabilidad de ser elegidos para el cruce proporcional a su propia función de adaptación, de modo que los individuos mejor adaptados al problema tendrán una probabilidad mayor de ser elegidos. Se dice que la elección de los individuos se basa en la ruleta sesgada.

El cruce se realiza mediante los llamados *operadores de cruce*. A continuación se suele efectuar, con alguna probabilidad, una mutación en los individuos obtenidos mediante el cruce. Para ello se utilizan los *operadores de mutación*. Las formas más básicas de estos dos tipos de operadores se presentan a continuación.

El *operador de cruce basado en un punto*, a partir de dos padres seleccionados, corta sus ristras de cromosomas en una posición escogida al azar, para producir dos subristras iniciales y dos subristras finales. Después se intercambian las subristras finales, produciéndose dos nuevos cromosomas completos. Ambos descendientes heredan genes de cada uno de los padres. En la Figura 2.18 se muestra un ejemplo de la forma de actuar de este operador.

El operador de cruce no se aplica a todos los pares de progenitores seleccionados, sino que se aplica con una probabilidad, normalmente alta –mayor que 0.5–. En el caso de que no se aplique, la descendencia se genera clonando los progenitores.

El *operador de mutación* se aplica a cada hijo de manera individual, y consiste en la alteración aleatoria –con una probabilidad pequeña, menor que 1.0 habitualmente– de cada uno de los genes que lo componen. La Figura 2.19 muestra un ejemplo de mutación simple.

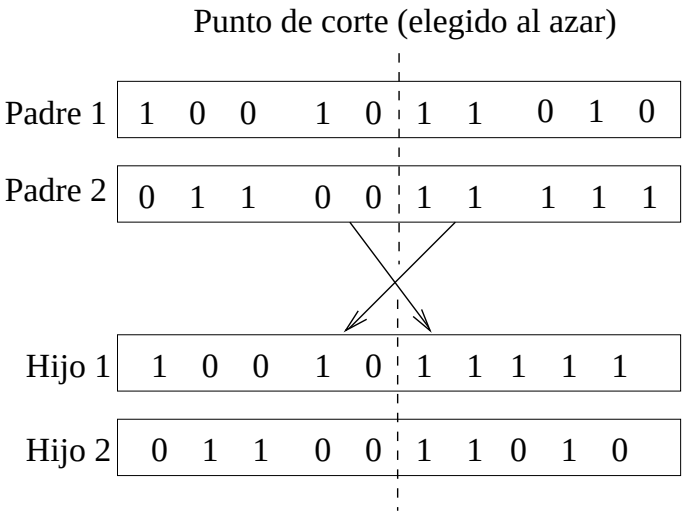


Figura 2.18: Operador de cruce basado en un punto.

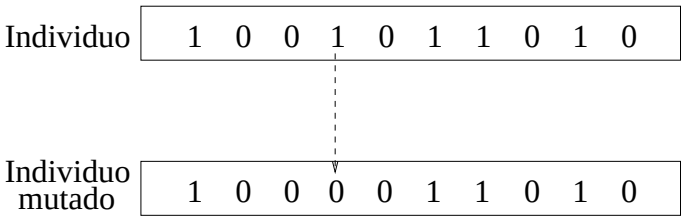


Figura 2.19: Operador de mutación.

Si bien puede en principio pensarse que el operador de cruce es más importante que el operador de mutación, ya que proporciona una exploración rápida del espacio de búsqueda, éste último asegura que ningún punto del espacio de búsqueda tenga probabilidad cero de ser examinado, y es de capital importancia para asegurar la convergencia de los Algoritmos Genéticos.

Es necesario establecer una serie de criterios para determinar la finalización del algoritmo genético. Un criterio de parada puede ser un número máximo de ciclos establecido de antemano, otro el hecho de que se lleven varias generaciones –nuevas poblaciones– sin mejorar el óptimo encontrado, otra el hecho de que la media de las funciones de adaptación de los individuos que forman la población se halle muy cercana –a un diez por ciento, por ejemplo– del valor del mejor individuo de la población. Una alternativa a esta última es la introducida por De Jong (1975), en su tesis doctoral: un gen ha convergido cuando al menos el 95 % de los individuos de la población comparten el mismo valor para dicho gen. Se dice que la población converge cuando todos los genes han convergido. Se puede generalizar dicha definición al caso en que al menos un $\beta\%$ de los individuos de la población hayan convergido. En la Figura 2.20 se ve la adaptación media y del mejor individuo de la población. Un valor aproximado en ambas adaptaciones puede ser tomado como criterio de parada del algoritmo.

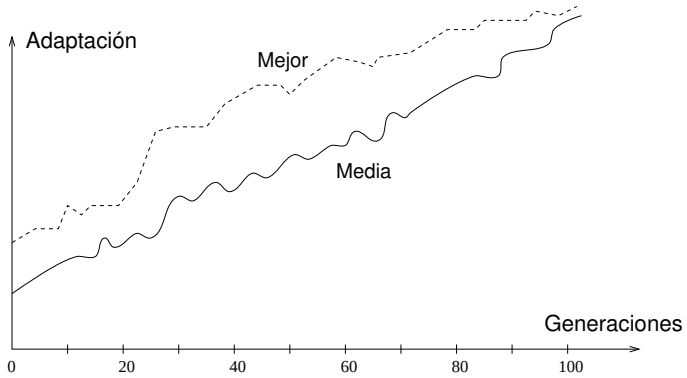


Figura 2.20: Adaptación media y mejor adaptación en un Algoritmo Genético Simple.

Ejemplo

Se presentan a continuación los conceptos presentados mediante un ejemplo, que consiste en hallar el máximo de la función $f(x) = 320 + (16 - x)(x - 16)$, siendo x un número entero perteneciente al conjunto $\{1, 2, \ldots, 32\}$. Evidentemente para lograr dicho óptimo, bastaría actuar por búsqueda

Tabla 2.9: Población inicial de la simulación efectuada a mano correspondiente al Algoritmo Genético Simple.

	Población inicial (fenotipos)	x valor genotipo	$f(x)$ valor (función adaptación)	$f(x)/\sum f(x)$ (probabilidad selección)	Probabilidad de selección acumulada
1	11110	30	124	0.13	0.13
2	00100	8	256	0.27	0.40
3	10111	23	271	0.28	0.68
4	01100	12	304	0.32	1.00
Suma			955		
Media			238.75		
Mejor			304		

exhaustiva, dada la baja cardinalidad del espacio de búsqueda. Se trata por tanto de un mero ejemplo con el que pretendemos ilustrar el comportamiento del algoritmo anteriormente descrito.

Suponiendo que el alfabeto utilizado para codificar los individuos sea $\{0, 1\}$, necesitaremos ristras de longitud 5 para representar los 32 puntos del espacio de búsqueda.

En la Tabla 2.9, hemos representado los 4 individuos que constituyen la población inicial, junto con su función de adaptación al problema, así como la probabilidad de que cada uno de dichos individuos sea seleccionado como progenitor, que se ha supuesto directamente proporcional a su función de evaluación.

Seleccionamos a continuación dos parejas de progenitores con el objeto de que al reproducirse formen 4 nuevos individuos. Para ello es suficiente, con obtener 4 números provenientes de una distribución uniforme $[0, 1]$, y compararlos con la última columna de la Tabla 2.9. Así por ejemplo, supongamos que dichos 4 números hayan sido: 0.55; 0.82; 0.30 y 0.66. Esto significa que los individuos seleccionados para el cruce han sido: el individuo 3 junto con el individuo 4, por un lado, y el individuo 2 junto con el individuo 3, por el otro.

Al tratarse de un ejemplo simple, realizamos el cruce en las dos parejas. Para ello escogeremos un número al azar entre 1 y $l - 1$ (siendo l la longitud de la ristra utilizada para representar el individuo). Notése que la restricción impuesta al escoger el número entre 1 y $l - 1$, y no l, se realiza con la finalidad de que los descendientes no coincidan con los padres.

Supongamos, tal y como se indica en la Tabla 2.10, que los puntos de cruce resulten ser 3 y 2. De esta manera obtendríamos los 4 descendientes descritos en la tercera columna de la Tabla

Tabla 2.10: Población en el tiempo 1, proveniente de efectuar los operadores de cruce y mutación sobre los individuos expresados en la tabla 2.9, los cuales constituyen la población en el tiempo 0.

Emparejamiento de los individuos seleccionados	Punto de cruce	Descen-dientes	Nueva población descendientes mutados	x valor genotipo	$f(x)$ función adaptación
10111	3	10100	10100	20	304
01100	3	01111	01111	15	319
00100	2	00111	10111	23	271
10111	2	10100	10100	20	304
Suma					1198
Media					299.5
Mejor					319

2.10. A continuación mutaríamos con una probabilidad, p_m, cercana a cero, cada uno de los bit de las cuatro ristras de individuos. En este caso suponemos que el único bit mutado corresponde al primer gen del tercer individuo. En las dos últimas columnas se pueden consultar los valores de los individuos, así como las funciones de adaptación correspondientes. Como puede observarse, tanto el mejor individuo como la función de adaptación media han mejorado sustancialmente al compararlos con los resultados de la Tabla 2.9.

2.5.3 Algoritmos de estimación de distribuciones de probabilidad

A pesar de que los AAGG obtienen resultados muy buenos en muchas de las áreas en que son utilizados, existen algunos campos en los que la utilización de los mismos no aporta beneficios. Este tipo de problemas, que reciben el nombre de *deceptivos*, atraen la atención de muchos investigadores en el área de la computación evolutiva, que realizan esfuerzos con el objetivo de paliar de algún modo las limitaciones que presentan los AAGG.

Un algoritmo posterior, inspirado en principio en los AAGG, es el *algoritmo de estimación de distribuciones de probabilidad*, (Estimation of Distribution Algorithm, EDA) (Mühlenbein y Paaß, 1996). En la aproximación EDA, no se utilizan los operadores de cruce y mutación. La siguiente población es muestreada a partir de la distribución de probabilidad que subyace en los individuos seleccionados de la población actual. En la Figura 2.21 se muestra el esquema de actuación de este tipo de algoritmos.

Estimation of Distribution Algorithm

Comienzo EDA
  $D_0 \leftarrow$ Generar N individuos (la población inicial) de forma aleatoria
  REPETIR para $l = 1, 2, \ldots$ HASTA QUE se cumpla el criterio de parada
    $D^s_{l-1} \leftarrow$ Seleccionar $S \leq N$ individuos de D_{l-1}
    basándose en algún método de selección
    $p_l(\mathbf{x}) = p(\mathbf{x}|D^s_{l-1}) \leftarrow$ Estimar la distribución de probabilidad conjunta de los individuos seleccionados
    $D_l \leftarrow$ Muestrear N individuos (la nueva población) a partir de $p_l(\mathbf{x})$.
Fin EDA

Figura 2.21: Esquema principal de la aproximación EDA.

De esta forma se puede realizar una búsqueda aleatorizada basada en la evolución, utilizando información probabilista para guiar el proceso. Los algoritmos EDA reemplazan los operadores de cruce y mutación realizando para ello los siguientes dos pasos:

1. se construye un modelo probabilista sobre las soluciones elegidas para ello,
2. las nuevas soluciones se generan de acuerdo al modelo probabilista construido.

El problema principal de la aproximación EDA reside en cómo se estima la distribución de probabilidad subyacente en los datos, $p_l(\mathbf{x})$. Obviamente, el cálculo de 2^n probabilidades (para un problema que tenga n variables binarias) es impracticable. Esto ha llevado a la presentación de aproximaciones donde se asume que la distribución de probabilidad puede ser factorizada en base a un modelo de probabilidad (Larrañaga y col., 1999, Pelikan y col., 1999).

La manera más sencilla de estimar la distribución de los casos seleccionados, es asumir independencia entre las variables[4] predictoras que forman parte del dominio. Las nuevas soluciones candidatas son muestreadas fijándose solamente en las proporciones de los valores de cada variable independientemente del resto. Population Based Incremental Learning (PBIL, Baluja 1994)y Bit-Based Simulated Crossover (BSC, Syswerda 1993) son dos de los algoritmos de este tipo que se pueden encontrar en la literatura.

El algoritmo Factorized Distribution Algorithm (FDA, Mühlenbein y col. 1999) cubre órdenes mayores de interacción. Esto se consigue utilizando una factorización de la distribución de probabilidad conjunta fijada de antemano. En cualquier caso, FDA necesita información previa sobre la descomposición y factorización del problema que no suele estar disponible.

[4] En la comunidad de investigadores dedicados a la *computación evolutiva*, se utiliza el término variable, mientras que en otros campos se habla de características. En esta memoria se utilizarán ambos términos como si fueran sinónimos.

Sin la necesidad de tener esta información extra, y teniendo en cuenta que las redes Bayesianas son representaciones gráficas que cubren un mayor grado de interacciones entre las variables, se han propuesto aproximaciones que las utilizan para estimar la distribución conjunta de las soluciones seleccionadas. Así, los algoritmos EBNA (Etxeberria y Larrañaga, 1999) y BOA (Pelikan y col., 1999) utilizan esta idea para tratar de representar interacciones múltiples entre variables.

En el trabajo desarrollado en esta tesis, se ha utilizado una versión del algoritmo Population Based Incremental Learning (PBIL, Baluja 1994), que será descrito posteriormente.

Parte III

REDES BAYESIANAS

Capítulo 3

Redes Bayesianas

3.1 Un poco de historia

Las redes Bayesianas (RRBB) deben su nombre al Reverendo Thomas Bayes[1](Londres, 1702 - Tunbridge, Kent, 1761), cuyo retrato es mostrado en la Figura 3.1. Fue uno de los seis primeros reverendos protestantes ordenados en Inglaterra. Educado de forma privada, no se conoce el nombre de su tutor, aunque algunas investigaciones apuntan a que este pudiera haber sido *de Moivre*.

Comenzó como ayudante de su padre (párroco también) en Holborn, y a finales de 1720 fue nombrado pastor en la capilla prebisteriana de Tunbridge Wells (Kent), a unas 35 millas al sureste de Londres. Abandonó los hábitos en 1752, aunque continuó viviendo en el mismo lugar.

Publicó sus teorías en un artículo titulado *Essay towards solving a problem in the doctrine of chances*, publicado por *The philosophical Transactions of the Royal Society of London*, en 1764. Las conclusiones por él presentadas fueron aceptadas por Laplace en una memoria de 1781, redescubiertas por Condorcet, y permanecieron inalterables hasta que Boole las cuestionó en su libro *Laws of Thought* (1854). Desde aquel momento, las técnicas de Bayes vienen siendo sometidas a controversia, habiéndose creado una familia de estadísticos que las apoyan fielmente, los denominados Bayesianos.

Bayes fue elegido miembro de la Royal Society en 1742, a pesar de no tenía por aquella época ninguna publicación en el área de las matemáticas. De hecho, no se publicó nada a su nombre mientras vivió, ya que enviaba sus trabajos de forma anónima.

[1]Se pueden encontrar notas bibliográficas sobre la mayoría de los científicos famosos en la dirección de Internet `http://www.vma.bme.hu/mathhist/BiogIndex.html`. En concreto, la información sobre el reverendo Thomas Bayes se ha extraído de la dirección `http://www.vma.bme.hu/mathhist/Mathematicians/Bayes.html`. Otra versión se puede encontrar en la dirección `http://www.stat.ucl.ac.be/ISpersonnel/beck/bayes.html`.

Figura 3.1: Reverendo Bayes (1702-1761).

3.1.1 El teorema de Bayes

De entre los descubrimientos que el reverendo Thomas Bayes realizó, nos vamos a centrar en el llamado teorema o fórmula de Bayes, que es en el que se basan las RRBB. Para presentarlo, se introducen brevemente unas nociones de probabilidad condicional, necesarias para dicho teorema.

Probabilidad condicional

La probabilidad de que el enunciado B sea verdadero, condicionado a que el enunciado A sea verdadero, se denota por $p(B \mid A)$, y se define, en caso de que $p(A) > 0$, como:

$$p(B \mid A) = p(A\&B)/p(A).$$

La probabilidad condicional se interpreta como una implicación especial, que es lo característico del razonamiento probabilístico, entendiéndose que "si A es verdadero, entonces B tiene una probabilidad $p(B \mid A)$ de ser verdadero".

A partir de la denominada *regla de la multiplicación* se puede expresar la probabilidad correspondiente a la intersección de un número finito de enunciados, por medio del producto de

probabilidades condicionadas, de la manera siguiente:

$$p(A_1\&A_2\&A_3\&...\&A_n) = p(A_n) \cdot \prod_{j=1}^{n-1} p(A_j \mid A_{j+1}\&...\&A_n).$$

Fórmula de Bayes

La definición de probabilidad condicionada, anteriormente expuesta, tiene dos versiones, en función de cual sea el enunciado que condiciona. Así se tiene que si A y B son dos enunciados tales que $p(A) > 0$ y $p(B) > 0$, se verifica que:

$$p(B \mid A) = p(A\&B)/p(A).$$

$$p(A \mid B) = p(A\&B)/p(B).$$

De las anteriores igualdades se obtiene que:

$$p(A \mid B) = \frac{p(B \mid A) \cdot p(A)}{p(B)}.$$

Además si aplicamos el denominado teorema de la probabilidad total al enunciado B, se tendría:

$$p(B) = p(B \mid A) \cdot p(A) + p(B \mid \neg A) \cdot p(\neg A).$$

De las dos últimas igualdades se obtiene, la denominada fórmula de Bayes:

$$p(A \mid B) = \frac{p(B \mid A) \cdot p(A)}{p(B \mid A) \cdot p(A) + p(B \mid \neg A) \cdot p(\neg A)}.$$

Razonamiento probabilístico

Mediante el teorema enunciado, es posible realizar lo que se ha dado en llamar razonamiento probabilístico, algunas de cuyas características son las siguientes[2]:

- El sistema deberá ser capaz de manejar desigualdades a la hora de realizar razonamientos encadenados.

  Si suponemos que tratamos de determinar el valor del enunciado B, una vez conocidos los valores asignados a $p(A)$ y $p(B \mid A)$, a partir del teorema de la probabilidad total, se tiene que:

  $$p(B) = p(A) \cdot p(B \mid A) + p(\neg A) \cdot p(B \mid \neg A).$$

[2] estas características son independientes del concepto de red Bayesiana.

Al no conocerse el valor de $p(B \mid \neg A)$, de la fórmula anterior se obtiene que:

$$p(B) \geq p(A) \cdot p(B \mid A).$$

Análogamente se deduce que:

$$p(\neg B) \geq p(A) \cdot p(\neg B \mid A).$$

De las dos desigualdades anteriores, se obtiene:

$$p(A) \cdot p(B \mid A) \leq p(B) \geq 1 - (p(A) \cdot (1 - p(B \mid A))).$$

- Posibilidad de utilizar otros esquemas de razonamiento que denominaremos *plausibles*, no aceptables en la lógica bivalente.

  Ejemplos de lo anterior son los siguientes esquemas:
  (a) Si se sabe que $p(B \mid A) = 1$, y A es falso, entonces parece que B tiene que ser menos creíble.
  (b) Si se sabe que $p(B \mid A) = 1$, y B es verdadero, entonces parece que A tiene que ser más creíble.

- Carácter *no monótono*. Es decir a medida que obtenemos nueva información (hechos) sobre el problema, los valores de incertidumbre de los enunciados varían.

- A partir de la fórmula de Bayes, se deduce que la implicación probabilística es *bidireccional*.

3.2 Presentación intuitiva

Antes de mostrar sucintamente los conceptos matemáticos que sustentan las redes Bayesianas, vamos a presentarlas de un modo intuitivo. En Pearl (1988), Neapolitan (1990), Jensen (1996), Castillo y col. (1997), Gámez y Puerta (1998), Díez y Nell (1998), o Cowell y col. (1999) se pueden encontrar buenas presentaciones de lo que son la redes Bayesianas.

En esta sección se presentarán los conceptos básicos de las RRBB a través del ejemplo que se ha venido utilizando en el capítulo anterior para introducir los paradigmas de clasificación supervisada.

Al igual que en la clasificación (supervisada o no), en el mundo de las RRBB disponemos de una serie de variables, $\{X_1, ..., X_n\}$, y queremos representar las relaciones que existen entre las mismas. Digamos, en primer lugar, que una red Bayesiana está compuesta por dos partes: la parte gráfica y la parte numérica. La representación gráfica se realiza mediante un grafo acíclico dirigido, en el que los nodos se corresponden a las variables del modelo, y en el que aparecen una

serie de conexiones en forma de flecha, que denominaremos *enlaces* o *arcos*, entre los diferentes nodos, que reflejan las relaciones existentes entre las variables a las que éstos representan.

La red Bayesiana no trivial más simple consta de dos variables asociadas mediante un arco. Supongamos que estas dos variables son las correspondientes a *Faja* y *Clase* en el ejemplo de la lengua materna. La Figura 3.2 presenta la representación gráfica asociada a una relación entre ambas variables en la que se ha supuesto que la variable *Faja* es madre[3] de la variable *Clase*.

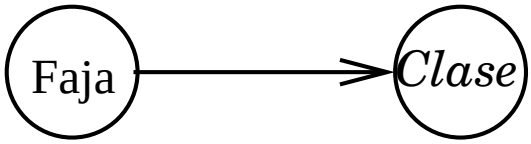


Figura 3.2: Red Bayesiana compuesta por dos nodos.

Debemos establecer la parte numérica asociada a la representación gráfica de red Bayesiana mostrada en la Figura 3.2. Para ello, es necesario conocer, en primer lugar, las probabilidades a priori de los valores de las variables asociadas a nodos que no tienen padres. Estimándolas a partir de la base de datos de ejemplo de la Tabla 2.1, en este caso podemos asumir que, para la variable que no tiene madre, estas probabilidades a priori son:
$P(Faja = No) = 0.75$
$P(Faja = Si) = 0.25$
ya que de los 16 casos de que consta la base de datos, en 12 la variable vale *No* y en los otros 4 la variable toma el valor *Si*.

Necesitamos también conocer (estimar) las probabilidades condicionadas de los valores de la variable hija respecto a los valores de la variable madre, esto es, los valores de $P(Clase|Faja)$. Basándonos de nuevo en la base de datos, estos valores se estiman contando los casos en los que se dan los valores de la variable para cada uno de los posibles valores de la variable a la que la estamos condicionando:

- $Clase = Andaluza$
  - $P(Clase = Andaluza \mid Faja = No) = 0.25$ (en tres de los doce casos en los que la variable *Faja* toma el valor *No*, la variable *Clase* toma el valor *Andaluza*),

[3] Se trata de forma indiferente a las variables que a los nodos, al considerarse que son equivalentes. Cuando existe un arco de un nodo a otro, decimos que el primer nodo es padre del segundo, o, lo que es lo mismo, que la variable asociada al primer nodo es madre de la asociada al segundo.

  - $P(Clase = Andaluza \mid Faja = Si) = 0.00$ (en cero de los cuatro casos en los que la variable *Faja* toma el valor *Si*, la variable *Clase* toma el valor *Andaluza*),

- $Clase = Catalana$
  - $P(Clase = Catalana \mid Faja = No) = 0.0833$
  - $P(Clase = Catalana \mid Faja = Si) = 1.00$
- $Clase = Gallega$
  - $P(Clase = Gallega \mid Faja = No) = 0.25$
  - $P(Clase = Gallega \mid Faja = Si) = 0.00$
- $Clase = Euskera$
  - $P(Clase = Euskera \mid Faja = No) = 0.4167$
  - $P(Clase = Euskera \mid Faja = Si) = 0.00$.

Se puede observar que, como cabía esperar, se cumple que:

$$P(Clase = Andaluza \mid Faja = No)+P(Clase = Catalana \mid Faja = No)+P(Clase = Gallega \mid Faja = No)+$$

$$P(Clase = Euskera \mid Faja = No) = 0.25 + 0.0833 + 0.4167 + 0.25 = 1$$

y que

$$P(Clase = Andaluza \mid Faja = Si)+P(Clase = Catalana \mid Faja = Si)+P(Clase = Gallega \mid Faja = Si)+$$

$$+P(Clase = Euskera \mid Faja = Si) = 0.00 + 1.00 + 0.00 + 0.00 = 1$$

Aplicando el teorema de la probabilidad total, se puede calcular la probabilidad a priori de cada uno de los valores de la variable hija, que según la base de datos son: $P(Clase = Andaluza) = \frac{3}{16}$, $P(Clase = Catalana) = \frac{5}{16}$, $P(Clase = Gallega) = \frac{3}{16}$ y $P(Clase = Euskera) = \frac{5}{16}$.

Así, por ejemplo:
$P(Clase = Andaluza) = P(Clase = Andaluza \mid Faja = No) \cdot P(Faja = No)+$
$+P(Clase = Andaluza|Faja = Si) \cdot P(Faja = Si) = 0.25\frac{12}{16} + 0.00\frac{4}{16} = \frac{3}{16}$.

Nos interesa particularmente establecer un método para calcular la probabilidad a posteriori de una variable dado que se ha evidenciado el valor de la otra. Esto se puede realizar tanto en un sentido como en el otro, esto es, conocido el valor de la variable padre podemos querer saber cuales son las probabilidades a posteriori de la variable hija y viceversa.

Aplicando el teorema de Bayes, podemos obtener estos valores. Por ejemplo, supongamos que se sabe que el caso que nos ocupa es de la clase *Catalana*. Veamos cual será la distribución de probabilidad a posteriori de la variable *Faja*:

$P(Faja = No \mid Clase = Catalana) = \frac{P(Faja=No)P(Clase=Catalana|Faja=No)}{P(Clase=Catalana)} = \frac{0.75\times 0.0833}{0.3125} = 0.20.$

$P(Faja = Si \mid Clase = Catalana) = \frac{P(Faja=Si)P(Clase=Catalana|Faja=Si)}{P(Clase=Catalana)} = \frac{0.25\times 1.00}{0.3125} = 0.80.$

Estructura de la red. Teoría de grafos

La estructura de las redes Bayesianas viene representada por un grafo. Presentamos algunas nociones básicas sobre teoría de grafos que pueden ser de utilidad para comprenderlas mejor.

Un grafo viene definido por un par (G, A), en donde G representa los nodos que lo forman y V es el conjunto de aristas que hay entre los mismos. Si las aristas son dirigidas, se denominan arcos dirigidos, o simplemente arcos.

Un grafo dirigido es un par (N, A) en donde todas las aristas son arcos dirigidos.

Un nodo X es padre de un nodo Y si existe un arco que va de X a Y, $X \to Y$. Se dice también que Y es hijo de X.

Se denomina familia del nodo X al conjunto formado por X y los padres de éste, $pa(X)$.

Un camino es una sucesión de nodos $\{X_1, \ldots, X_L\}$ pertenecientes a un grafo (N, A), de modo que $X_i \neq X_j$ para $1 < i < j < L$ y

$$(X_i, X_{i+1}) \in A \quad ó \quad (X_{i+1}, X_i) \in A, \forall i = 1, \ldots, L-1$$

Un ciclo es una sucesión de nodos $\{X_1, \ldots, X_L\}$ pertenecientes al grafo (N, A) tal que $X_i \neq X_j$ para $1 < i < j < L$, y $\forall i < L-1 \ \exists (X_i, X_{i+1}) \in A$ y existe el arco $(X_L, X_1) \in A$ que cierra el ciclo.

La Figura 3.3 muestra un ejemplo de un grafo dirigido.

En la Figura 3.4 se muestra un ejemplo de un ciclo de longitud 4.

Los grafos acíclicos dirigidos juegan un papel central en la teoría de las redes Bayesianas. En lo que sigue se introducen los conceptos básicos sobre los mismos.

Sea $G = (N, A)$ un grafo (dirigido o no dirigido). Se dice que G es acíclico si G no contiene ciclos.

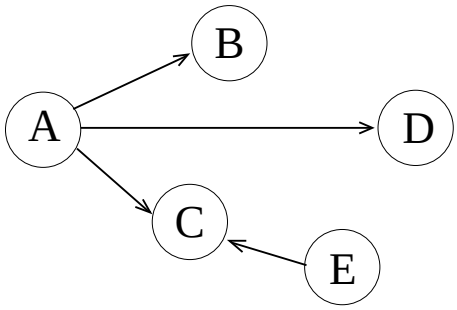


Figura 3.3: Ejemplo de grafo dirigido.

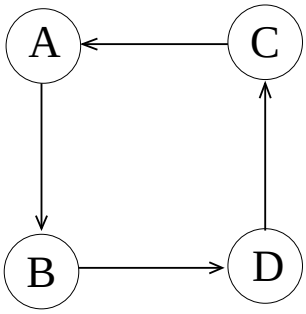


Figura 3.4: Ejemplo de ciclo de longitud 4.

Un grafo $G = (N, A)$ es un grafo acíclico dirigido (en inglés *directed acyclic graph*, DAG) si G es acíclico y dirigido.

Sea $G = (N, A)$ un grafo dirigido. Sean u y v vértices en N. Se dice que u es un padre de v, o bien que v es un hijo de u, si y sólo si $(u, v)\epsilon R$. Se dice que u es un ancestro de v, o bien que v es un descendiente de u, si existe un camino de u a v. Un vértice sin padres se denomina vértice raíz.

La Figura 3.5 ilustra los conceptos anteriores.

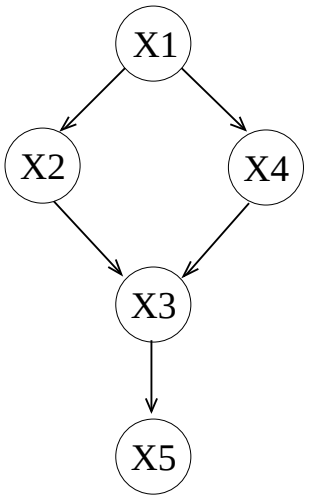


Figura 3.5: Grafo acíclico dirigido. X_2 y X_4 son padres de X_3 y ancestros de X_5. X_1 es un vértice raíz.

Un DAG, $G = (N, A)$, está simplemente conectado si dados dos vértices cualesquiera, $u, v \in N$, existe como máximo un camino entre u y v. En caso contrario se dice que el DAG está múltiplemente conectado.

En el contexto de redes Bayesianas, a los grafos acíclicos simplemente conectados, habitualmente se les denomina poliárboles.

Un DAG, $G = (N, A)$, se denomina un bosque, si cada vértice $v\epsilon N$ tiene como máximo un padre.

Un bosque se denomina árbol si exactamente un vértice no tiene padres. Dicho vértice es la raíz del árbol.

En la Figura 3.6 se muestran diferentes tipos de estructuras DAG.

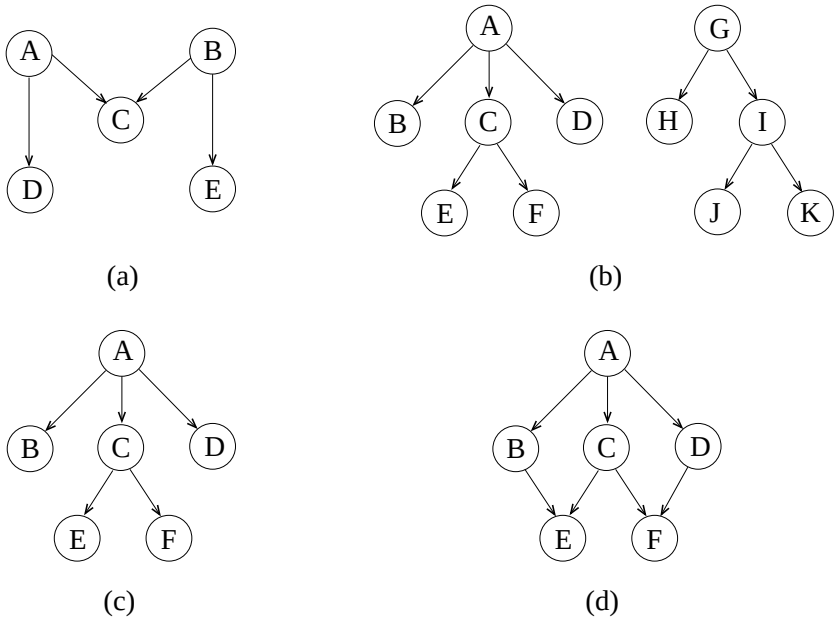


Figura 3.6: El DAG en (a) está simplemente conectado, el de (b) es un bosque, el de (c) es un árbol y el de (d) está múltiplemente conectado.

Ampliamos ahora el ejemplo añadiendo otro nodo mas, que representará en este caso a la variable *Boina*. En la Figura 3.7 se observa la estructura de red que utilizaremos para continuar con el ejemplo.

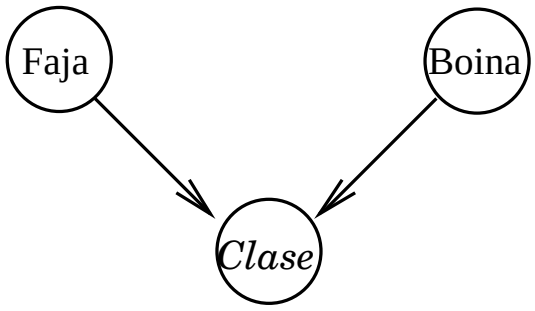


Figura 3.7: Red Bayesiana compuesta por tres nodos.

La probabilidad a priori de la variable *Boina*, estimada una vez mas de la base de datos, viene dada por:

$$P(Boina = No) = 0.50$$
$$P(Boina = Si) = 0.50.$$

En este caso, la variable *Clase* pasa a tener dos madres, con lo que ambas influirán en los valores de probabilidad de la clase. En la Tabla 3.1 se pueden apreciar los valores estimados de las probabilidades condicionadas de la variable *Clase*, para su primer valor (*Andaluza*) respecto a cada combinación de los valores de sus dos madres. Una tabla similar se establecerá para cada uno de los valores posibles de la variable hija. Hay que tener en cuenta que si una variable tiene muchas madres el tamaño de las tablas asociadas a la misma puede ser muy grande, con lo que el cálculo de probabilidades asociado se complica en gran manera, así como la memoria necesaria para almacenar toda la información.

Aplicando el teorema de Bayes, la probabilidad de que una persona dada, de la que no tengamos ninguna información, pertenezca a la clase Andaluza, viene dada por la siguiente fórmula:

$$P(Clase = Andaluza) = \sum_{f,b} P(Andaluza \mid Faja = f, Boina = b) \cdot P(Faja = f, Boina = b).$$

Al no existir un arco entre los dos nodos padres, estamos considerando implícitamente que ambos son de algún modo independientes. A este modo de independencia reflejado por la estructura

Tabla 3.1: Probabilidades condicionadas de un valor de la variable hija respecto a los valores de sus madres.

Faja | Boina → ↓	No	Si
No	0.50	1
Si	0.50	0

gráfica de la red se le denomina *independencia condicional* de un nodo sobre el otro dado el valor de su hijo común. Más tarde se formalizará este concepto, pero de momento baste saber que ello implica que $P(Faja|Boina) = P(Faja)$ y viceversa, o bien $P(Faja, Boina) = P(Faja) \cdot P(Boina)$, con lo que la fórmula anterior se puede enunciar de un modo más sencillo:

$$P(Clase = Andaluza) = \sum_f \sum_b P(Andaluza \mid Faja = f, Boina = b) \cdot P(Faja = f) \cdot P(Boina = b).$$

Se ha presentado aquí una propiedad fundamental de las RRBB: Tanto la presencia de un arco entre dos nodos, como la ausencia del mismo, aporta información respecto a la relación que se asume entre las variables subyacentes.

Supongamos ahora que sabemos que una persona determinada utiliza boina. Veamos como cambia la probabilidad de que su lengua materna sea la andaluza.

$$P(Clase = Andaluza \mid Boina = Si) = \frac{P(Clase = Andaluza, Boina = Si)}{P(Boina = Si)} = \frac{\frac{1}{16}}{\frac{8}{16}} = \frac{1}{8}.$$

Como podemos observar, esta es la estimación que se obtiene en la base de datos, ya que de los 8 casos en los que la variable *Boina* adopta su valor *Si*, en tan sólo en uno de ellos la variable *Clase* toma su valor *Andaluza*.

Consideremos un último ejemplo en el que van a intervenir todas las variables del modelo. Supongamos que la estructura de la RB es la que muestra la Figura 3.8. En esta ocasión, la variable asociada a la clase tiene dos madres y dos hijas.

Separación en grafos dirigidos

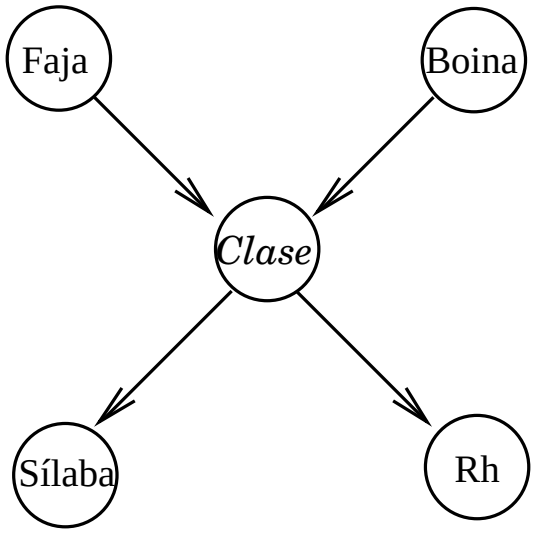


Figura 3.8: Red Bayesiana compuesta por cinco nodos.

Formalizamos en este punto un aspecto fundamental en las RRBB, cual es el concepto de separación entre los nodos del mismo. Mediante el criterio de separación conocido como $D\text{-}separation$ (Pearl, 1988), se puede indicar si un grafo verifica o no una relación de independencia dada. Presentamos formalmente el concepto, junto con las definiciones previas necesarias para ello.

Definición 3.2.1 *Dado un grafo dirigido y un camino no dirigido* $(... - U - A - V - ...)$*, el nodo A se denomina un* **nodo de aristas convergentes en un camino** *si las dos aristas del camino convergen a este nodo en el grafo dirigido, es decir, si el grafo dirigido contiene las aristas* $U \rightarrow A$ *y* $V \rightarrow A$*.*

Por ejemplo, en la red Bayesiana mostrada en la Figura 3.8, el nodo $Clase$ es un nodo de aristas convergentes en el camino $Faja$ - $Clase$ - $Boina$. A este tipo de nodos se le suele denominar también vértice en uve o nodo cabeza-cabeza.

Definición 3.2.2 D-separación. *Sean* X*,* Y *y* Z *tres subconjuntos disjuntos de nodos en un grafo acíclico dirigido* D*; entonces, se dice que* Z *D-separa* X *e* Y *si y sólo si a lo largo de todo camino no dirigido entre cualquier nodo de* X *y cualquier nodo de* Y *existe un nodo intermedio* A *tal que, o bien*

1. *A es un nodo de aristas convergentes en el camino y ni A ni sus descendientes están en Z, o bien*
2. *A no es un nodo de aristas convergentes en el camino y A está en Z.*

Cuando Z D-separa X e Y en D, se escribe $I(X, Y \mid Z)_D$ para indicar que la relación de independencia viene dada por el grafo D; en caso contrario, se escribe $D(X, Y \mid Z)_D$, para indicar que X e Y son condicionalmente dependientes dado Z en el grafo D.

En el ejemplo mostrado en la Figura 3.8, y a raíz de esta definición, se pueden deducir, entre otras, las siguientes dependencias e independencias condicionales entre conjuntos de nodos:

- $I(Faja,\ Rh \mid Clase)$
- $I(Sílaba,\ Rh \mid Clase,\ Faja)$
- $I(Faja,\ Boina \mid \emptyset)$
- $I(\{Boina,\ Faja\},\ Sílaba \mid Clase)$
- $D(Faja,\ Boina \mid Clase)$

Continuando con la presentación intuitiva de las RRBB, a partir de la red Bayesiana mostrada en la Figura 3.8, de la base de datos podemos estimar las probabilidades condicionadas necesarias. En este caso, hay que establecer las probabilidades condicionadas de la variable *Rh* respecto a la variable *Clase* –ver Tabla 3.2–, y de la variable *SílabasApellido* respecto a la variable *Clase* –ver Tabla 3.3–.

$\mathrm{P}(Rh = Positivo \mid Clase{=}Andaluza) = \frac{2}{3}$	$\mathrm{P}(Rh = Negativo \mid Clase{=}Andaluza) = \frac{1}{3}$
$\mathrm{P}(Rh = Positivo \mid Clase{=}Catalana) = \frac{2}{5}$	$\mathrm{P}(Rh = Negativo \mid Clase{=}Catalana) = \frac{3}{5}$
$\mathrm{P}(Rh = Positivo \mid Clase{=}Gallega) = \frac{2}{3}$	$\mathrm{P}(Rh = Negativo \mid Clase{=}Gallega) = \frac{1}{3}$
$\mathrm{P}(Rh = Positivo \mid Clase{=}Euskera) = \frac{2}{5}$	$\mathrm{P}(Rh = Negativo \mid Clase{=}Euskera) = \frac{3}{5}$

Tabla 3.2: Probabilidades condicionadas para el nodo *Rh*.

Hay que tener en cuenta el hecho de que esta es una variable especial en la que el número de valores no está completamente definido, ya que puede haber apellidos de más de 10 sílabas (Agirregomezkortaurrutibeaskoetxea) o de valores que no aparecen en la base de datos, como por

P(*SílabasApellido* = 2|*Clase*=*Andaluza*) = 0	P(*SílabasApellido* = 3|*Clase*=*Andaluza*) = $\frac{2}{3}$
P(*SílabasApellido* = 4|*Clase*=*Andaluza*) = $\frac{1}{3}$	P(*SílabasApellido* = 5|*Clase*=*Andaluza*) = 0
P(*SílabasApellido* = 6|*Clase*=*Andaluza*) = 0	P(*SílabasApellido* = 7|*Clase*=*Andaluza*) = 0
P(*SílabasApellido* = 10|*Clase*=*Andaluza*) = 0	
P(*SílabasApellido* = 2|*Clase*=*Catalana*) = $\frac{1}{5}$	P(*SílabasApellido* = 3|*Clase*=*Catalana*) = $\frac{3}{5}$
P(*SílabasApellido* = 4|*Clase*=*Catalana*) = $\frac{1}{5}$	P(*SílabasApellido* = 5|*Clase*=*Catalana*) = 0
P(*SílabasApellido* = 6|*Clase*=*Catalana*) = 0	P(*SílabasApellido* = 7|*Clase*=*Catalana*) = 0
P(*SílabasApellido* = 10|*Clase*=*Catalana*) = 0	
P(*SílabasApellido* = 2|*Clase*=*Gallega*) = $\frac{1}{3}$	P(*SílabasApellido* = 3|*Clase*=*Gallega*) = $\frac{1}{3}$
P(*SílabasApellido* = 4|*Clase*=*Gallega*) = $\frac{1}{3}$	P(*SílabasApellido* = 5|*Clase*=*Gallega*) = 0
P(*SílabasApellido* = 6|*Clase*=*Gallega*) = 0	P(*SílabasApellido* = 7|*Clase*=*Gallega*) = 0
P(*SílabasApellido* = 10|*Clase*=*Gallega*) = 0	
P(*SílabasApellido* = 2|*Clase*=*Euskera*) = 0	P(*SílabasApellido* = 3|*Clase*=*Euskera*) = $\frac{1}{5}$
P(*SílabasApellido* = 4|*Clase*=*Euskera*) = 0	P(*SílabasApellido* = 5|*Clase*=*Euskera*) = $\frac{1}{5}$
P(*SílabasApellido* = 6|*Clase*=*Euskera*) = $\frac{1}{5}$	P(*SílabasApellido* = 7|*Clase*=*Euskera*) = $\frac{1}{5}$
P(*SílabasApellido* = 10|*Clase*=*Euskera*) = $\frac{1}{5}$	

Tabla 3.3: Probabilidades condicionadas para el nodo *SílabasApellido*.

ejemplo de 1 sílaba (Paz). Por el momento no nos ocuparemos de ellos, pero será necesario tener en mente este tipo de cuestiones a la hora de establecer los valores de probabilidad asociados a los nodos de las RRBB.

Supongamos ahora que sabemos todas las características de un caso de estudio, y queremos calcular cuales son las probabilidades a posteriori de cada uno de los valores de la clase. Sea un caso con los siguientes valores en las variables: {*Faja* = *No*, *SílabasApellido* = 3, *Boina* = *No*, *Rh* = *Negativo* }. ¿Cuales serán ahora las probabilidades a posteriori de cada uno de los valores de la clase?

El teorema de Bayes nos indica que, para un valor concreto de la variable *Clase*, por ejemplo para el valor *Catalana*, el cálculo de la probabilidad asociada se realiza mediante la siguiente fórmula, en la que utilizamos, para abreviar, $P(Catalana)$ en lugar de $P(Clase = Catalana)$:

$$P(Catalana \mid Faja = No, SílabasApellido = 3, Boina = No, Rh = Negativo) =$$

$$= \frac{P(Catalana) \cdot P(Faja = No, SílabasApellido = 3, Boina = No, Rh = Negativo \mid Catalana)}{P(Faja = No, SílabasApellido = 3, Boina = No, Rh = Negativo)}$$

Nuevamente necesitamos realizar asunciones para utilizar la probabilidad conjunta que aparece. Hasta el momento se ha utilizado una hipótesis:

1. Independencia a priori de los nodos que no tienen antepasados comunes.

Enunciamos a continuación dos hipótesis más:

2 Independencia condicional de dos nodos con un padre común, dado que se conoce el valor de la variable asociada al nodo padre.

$$P(SílabasApellido, Clase \mid Rh) = P(SílabasApellido, Clase)$$

esto es,

$$P(SílabasApellido \mid Clase, Rh) = P(SílabasApellido \mid Clase)$$

3 La independencia condicional (para cada valor de la variable *Clase*) entre las variables madres e hijas de la misma:

$$P(Boina, Faja \mid Clase, SílabasApellido, Rh) = P(Boina, Faja \mid Clase)$$

De este modo, tenemos que:

$$P(Boina, Faja, SílabasApellido, Rh|Clase) = P(Boina, Faja \mid Clase) \cdot P(SílabasApellido) \cdot P(Rh)$$

con lo que el cálculo de las probabilidades a priori de los valores de la variable *Clase* se realiza de manera sencilla. En el ejemplo presentado, en el que la evidencia viene dada por el valor de las otras cuatro variables, $\{Faja = No,\ SílabasApellido = 3,\ Boina = No,\ Rh = Negativo\ \}$, tenemos que:

$$P(Clase = Gallega \mid Faja = No, SílabasApellido = 3, Boina = No, Rh = Negativo) =$$

$$= \frac{P(Cl = Gallega) \cdot P(Fa = No, Sílabas = 3, Boi = No, Rh = Neg \mid Cl = Gallega)}{P(Fa = No, Sílabas = 3, Boi = No, Rh = Neg)} =$$

$$= \frac{P(Cl = Gallega) \cdot P(Fa = No, Boi = No \mid Cl = Gallega) \cdot P(Sílabas = 3) \cdot P(Rh = Neg)}{P(Fa = No, Sílabas = 3, Boi = No, Rh = Neg)} =$$

$$= \frac{\frac{3}{16} \cdot \frac{1}{3} \cdot \frac{7}{16} \cdot \frac{8}{16}}{\frac{1}{16}} = 0.65625.$$

En donde se han utilizado las siguientes abreviaturas:
$Cl = Clase$, $Boi = Boina$, $Sílabas = SílabasApellido$, $Fa = Faja$, $Neg = Negativo$.

Estimación de los parámetros de una red Bayesiana

Las RRBB tienen dos partes fundamentales: la estructura y los valores de probabilidad asociados a los nodos. Durante la presentación intuitiva se han estimado los valores de probabilidad de la base de datos utilizando la frecuencia relativa, esto es, el estimador máximo verosímil de las probabilidades. A pesar de que esta estimación aparenta ser una buena aproximación, hay que tener en cuenta que, para problemas reales, suele suceder que los casos de la base de datos no abarcan todas las posibilidades de combinaciones entre valores de variables, con lo que este tipo de estimación puede llevar a una estimación de parámetros con abundancia de ceros, o lo que es peor, a situaciones en las que la estimación de la frecuencia indica que hay cero casos de cero posibles. Otro problema de este tipo de estimación es el sobreajuste, ya que la tendencia a estimar probabilidades nulas acarrea por otro lado una tendencia a estimar unos, lo cual puede hacer que la red Bayesiana no se comporte de forma adecuada para datos no pertenecientes a la base de datos. Existen varias aproximaciones que intentan solucionar estos problemas. Una de ellas, denominada *ley de la sucesión de Laplace* (Good, 1965) utiliza lo siguiente: en lugar de estimar la probabilidad directamente como $\frac{Casos favorables}{Casos totales}$, se utiliza como estimación inicial el número que se obtiene al dividir $\frac{N_{ijk}+1}{N_{ij}+r_i}$, donde N_{ijk} es el número de casos favorables[4], N_{ij} es el número de casos posibles[5], y r_i es el número de valores posibles de la variable i-ésima.

Para especificar la distribución de probabilidad de una red Bayesiana, se debe proporcionar la distribución de probabilidad a priori de todos los vértices raíz (vértices sin predecesores), así como las probabilidades condicionadas de todos los vértices no raíz, para cada posible combinación de sus padres o predecesores directos. Estos números, en conjunción con el DAG especifican totalmente la red Bayesiana. La probabilidad conjunta de cualquier punto n dimensional $(x_1, ..., x_n)$ correspondiente a una realización de la variable aleatoria n-dimensional $(X_1, ..., X_n)$ puede calcularse como:

$$P(x_1, ..., x_n) = \prod_{i=1}^{n} P(x_i|pa(x_i))$$

donde x_i representa el valor de la variable X_i, y $pa(x_i)$ representa el valor de los padres de X_i.

[4] N_{ijk} representa el número de veces que la variable i toma el valor k de entre aquellos casos en los que sus variables madres toman su j-ésima combinación.

[5] N_{ij} representa el número de veces que las variables madres de la i-ésima variable toman su j-ésima combinación.

3.3 Definición formal de red Bayesiana

Una definición de red Bayesiana es la siguiente (Neapolitan, 1990):

Definición 3.3.1 *Sea n un conjunto finito de variables discretas definidas en el mismo espacio probabilístico, sea P su distribución de probabilidad conjunta, y sea $G = (V, A)$ un DAG. Para cada $X \epsilon V$, sea $pa(X) \subseteq V$ el conjunto de todos los padres de X, sea $d(X) \subseteq V$ el conjunto de todos los descendientes de X. Además para cada $X \epsilon V$, se define $a(X) \subseteq V$ como $a(X) = V \setminus (d(X) \cup X)$, es decir, el conjunto de variables en V, excluyendo X y sus descendientes. Si para cada subconjunto $W \subseteq a(X)$, W y X son condicionalmente independientes dado $pa(X)$, es decir si se cumple alguna de las dos condiciones siguientes:*

$$(i) P(X \mid pa\ (X)) = 0 \ \acute{o}\ P(W \mid pa\ (X)) = 0$$
$$(ii) P(X \mid W \& \ pa\ (X)) = P(X \mid pa\ (X))$$

entonces diremos que $C = (V, A, P)$ es una red Bayesiana.

Según la definición anterior, dado un grafo acíclico dirigido, y una distribución de probabilidad sobre sus variables, se dice que hay *separación direccional* si, dado un vértice X cualquiera, el conjunto de sus padres, $pa(X)$, separa condicionalmente este vértice de cualquier otra variable Y en que no haya descendientes de X. Es decir, si se verifica que:

$$P(x|pa(x), y) = P(x|pa(x)).$$

Podemos por tanto definir una red Bayesiana como un grafo acíclico dirigido más una distribución de probabilidad sobre sus variables, que cumple la propiedad de separación direccional. En la Figura 3.9 se presenta un ejemplo de una red Bayesiana múltiplemente conectada.

3.4 Propagación de la evidencia en redes Bayesianas

3.4.1 Introducción

La propagación de la evidencia o inferencia probabilista en redes Bayesianas –*Probabilistic Inference in Bayesian NETworks (PIBNET)*–, consiste en realizar los cálculos necesarios para obtener la probabilidad a posteriori de una o varias variables dados los valores asignados a otras variables en la red Bayesiana, esto es, el cálculo de $P(S_1|S_2)$, donde tanto S_1 como S_2 designan bien una única variable, o bien una conjunción de variables. Los valores de las variables que forman parte de S_2 constituyen la *evidencia*.

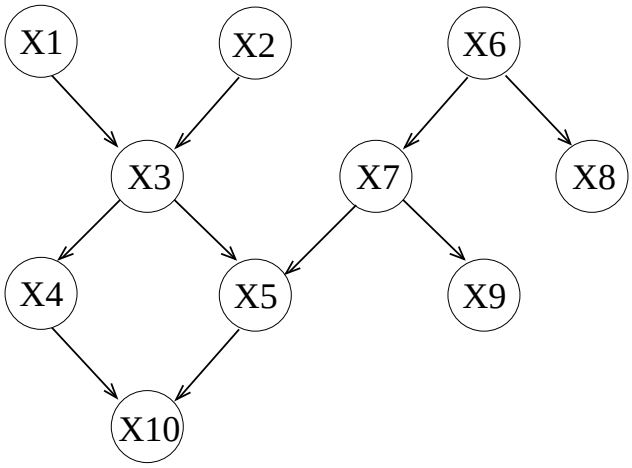


Figura 3.9: Red Bayesiana múltiplemente conectada.

Cooper (1990) demostró que la computación de la probabilidad a posteriori es un problema NP-duro, incluso para una única variable, y como consecuencia que las otras formas de inferencia probabilística, son también problemas NP-duros.

La inferencia probabilística en ciertos tipos restringidos de redes Bayesianas puede efectuarse de forma eficiente. Así, en redes simplemente conectadas el *PIBNET* se resuelve en tiempo lineal en función del tamaño de la red. Sin embargo en redes múltiplemente conectadas es un problema más difícil desde el punto de vista computacional. Todos los algoritmos conocidos, tienen un tiempo que en el peor de los casos es exponencial respecto al número de variables no instanciadas.

Al tratarse de un problema NP-duro, se han tratado de encontrar diversas alternativas para realizar la propagación:

1. *Algoritmos de aproximación*, los cuales producen soluciones inexactas, pero que garantizan que la verdadera solución se encuentra dentro de cierto entorno de la solución obtenida. Dentro de esta categoría de algoritmos pueden englobarse los denominados algoritmos de propagación basados en la simulación. Dagum y Luby (1993) han demostrado que el tratamiento del problema *PIBNET* por medio de algoritmos de aproximación también pertenece a la categoría de problemas NP-duros.

2. *Algoritmos para casos especiales* capaces de resolver el *PIBNET* en forma eficiente para tipos especiales de redes Bayesianas. Un ejemplo de ello puede ser el anteriormente mencionado para redes simplemente conectadas.

3.4.2 Métodos exactos

La complejidad de los algoritmos exactos es muy sensible respecto a la estructura de la red Bayesiana. Así por ejemplo, Kim y Pearl (1983) proporcionan un algoritmo exacto de propagación en poliárboles con una complejidad lineal respecto al número de variables. Sin embargo, todos los algoritmos exactos conocidos para redes Bayesianas múltiplemente conectadas, presentan una complejidad que en el peor de los casos es exponencial con el número de variables no instanciadas.

Junto con la aproximación *HUGIN*, basada en las ideas de Lauritzen y Spiegelhalter (1988), y mejorada por los trabajos de Jensen y col. (1990a, 1990b), el algoritmo exacto más difundido se debe a Pearl (1986). Una variante de éste último algoritmo se encuentra en Díez (1992).

En todos los experimentos realizados con redes Bayesianas se ha utilizado el software *HUGIN*, y, por ende, la propagación *HUGIN*. Profundizar en el modo en que se realiza ésta excede el ánimo de esta memoria; sólo indicar que se efectúa una transformación de la red Bayesiana en lo que se ha dado en llamar 'árbol de juntura' o 'árbol de cliques', en el que se realiza una partición no disjunta de los nodos de la red, y luego se produce una serie de pasos de mensajes indicando los cambios que las evidencias introducidas producen en los valores de probabilidad reflejadas en los diferentes nodos.

3.4.3 Métodos basados en la simulación

La simulación estocástica de redes Bayesianas es una técnica que se utiliza en redes Bayesianas como alternativa a los métodos exactos de propagación, debido fundamentalmente a su flexibilidad, facilidad de implementación, y al hecho de que la complejidad computacional asociada permanece manejable a medida que aumenta el tamaño y la complejidad estructural de la red.

En este tipo de aproximación la dificultad radica en decidir en que momento se ha logrado el grado de precisión deseado. También hay que destacar la lentitud de la técnica.

El procedimiento de simulación de redes Bayesianas más extendido es el muestreo lógico probabilístico, introducido por Henrion (1988).

3.5 Aprendizaje estructural de redes Bayesianas

Existen varias maneras de establecer la estructura de una RB:

1. El experto en el dominio que se está modelando mediante una red Bayesiana decide cual es la estructura de la misma, basándose en su conocimiento sobre las relaciones entre las variables.

2. Se obtiene la estructura de un modo automático mediante un algoritmo de aprendizaje.

   - A partir de una base de datos.
   - A partir de una lista de dependencias/independencias condicionales.
   - Utilizando tanto la lista de dependencias/independencias como la base de datos.

3. Mediante una técnica mixta que recoja tanto las opiniones del experto como los descubrimientos obtenidos mediante la aplicación de un algoritmo de aprendizaje.

Durante la última década, se han desarrollado un gran número de algoritmos cuyo propósito es el de inducir la estructura de la red Bayesiana que mejor se adapte a los datos, esto es, que represente las dependencias e independencias condicionales que aparecen reflejadas en los datos contenidos en una base de datos. La necesidad de estos algoritmos viene determinada por el hecho de que es muy difícil que un experto pueda establecer de forma precisa la estructura y todas las probabilidades necesarias para rellenar las tablas asociadas a los nodos de una RB.

Se presentan a continuación una serie de aproximaciones para el aprendizaje estructural de RRBB, relacionadas todas ellas con redes de estructura general, o múltiplemente conectadas. Existen otros algoritmos que asumen cierta simplificación en la estructura final de la red –árbol, poliárbol,...– para reducir de este modo el tamaño del espacio de búsqueda.

Las diferentes aproximaciones se dividen teniendo en cuenta si necesitan establecer un orden entre las variables –otro tipo de simplificación– o no. En Heckeman y col. (1995) se presenta una revisión más completa.

El hecho de asumir un orden entre las variables tiene el siguiente significado: dadas las n variables del modelo $X_1, X_2, ..., X_n$ que se suponen ordenadas, X_i puede tener como madre a la variable X_j sólo si $j < i$, esto es, si en el orden que se ha establecido la variable X_j precede a la variable X_i. Al imponer esta restricción, se reduce el espacio de búsqueda, ya que el espacio de estructuras de redes Bayesianas que la cumplen tiene una cardinalidad de $2^{\binom{n}{2}}$. Algunos de los métodos desarrollados bajo esta idea pueden verse en Cooper y Herskovits (1990) y en Bouckaert (1994).

Si no se asume la existencia de un orden entre variables, el espacio de búsqueda tiene un mayor tamaño, y su cardinalidad viene dada por la fórmula de Robinson (1977):

$$f(n) = \sum_{i=1}^{n} (-1)^{i+1} \binom{n}{i} 2^{i(n-i)} f(n-i); f(0) = 1; f(1) = 1.$$

Hay numerosos autores investigando en algoritmos de tipo general, sin asunciones simplificadoras. Entre ellos se encuentran Bouckaert (1992), Provan y Singh (1995) y Larrañaga y col. (1996b).

Es necesario un criterio para establecer cual de entre varias redes Bayesianas es la mejor. Los diferentes algoritmos que se presentan a continuación utilizan diversas medidas para establecer la calidad de las RRBB. A estas medidas se les denomina habitualmente *Métricas*, y están basadas en medidas estadísticas.

El algoritmo K2

El algoritmo K2 (Cooper y Herskovits, 1992) asume que se ha establecido un orden entre las variables. Utiliza una métrica Bayesiana, establecida por los mismos autores, para evaluar la red Bayesiana que se va construyendo.

El algoritmo K2, dada una base de datos D, busca la estructura de RB B_{S^*} con $P(B_S, D)$ maximal, siendo $P(B_S, D)$ la probabilidad a posteriori de la estructura dados los datos, tal y como se describe en el siguiente teorema, enunciado por Cooper y Herskovits (1992).

Teorema 1 *Sea Z un conjunto de n variables discretas, donde una variable X_i de Z tiene r_i posibles asignaciones de valores: $(v_{i1}, \ldots, v_{ir_i})$. Sea D una base de datos que contiene N casos, donde en cada caso aparece una asignación de valor para cada variable de Z. Supongamos que B_S denota una estructura de red Bayesiana en la que intervienen como variables únicamente las del conjunto Z. Cada variable X_i de B_S tiene un conjunto de madres, que se representan mediante una lista de variables π_i. Denotamos mediante w_{ij} la j-ésima única instanciación de π_i relativa a D. Supongamos que existen q_i instanciaciones únicas de π_i. Definimos N_{ijk} como el número de casos en D en los cuales la variable X_i toma su valor v_{ik} y además π_i se instancia como w_{ij}. Sea $N_{ij} = \sum_{k=1}^{r_i} N_{ijk}$. Si se da un modelo de red Bayesiana, estos casos suceden de modo independiente, no hay casos con valores perdidos en las variables y la función de densidad $f(B_P|B_S)$ es uniforme, entonces se deduce que*
$P(B_S|D) = P(B_S) \prod_{i=1}^{n} g(i, \pi_i)$, *donde* $g(i, \pi_i) = \prod_{j=1}^{q_i} \frac{(r_i-1)!}{(N_{ij}+r_i-1)!} \prod_{k=1}^{r_i} N_{ijk}!$

Este algoritmo asume que todas las estructuras de RRBB son equiprobables a priori. El algoritmo busca, para cada nodo, el conjunto de padres que maximiza $g(i, \pi_i)$ –métrica de Cooper y Herskovits (1992)–. K2 es un heurístico greedy. Comienza asumiendo que un nodo no tiene padres, añadiendo en cada paso al conjunto de padres aquel nodo de entre los anteriores al que se está tratando –según el orden total dado–, que incrementa en mayor medida la probabilidad de la estructura de red resultante. Obviamente, este algoritmo no garantiza que la estructura que se obtiene mediante su utilización sea la más probable dados los datos.

Algoritmo K2

```
INPUT: Un conjunto de n nodos, un orden entre los nodos,
una cota superior u del numero de padres que puede tener un nodo,
y una base de datos D conteniendo N casos.
OUTPUT: Para cada nodo, el conjunto de sus nodos padres.
BEGIN K2
   FOR i := 1 TO n DO
        BEGIN
             π_i := ∅;
             P_old := g(i, π_i):
             OKtoProceed := TRUE
             WHILE OKtoProceed AND |π_i| < u DO
                  BEGIN
                       Sea z el nodo en Pred(x_i) \ π_i que maximiza g(i, π_i ∪ {z});
                       P_new := g(i, π_i ∪ {z});
                       IF P_new > P_old THEN
                                 BEGIN
                                      P_old := P_new;
                                      π_i := π_i ∪ {z}
                                 END
                       ELSE OKtoProceed := FALSE;
                  END;
             WRITE('Nodo:', X_i, 'Padres de este nodo:', π_i)
        END;
END K2.
```

Figura 3.10: El algoritmo *K2*.

La métrica BIC

Un método para evaluar la bondad de una RB respecto a los datos, es el denominado *Bayesian information criterion* (BIC) (Schwarz 1978). Mediante el mismo, para evaluar una estructura de red Bayesiana, S, con respecto a una base de datos D que contiene N casos, se utiliza la métrica BIC, denotada mediante $BIC(S, D)$ y que es calculada de la siguiente manera:

$$BIC(S,D) = \sum_{i=1}^{n}\sum_{j=1}^{q_i}\sum_{k=1}^{r_i} N_{ijk}\log\frac{N_{ijk}}{N_{ij}} - \frac{\log N}{2}\sum_{i=1}^{n}(r_i-1)q_i$$

donde N_{ijk}, N_{ij} y q_i son los previamente definidos.

La parte cuantitativa de la red Bayesiana, esto es, las distribuciones de probabilidad θ_{ijk} asociadas a los nodos, son calculadas utilizando los valores esperados, tal y como se muestra en Cooper y Herskovits (1992):

$$E[\theta_{ijk} \mid S, D] = \frac{N_{ijk}+1}{N_{ij}+r_i}.$$

Mediante la utilización de esta métrica se evalúan las estructuras de red Bayesiana dados los datos. Obviamente, para ello necesitamos tener la estructura, con lo que el problema de encontrar la estructura que mejor se adapte a los datos permanece. Encontrar la mejor estructura buscándola en el espacio de todas las posibles es un problema NP-duro (Chickering y col. 1994). Se han utilizado con éxito técnicas de búsqueda global para encontrar la estructura de la RB (Bouckaert, 1993, Larrañaga y col. 1996a, 1996b, 1996c, Etxeberria y col. 1997, Myers y col., 1999, Wong y col., 1999), aunque el costo computacional que exigen hace que se tienda a utilizar métodos más sencillos, aunque no tan ambiciosos, para lograr una estructura aceptable.

Un algoritmo interesante para el aprendizaje estructural de redes Bayesianas dado un conjunto de datos es el algoritmo B (Buntine 1991). Este es un algoritmo greedy que comienza con una estructura de red totalmente desconexa, esto es, sin arcos, y añade en cada paso el arco que en mayor medida incremente el valor de la métrica que se esté utilizando (por ejemplo, la métrica BIC). El algoritmo finaliza cuando se llega a un punto en el cual no hay ningún arco que al ser incluido en la estructura incremente el valor de la métrica utilizada.

La aproximación Minimum Description Length

En esta aproximación (Rissanen, 1987) se utiliza la idea de que la representación que mejor se adapta a los datos es aquella que minimiza la suma de las longitudes de codificación del modelo y de los datos dado el modelo. De este modo, se trata de buscar un equilibrio entre la complejidad

de la red Bayesiana utilizada para representar el modelo, y la precisión con que éste representa las relaciones entre las variables. Un modelo de red muy complejo puede resultar muy exacto respecto al comportamiento de los datos, pero la longitud necesaria para su codificación también será elevada.

En el caso de las RRBB, los modelos complejos se asocian con redes densamente conectadas, siendo el objetivo hallar redes menos densas que representen de forma aceptable el comportamiento de los datos.

Se necesita algún método para codificar la estructura gráfica y las distribuciones de probabilidad, teniendo en cuenta que la longitud de las codificaciones elegidas aumentarán en ambos casos con la densidad de la red. Para codificar los datos dado el modelo, se utiliza un tipo de codificación que crece de forma inversamente proporcional a la densidad.

Existen diferentes algoritmos de aprendizaje estructural que emplean el principio MDL. Algunos de ellos son los presentados por Bouckaert (1993), Suzuki (1993), Lam y Bacchus (1993) y Friedman y Goldszmidt (1996).

El algoritmo BENEDICT

Este algoritmo se encuentra entre los denominados métodos híbridos, en los que se utiliza una técnica de búsqueda guiada por una métrica, pero a la vez se utilizan test de dependencia o independencia condicional. Se trata de cuantificar la *discrepancia* entre cualquier red candidata y la base de datos, midiendo para ello las desavenencias entre las independencias condicionales representadas en la red, con las correspondientes independencias condicionales que puedan deducirse de la base de datos. El nombre BENEDICT (Acid y de Campos, 1996) se obtiene de las letras que se han puesto en mayúsculas en los vocablos ingleses BElief NEtworks DIscovery using Cut-set Techniques.

BENEDICT utiliza un proceso greedy de búsqueda: partiendo de un grafo totalmente inconexo, esto es, en el que se encuentran los n nodos correspondientes a las n variables del problema, sin que exista ningún arco entre ellos, el algoritmo trata de insertar cada uno de los arcos posibles, incluyendo de forma permanente en la estructura gráfica de la RB aquel que produce una mayor reducción en la discrepancia. Este proceso se repite, partiendo ahora de la estructura de RB obtenida al trazar el arco, hasta que se satisface una condición de finalización.

La versión original del algoritmo asume que se ha establecido un orden total entre las variables del modelo, y utiliza la entropía cruzada de Kullback-Leibler (1968) para medir las discrepancias entre un subconjunto[6] de las independencias condicionales representadas.

[6] Dado que el número de independencias representadas en un grafo puede ser muy elevado, se utilizan en este algoritmo sólo las independencias del tipo $I(X_i, X_j|\Pi_G(X_i))$, donde se han tenido en cuenta que, en un grafo acíclico dirigido G todo nodo X_j que no sea descendiente del nodo X_i se encuentra d-separado del mismo por el conjunto de sus padres en el grafo, $\Pi_G(X_i)$. Se supone que $j > i$ en el orden total utilizado.

3.6 Ventajas e inconvenientes de las redes Bayesianas

Finalizamos esta introducción a las RRBB con una reseña a sus principales ventajas e inconvenientes.

Incluiremos entre las *ventajas* de la utilización de las Redes Bayesianas como formalismo de representación de la incertidumbre el hecho de que constituyen un *sistema normativo*, es decir, se encuentran fundamentadas en una teoría matemática. En esto se diferencian de los primeros *sistemas expertos*, como MYCIN (Shortliffe y col., 1975), que se implementaron de una forma mas artesanal o intuitiva.

Otra importante ventaja proviene del hecho de que la *representación* se encuentra *estructurada* de tal manera que las posibles modificaciones significan añadir o quitar enlaces entre variables, añadir o quitar variables, cambios en las tablas de probabilidad asociadas, todo ello sin que exista el peligro de caer en inconsistencias. Nos parece importante el resaltar la posibilidad de efectuar distintos tipos de *razonamiento*: *abductivo, predictivo o intercausal.*

Como *inconvenientes*, señalaremos el *alto costo computacional de los algoritmos de propagación* de la evidencia, sobre todo en el caso de redes Bayesianas de gran tamaño. Otro problema puede provenir de la *obtención del conocimiento*, tanto a nivel cualitativo como a nivel cuantitativo.

Capítulo 4

Aportaciones realizadas en el área de las redes Bayesianas

Se presentan en este capítulo las aportaciones que en esta tesis han realizado al área de las redes Bayesianas, en concreto en lo que al aprendizaje estructural de las mismas con objeto clasificatorio se refiere. Ver Larrañaga y col., (1996d, 1996e), Larrañaga y col., (1997a), Sierra y Larrañaga, (1998) y Dizdaverich y col. (1998) para aplicaciones a problemas reales.

4.1 Introducción

Los *sistemas expertos* son los programas de ordenador más populares de entre los que se enmarcan en el área de la mal llamada *inteligencia artificial*. Se utilizan con el ánimo de ayudar –a veces substituir– al ser humano en la realización de tareas que requieren de cierto conocimiento y experiencia. Aunque existen dominios de aplicación en los que las tareas pueden ser representadas mediante reglas lógicas, otros dominios se caracterizan por la incertidumbre inherente en los mismos.

Durante bastante tiempo, la teoría de la probabilidad no se tuvo en cuenta para realizar razonamiento bajo incertidumbre, necesario para construir sistemas expertos en estos dominios, debido fundamentalmente al alto costo computacional que se requería para trabajar con el gran número de probabilidades que se necesitaban. A finales de la década de 1980, Lauritzen y Spiegelhalter (1988) mostraron que estas dificultades pueden suavizarse si se aprovecha el carácter modular de los modelos gráficos asociados a los denominados *sistemas expertos probabilísticos*, que son implementados mediante redes Bayesianas.

Las RRBB constituyen un entorno para realizar razonamiento bajo incertidumbre. Como ya se ha visto, son grafos acíclicos dirigidos, en los que los nodos representan las variables del modelo,

y los arcos las dependencias e independencias que se verifican entre las mismas. Basadas en el concepto de independencia condicional, cada nodo tiene asociados unos valores de probabilidad, que son las probabilidades a priori en el caso de un nodo sin padres –o nodo raíz– o las probabilidades condicionadas de los valores del nodo respecto a los valores de sus nodos padres en otro caso. Estas probabilidades, en conjunción con el grafo, forman la red Bayesiana.

Una vez construida, la RB constituye un entorno eficiente para realizar inferencia probabilística, bien mediante métodos de propagación exactos, bien mediante métodos aproximados. En cualquier caso, continúa existiendo el problema de la obtención de la estructura de la RB. Nuestro objetivo es, en este punto, realizar de un modo automático el aprendizaje estructural de la red Bayesiana. En este capítulo se estudia el comportamiento de diversos modelos de RB aprendidos mediante AGs a la hora de ser considerados como clasificadores bajo incertidumbre.

En los diversos métodos que se utilizan habitualmente para realizar la así llamada clasificación supervisada, se realiza el aprendizaje de un modelo a partir de un fichero de datos, pero teniendo siempre en cuenta el hecho de que existe una variable especial, que es la que deseamos clasificar.

Así, en los métodos de construcción de árboles de clasificación, se realiza la expansión o la poda de un árbol en base a los resultados obtenidos al clasificar la variable especial (Mitchel, 1997). En la regresión logística, cuando la variable a clasificar es binaria, se tiene muy en cuenta la variable objetivo a la hora de estimar el valor de los parámetros (Hosmer y Lemeshow, 1989), al igual que en el análisis discriminante (Fisher, 1936). En la inducción de Reglas se mide la potencia de las condiciones a partir del beneficio obtenido a la hora de determinar la clase de pertenencia de los ejemplos que las satisfacen (Hand, 1997). En redes neuronales se realiza el aprendizaje en base a los resultados obtenidos (McCulloch y Pitts, 1943) e incluso en algunas versiones de K-NN (Dasarathy, 1991) se tiene en cuenta la variable especial para determinar cual va a ser la distancia a utilizar o la ponderación a aplicar a cada una de las variables predictoras.

En el aprendizaje estructural de RRBB no se acostumbra a tener en cuenta la existencia de una variable especial, aprendiéndose una distribución de probabilidad considerando a todas las variables por igual mediante algún tipo de métrica (Buntine, 1994).

Existe un gran número de problemas, que dentro del ámbito de la llamada Inteligencia Artificial se enmarcan en lo que se conoce como clasificación supervisada. Así, la mayoría de los sistemas expertos utilizados para el diagnóstico médico, los sistemas de reconocimiento de caracteres, los sistemas de concesión de créditos, etc., son problemas que se abordan habitualmente mediante una o varias de las técnicas antes mencionadas.

Se presentan aquí una serie de modelos utilizados para aplicar las RRBB en problemas de clasificación supervisada de un modo eficiente, teniendo en cuenta para ello la existencia de una

variable muy especial, que es la que deseamos clasificar mediante la estructura de RB obtenida en base a los datos.

Utilizamos los AAGG (Holland, 1975) para movernos de forma inteligente por el espacio de búsqueda, esto es, por el espacio formado por las estructuras de RB que se consideran en cada una de las diferentes aproximaciones.

Obviamente, el espacio total en el que nos movemos es el de todas las posibles estructuras válidas de RB para el caso considerado, con lo que el algoritmo más general realiza una búsqueda en todo el espacio. Habida cuenta que el número de estructuras posibles es muy elevado respecto al numero de variables –recordemos que el problema de aprendizaje estructural de RBs es NP-duro (Chickering y col., 1995)–, hemos considerado una serie de estructuras que limiten de alguna forma el tamaño del espacio de búsqueda para que el aprendizaje sea mas rápido, sin perder por ello fiabilidad en el método.

La estructura mas sencilla que hemos utilizado es la denominada Naive Bayes, en la que la variable a clasificar es madre del resto de variables (variables predictoras) no habiendo entre estas últimas ninguna relación directa. Este tipo de estructuras funciona sorprendentemente bien desde el punto de vista clasificatorio, en problemas de tipo médico, probablemente debido a que la toma de datos, esto es, la medición de síntomas de los pacientes se realiza de forma que cada uno de ellos añada información independiente a la de los demás, sin que se solapen las informaciones de la variables predictoras. A pesar de que el Naive Bayes supone considerar independientes entre sí a dichas variables dada la clase, los resultados son francamente buenos.

La primera extensión a este modelo, y la más intuitiva, consiste en no considerar independientes las variables predictoras, y permitir que exista algún tipo de dependencia entre las mismas, realizando la búsqueda en el espacio de aquellas estructuras en las que todas las variables predictoras son hijas de la variable a predecir, pero existe también algún tipo de relación entre estas últimas. A esta aproximación se le ha denominado Tree Augmented Network (TAN), y el método de aprendizaje mediante AAGG de esta estructura recibe el acrónimo TAN-GA.

Otra manera de aplicar este mismo concepto es la de agrupar varias variables en un solo nodo, utilizando las probabilidades conjuntas de las mismas para realizar los cálculos. Los nodos son hijos del correspondiente a la variable a clasificar (C), y no hay relaciones entre ellos.

Una nueva extensión consiste en hacer que todas las variables que participan en el sistema constituyan el llamado Markov Blanquet de la variable a clasificar, esto es, que todas ellas sean o bien madre de C, o hija de C o madre de alguna hija de C. En nuestra aproximación una variable puede tener varias de estas características a la vez, siempre que no se formen ciclos, claro está.

Existen, por otro lado, un gran número de algoritmos que realizan el aprendizaje estructural de la RB sin tener en cuenta las características especiales que conlleva el hecho de que vaya a ser

utilizada como clasificador. Asi, algoritmos de aprendizaje de una estructura general, (Bountine, 1994) realiza una búsqueda en la que el objetivo es hallar una distribución de probabilidad, que se supone es la que mejor se adapta a los datos.

Nosotros hemos utilizado la métrica de Cooper y Herskovits (1992) para realizar este tipo de aprendizaje estructural, utilizando los AAGG para encontrar la RB más probable a posteriori, según la idea del algoritmo K2. Comparamos los resultados obtenidos por estas redes generales con las obtenidas considerando la especificidad del problema a tratar.

Existen trabajos similares a este, como el realizado por Ezawa y Norton (1995), sobre fraudes a la compañía telefónica ATT, en el que realiza una aproximación simplificada de lo que sería el TAN-GA. Heckerman (1996) presenta las RB como posibles modelo para la realización de minería de Datos. Spiegelhalter y Dawid (1993) dan ideas sobre medidas reales a la hora de evaluar una RB. Friedman y Goldszmidt (1996) utilizan las ideas de Naive Bayes y Tree Augmented Network. Domingos y Pazzani (1997) aportan la extensión al Naive Bayes consistente en hallar la partición óptima de las variables predictoras, utilizando luego las distribuciones conjuntas en los nodos de un Naive Bayes. Acid (1999), en sus aproximaciones Benedicte, Inocencio I, Inocencio II y Zosimo, realiza también el aprendizaje teniendo en cuenta la variable especial.

Presentamos en la siguiente sección los diferentes modelos que se han tenido en consideración para la utilización de las RRBB como paradigma de clasificación supervisada, mostrando a continuación una metodología para la inducción automática de la estructura de una red Bayesiana a partir de una base de datos. Se trata de una aproximación basada en los *algoritmos genéticos*, que realiza una búsqueda en el espacio de posibles estructuras de redes Bayesianas con el objetivo de encontrar aquella que mejor comportamiento clasificatorio manifieste. Se presentan a continuación diversas aproximaciones en las que se han ido exigiendo diversas condiciones a las estructuras de la RB en aras a reducir la cardinalidad del espacio de búsqueda.

4.2 Redes Bayesianas en clasificación supervisada

4.2.1 Introducción

Sean: θ la clase; $\mathbf{x}$ vector de características de un caso dado; $P(\theta \mid \mathbf{x})$ probabilidad de que un objeto con características $\mathbf{x}$ pertenezca a la clase θ.

Se trata de encontrar la clase θ^* verificando: $P(\theta^* \mid \mathbf{x}) = \max_\theta P(\theta \mid \mathbf{x})$. Utilizando el teorema de Bayes, tenemos que:

$$P(\theta \mid \mathbf{x}) = \frac{P(\mathbf{x} \mid \theta)P(\theta)}{\sum_k P(\mathbf{x} \mid \theta_k)P(\theta_k)}$$

donde $P(\theta_k)$ denota la probabilidad a priori de cada clase.

Las aproximaciones que para el aprendizaje de redes Bayesianas con objetivos clasificatorios tienen en cuenta, en general, el hecho de que existe una variable especial en la red Bayesiana, cual es la correspondiente a la clase del problema. Sólo una de las aproximaciones que se van a presentar realiza una búsqueda de una estructura general sin otra restricción que la impuesta por la característica de aciclidad de la RB a aprender.

Los métodos de aprendizaje que se presentan a continuación se enmarcan dentro de los denominados *métrica + búsqueda*, es decir, mediante una función se valora cada una de las estructuras de RB candidata, y se procede a efectuar una búsqueda dentro del espacio de posibles estructuras.

4.2.2 Naive-Bayes

Una de las formas más simples que se pueden idear al considerar la estructura de una red Bayesiana con objetivos clasificatorios, es el denominado *Naive-Bayes* (Duda y Hart 1973). Su denominación proviene de la hipótesis ingenua sobre la que se construye, que consiste en considerar que todas las variables predictoras son condicionalmente independientes dada la variable a clasificar –véase la Figura 4.1–.

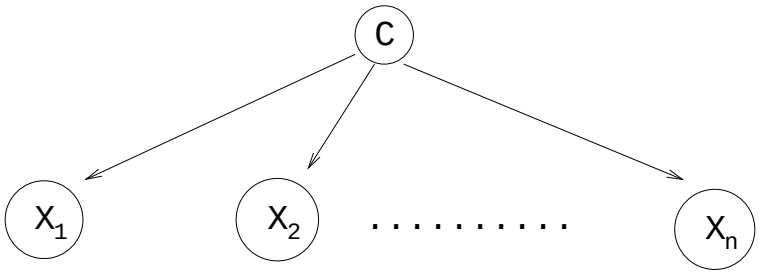


Figura 4.1: Naive-Bayes.

La probabilidad de que el j -ésimo ejemplo pertenezca a la clase i-ésima de la variable C, puede calcularse, sin más que aplicar el teorema de Bayes, de la siguiente manera:

$$P(C = \theta_i \mid X_1 = x_{1j}, ..., X_n = x_{nj}) \propto P(C = \theta_i) \cdot P(X_1 = x_{1j}, ..., X_n = x_{nj} \mid C = \theta_i).$$

En el caso de que las variables predictoras sean condicionalmente independientes dada la variable C, se obtiene que:

$$P(C = \theta_i \mid X_1 = x_{1j}, ..., X_n = x_{nj}) \propto P(C = \theta_i) \cdot \prod_{r=1}^{n} P(X_r = x_{rj} \mid C = \theta_i).$$

El modelo *Naive-Bayes* presenta un comportamiento muy dependiente del tipo de dominio. Así por ejemplo, en dominios médicos, donde el conocimiento sobre el problema es elevado y por tanto

tan sólo se recoge información relativa a variables que podríamos decir que se complementan, el *Naive-Bayes* proporciona resultados aceptables, mientras que en dominios poco estructurados, en los que las variables del sistema se encuentran altamente correlacionadas, el comportamiento del *Naive-Bayes* suele ser más bien pobre.

4.2.3 Estructura de Árbol Aumentado

Friedman y col. (1997) realizan una primera extensión del modelo anterior con las estructuras *TAN (Tree Augmented Network)*, que obtiene mejores resultados que los obtenidos por el *Naive-Bayes*, a la vez que mantiene la simplicidad computacional y la robustez del anterior.

Un modelo *TAN* es una red Bayesiana donde la variable a clasificar no tiene padres, es padre de todas las variables predictoras, y se permite además que existan relaciones madre-hija entre las variables predictoras, siempre que no se formen ciclos. Véase por ejemplo la Figura 4.2.

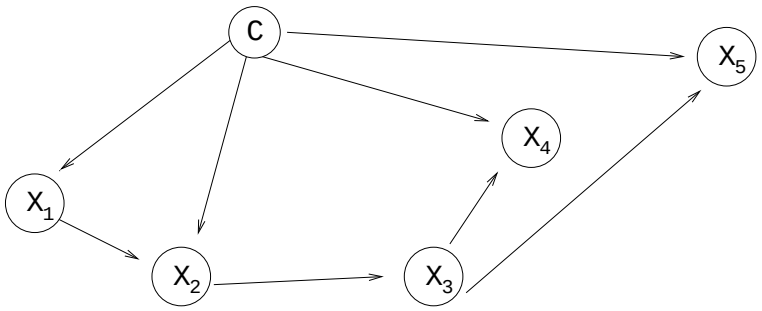


Figura 4.2: Estructura de Árbol Aumentado.

El algoritmo propuesto por Friedman y col. (1997) para el aprendizaje de estructuras TAN es como sigue:

1. Calcular $I_P(X_i, X_j \mid C)$ para cada par de variables predictoras, con $i \neq j$.
2. Construir un grafo no dirigido completo en el cual los vértices son las variables predictoras $X_1, ..., X_n$. Asignar a cada arista conectando las variables X_i y X_j un peso dado por $I_P(X_i, X_j \mid C)$.
3. Construir un árbol expandido de máximo peso.
4. Transformar el árbol resultante no dirigido en uno dirigido, escogiendo una variable raíz, y direccionando todas las aristas partiendo del nodo raíz.

5. Construir un modelo *TAN* añadiendo un nodo etiquetado como C, y posteriormente un arco desde C a cada variable predictora X_i.

4.2.4 Partición

Domingos y Pazzani (1996) presentan un modelo que puede considerarse que se posiciona en un lugar intermedio entre los modelos extremos, en los que, por una parte se tienen que calcular las $(r-1)2^n$ distribuciones de probabilidad –para el caso de que la variable C admita r posibles valores, y las variables predictoras sean dicotómicas–, es decir el modelo necesita las siguientes probabilidades:

$$P(C = \theta_i \mid X_1 = x_{1j}, ..., X_n = x_{nj})$$

y por otra parte el modelo que hemos denominado *Naive-Bayes*, en el cual se hace necesario el calcular:

$$P(C = \theta_i \mid X_1 = x_{1j}, ..., X_n = x_{nj}) \propto P(C = \theta_i) \times \prod_{r=1}^{n} P(X_r = x_{rj} \mid C = \theta_i),$$

y por tanto no necesitaríamos más que $(r-1)+n$ probabilidades.

Veamos escuetamente lo propuesto por Pazzani, apoyándonos en un simple ejemplo. Supongamos un dominio con 4 variables predictoras X_1, X_2, X_3, X_4 y una variable a predecir C. Supongamos asimismo que la variable X_2 no es relevante para C, y que además las variables X_1 y X_3 son condicionalmente dependientes dada C. Tendríamos una situación que gráficamente puede ser expresada según la red Bayesiana central de la Figura 4.3.

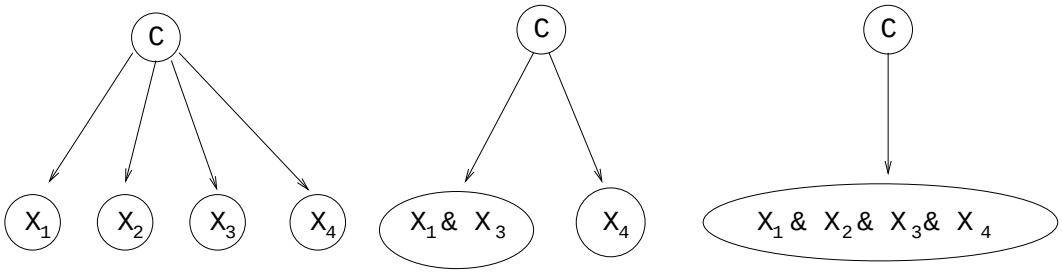


Figura 4.3: Ejemplo de partición del conjunto de variables.

A nivel de fórmulas esto queda expresado de la siguiente forma:

$$P(C = \theta_i \mid X_1 = x_{1j}, ..., X_4 = x_{4j}) \propto$$

$$P(C = \theta_i) \cdot P((X_1 = x_{1j}, X_3 = x_{3j}) \mid C = \theta_i) \cdot P(X_4 = x_{4j} \mid C = \theta_i).$$

Queda por determinar que variables son no relevantes, y cuales de las variables van a agruparse y necesitar que se calcule para las mismas las probabilidades condicionadas correspondientes.

4.2.5 Aproximación Markov Blanket

El método *naive Bayes* tiene en consideración el hecho de que existe una variable especial en el problema a tratar, que es la que representa a la clase, esto es, la que indica la clasificación (real o predicha) de un caso determinado. En cualquier caso, debido a su simplicidad estructural, no alcanza a considerar de forma adecuada la semántica intrínseca de las RRBB.

Teniendo en cuenta que en una red Bayesiana –véase Figura 4.4– cualquier variable tan sólo se encuentra influenciada por el denominado *Markov blanket* (MB) relativo a la misma –es decir por el conjunto de sus variables madres, sus variables hijas, y por las variables que son madres de las hijas–, parece intuitivo tener en cuenta modelos clasificatorios que sean *Markov blanket* de la variable a clasificar. Se buscan modelos en los que **todas las variables** forman parte del MB de la variable a clasificar. Esto es debido a que, a pesar de que se utilizan técnicas de aprendizaje automático, el experto ha sido el responsable del diseño de la base de datos de casos, sobre todo en lo que a la elección de las variables que se van a tener en consideración se refiere, de modo que podemos asumir que todas ellas tienen alguna influencia sobre la clasificación a realizar.

El concepto de *Markov blanket* asociado a una variable se ha utilizado en el denominado muestreo de Gibbs –véase por ejemplo Pearl (1987)–.

Existen varios procedimientos para buscar dentro del espacio de posibles *Markov blanket* de la variable a clasificar. Por ejemplo Sierra y Larrañaga (1998) utilizan los algoritmos genéticos para llevar a efecto tal búsqueda.

Aproximación Markov Blanket relajada

La aproximación *Markov blanket* presenta dos problemas principales. El primero de ellos se debe a que el espacio de búsqueda no se reduce de forma considerable, lo que lleva a un proceso de aprendizaje estructural lento y costoso. El segundo problema consiste en que se ha detectado durante la fase experimental que la aproximación MB tiene una tendencia a sobreajustarse a los datos de entrenamiento, ofreciendo debido a ello peores resultados de los esperados con los datos del fichero de test.

Con el objetivo de paliar ambos problemas, se ha ideado una nueva aproximación que relaja las condiciones que deben de cumplirse para que una estructura sea considerada Markov Blanquet. Para ello, se añaden las siguientes dos restricciones:

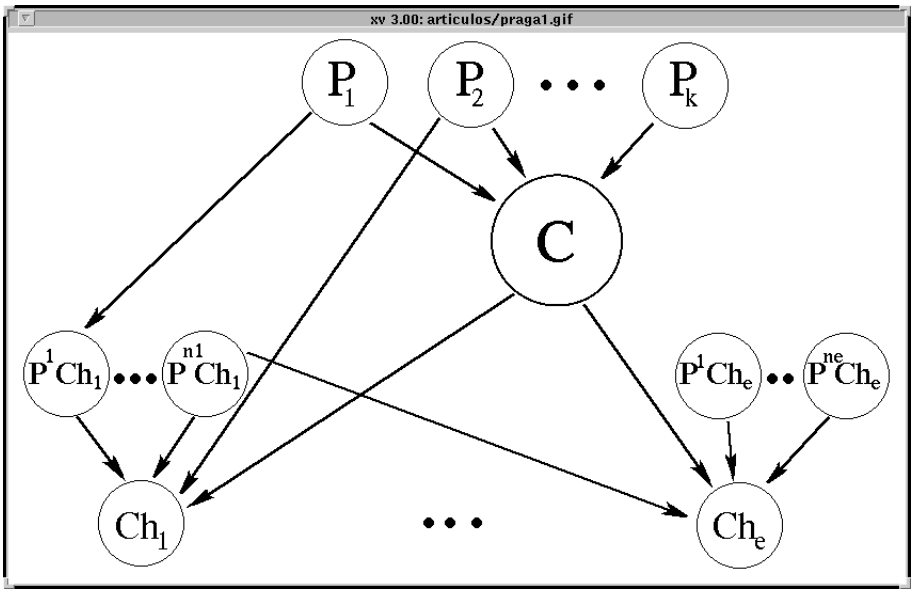


Figura 4.4: Markov blanket.

1. No es necesario que todas las variables del modelo formen parte del MB de la variable a clasificar. De este modo se reduce el espacio de búsqueda, a la vez que se realiza, de forma implícita una selección de las variables que intervienen en la clasificación.

2. No se permiten ciertas relaciones entre las variables que pueden aparecer en la aproximación MB:

   (a) Cualquier variable madre de la variable a clasificar, no puede ser a su vez madre de alguna que sea hija de esta variable especial.

   (b) Una variable que sea madre de la variable a clasificar no puede ser madre de un hijo de la variable a clasificar, y además una variable tan sólo puede ser madre de un hijo de la variable a clasificar.

   De este modo reducimos aún más el espacio de búsqueda, obteniendo además estructuras con menor tendencia al sobreajuste a los datos de entrenamiento.

4.3 Algoritmos genéticos en la inducción de redes Bayesianas

Para realizar la búsqueda en el espacio de modelos, se ha utilizado un algoritmo genético diseñado específicamente para tratar este problema. En él, los individuos son estructuras de RRBB, y la

evaluación de cada una se realiza en función de su comportamiento como clasificador.

4.3.1 Notación y representación

Sea $\mathcal{D}$ el conjunto de estructuras de RB en un dominio determinado en el que intervienen n variables, y sea S el alfabeto binario, siendo $\{0,1\}$ los valores incluidos en el mismo. Una estructura de RB se puede representar mediante una matriz $n \times n$, M, tal que sus elementos, m_{ij}, verifiquen:

$$m_{ij} = \begin{cases} 1 & \text{si el nodo } i \text{ es padre del nodo } j, \\ 0 & \text{en caso contrario.} \end{cases}$$

En la Figura 4.5 se puede ver un ejemplo de una red Bayesiana con su representación matricial correspondiente.

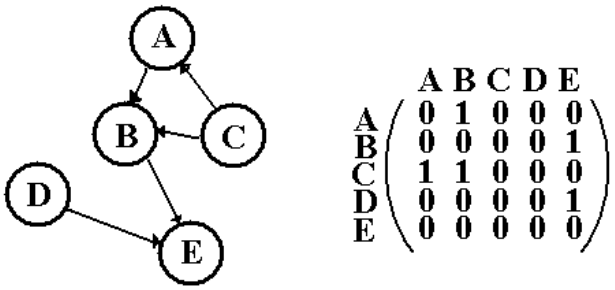


Figura 4.5: Ejemplo de representación matricial de la estructura de una red Bayesiana.

Para ver como funcionan los operadores genéticos con este tipo de individuos, vamos a estudiar dos posibles casos: que se asuma un orden entre las variables, o que esta asunción no sea necesaria.

Asumiendo un orden entre los nodos

En este caso, las matrices binarias asociadas a las estructuras son triangulares superiores, esto es, matrices en las que los elementos situados bajo la diagonal principal son, todos ellos, 0. Debido a esta propiedad, los operadores genéticos son cerrados respecto a las condiciones de DAG. Dicho de otro modo, la operación de cruce en un punto de dos individuos que representan redes Bayesianas en las cuales se ha asumido un orden entre los nodos obtiene como resultado dos redes Bayesianas con estructura válida, esto es, dos DAGs.

Representamos un individuo mediante la cadena de caracteres binaria:

$$m_{12}m_{13}m_{14}\ldots m_{1n}m_{23}m_{24}\ldots m_{2n}\ldots m_{n-2n-1}m_{n-2n}m_{n-1n}$$

que es una forma de indicar las posiciones correspondientes a los elementos de la matriz binaria que se hallan por encima de la diagonal principal, siendo esto suficiente para representar toda la matriz, puesto que el resto de los elementos de la misma son ceros.

Veremos ahora como se comportan los operadores de cruce en un punto y de mutación con esta representación, utilizando para ello unos sencillos ejemplos.

Ejemplo 1

Consideremos un dominio con tres variables, en el cual se han definido las dos estructuras que se pueden observar en la Figura 4.6(a). Utilizando la mencionada representación, las redes se corresponden a las cadenas binarias 101 y 010, respectivamente.

Supongamos ahora que estas dos estructuras son cruzadas –cruce de un punto–, y que el punto de corte es el situado entre el primer y el segundo bit. Esto produce dos individuos hijo representados por las cadenas binarias 110 y 001. Las estructuras de RB asociadas vienen representadas en la Figura 4.6(b).

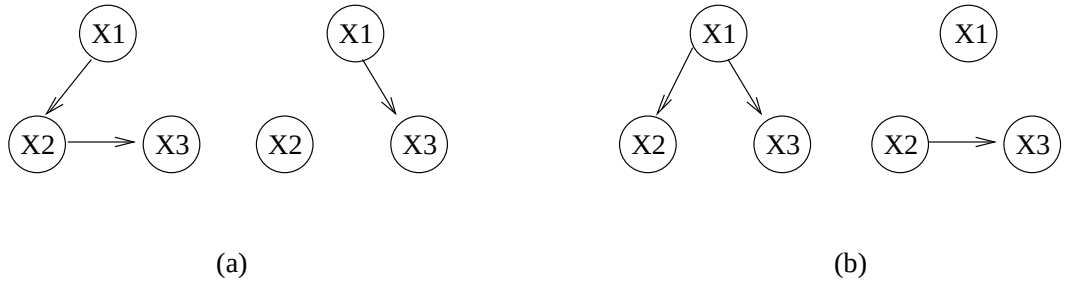


Figura 4.6: Asunción de orden: cruce basado en un punto entre dos estructuras de red Bayesiana.

Ejemplo 2

Consideremos ahora la estructura de RB representada en la Figura 4.7(a). La cadena binaria que la representa es 010. Supongamos que la operación de mutación simple afecta al valor del tercer bit, cuyo valor es alterado, pasando a ser 1. Esto da como resultado la cadena 011 que se corresponde a la estructura de RB que se muestra en la Figura 4.7(b).

Figura 4.7: Asunción de orden: mutación de una estructura de red Bayesiana.

No asumiendo un orden entre los nodos

Si no se realiza la asunción de orden entre las variables, los individuos de la población vienen representados mediante cadenas binarias de la siguiente forma:

$$m_{11}m_{12}\ldots m_{1n}m_{21}m_{22}\ldots m_{2n}\ldots m_{n1}m_{n2}\ldots m_{nn}$$

en las que se representa la matriz al completo.

Como se puede apreciar en los ejemplos que se dan a continuación, los operadores binarios no son cerrados, en este caso, con respecto a la condición de DAG necesaria para que un grafo dirigido pueda representar la estructura de una RB.

Ejemplo 3

Consideremos de nuevo un dominio de tres variables, en el cual se han definido dos estructuras de RB tal y como se puede apreciar en la Figura 4.8(a). Utilizando la representación binaria, las redes vienen definidas por las cadenas binarias 001000000 y 000001100. Al aplicar a estas dos representaciones binarias el cruce basado en un punto, siendo el punto de corte el situado entre los bits tercero y cuarto, obtenemos como representación binaria de los hijos las cadenas 001001100 y 000000000, cuyas representaciones gráficas asociadas se muestran en la Figura 4.8(b). La primera de las representaciones gráficas obtenida no es un DAG, ya que presenta un ciclo entre las variables X_1 y X_3.

Ejemplo 4

Consideremos ahora el DAG representado por la Figura 4.9(a). Su representación binaria viene dada por la cadena 000001100. Supongamos que el operador de mutación modifica el valor del primer

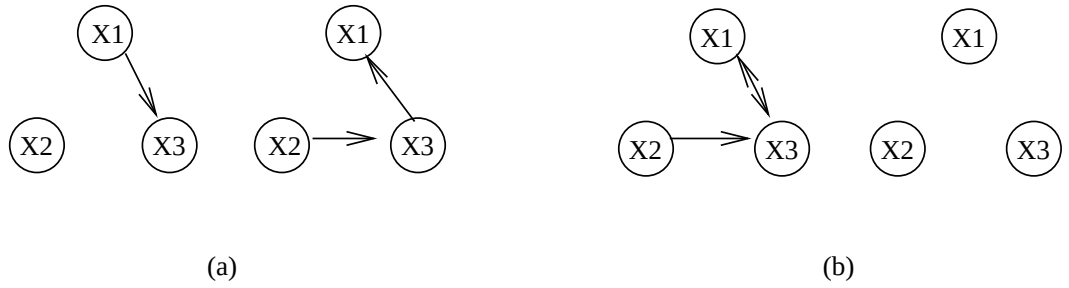


Figura 4.8: Sin asunción de orden: cruce basado en un punto entre dos estructuras de red Bayesiana.

bit, obteniéndose de este modo la cadena 100001100, que corresponde a la representación gráfica mostrada en la Figura 4.9(b). Se puede observar que el operador de mutación ha obtenido una estructura gráfica que presenta un ciclo, con lo que no es una estructura válida de red Bayesiana.

Figura 4.9: Sin asunción de orden: mutación de una estructura de red Bayesiana.

Para poder utilizar el algoritmo genético, a pesar de que los operadores genéticos no son cerrados en las estructuras DAG, necesitamos introducir un *operador de reparación*, que transforma las estructuras que no satisfacen la condición de DAG (bien obtenidas mediante el cruce, bien mediante la mutación) en estructuras que si la satisfacen. Dicho operador de reparación actúa eliminando de forma aleatoria los arcos que forman parte de los ciclos hasta que la estructura se convierte en un DAG.

Esta aproximación ha sido evaluada empíricamente mediante la simulación de la red ALARM (Beinlich y col., 1989). Los resultados obtenidos pueden verse en Larrañaga y col. (1996a). Otra aproximación, en la que los individuos de la población son posibles órdenes entre las variables, ha sido propuesta también por Larrañaga y col. (1996b).

Reparadores específicos

Tanto la aproximación TAN-GA como la MB-GA requieren de reparadores específicos, ya que los operadores de cruce basado en un punto y mutación por alteración del valor de un bit no son cerrados en el espacio de búsqueda de las estructuras buscadas, esto es, el cruce basado en un punto de dos estructuras TAN no produce necesariamente una estructura TAN, al igual que el cruce basado en un punto de dos estructuras MB no produce necesariamente una estructura MB. Lo mismo sucede con el operador de mutación utilizado.

Para resolver este problema se introduce un operador de reparación que convierte los individuos obtenidos mediante el cruce o la mutación en estructuras válidas dentro del espacio de búsqueda en que nos estemos moviendo. Esto se hace en ambos casos mediante la adición y eliminación de arcos, de forma aleatoria, hasta que se consigue una estructura válida. Obviamente, no todos los arcos pueden ser eliminados, ya que en las estructuras TAN los que van del nodo asociado a la variable que representa a la clase al resto de nodos se deben mantener siempre. Vemos los pasos a seguir para la reparación de los individuos en ambos casos:

- TAN
    1. Añadir los arcos no existentes entre el nodo que representa a la clase y el resto de nodos
    2. Eliminar, aleatoriamente, los arcos entre los nodos que representan a las variables aleatorias hasta que se obtenga una estructura sin ciclos.
- MB
    1. Si existen nodos fuera del MB correspondiente a la clase, conectarlo aleatoriamente como hijo, padre, o padre de un hijo de la misma
    2. Eliminar aleatoriamente los arcos que generen ciclos, sin que por ello quede algún nodo fuera del MB del nodo asociado a la variable clase.

4.3.2 Algoritmo genético para hallar la partición óptima de variables

Mención especial recibe el algoritmo genético implementado para hallar la partición óptima de variables, en base a la idea de Domingos y Pazzani (1996), ya que en este caso las estructuras de red Bayesiana son fijas –modelo *naive-Bayes*– y el espacio de búsqueda es el de las posibles particiones que se pueden realizar. En la implementación realizada, el tamaño de los individuos puede ser variable, representando cada uno de ellos una partición del conjunto de variables. Al poder ser el número de subconjuntos diferente en cada individuo, el cruce es específico para esta

representación, mientras que la mutación consiste en cambiar una variable de un subconjunto a otro, de forma aleatoria.

El operador de cruce elige una de entre las dos cardinalidades de subconjuntos, y va rellenándolos eligiendo entre los subconjuntos de los padres. Si quedan subconjuntos vacíos elige variables de otros subconjuntos y las mueve a los mismos, todo ello, por supuesto, al azar.

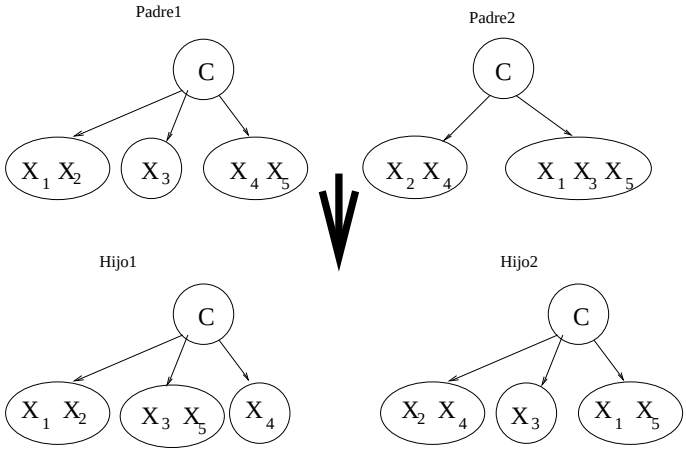


Figura 4.10: Cruce entre dos posibles particiones.

En la Figura 4.10 se muestra un ejemplo de cruce entre dos posibles particiones. Para cada hijo, obtenemos al azar un número de particiones –igual al de alguno de los padres–. En el ejemplo, hay que elegir ente tres y dos particiones. Supongamos que en ambos casos obtenemos un tres, aunque el sorteo se hace independientemente para cada uno de los hijos. A continuación elegimos aleatoriamente las particiones de los padres para rellenar las de los hijos, hasta que se han elegido todos los nodos. Si alguno de los nodos de la partición elegida ya esta en las anteriores, se elimina de la nueva partición. En el ejemplo de la Figura 4.10, el hijo 1 surge de la siguiente manera: se elige la primera partición de padre 1, con lo que $\{X_1, X_2\}$ forman el primer subconjunto. A continuación se elige el segundo nodo del segundo padre, con lo que las variables $\{X_3, X_5\}$ formarán el segundo subconjunto –x_1 se elimina, al estar en un subconjunto anterior–. A continuación se van eligiendo subconjuntos restantes de los padres hasta que en uno de ellos aparece la variable X_4 –la restante–, que pasa a ser la única en el tercer subconjunto. De forma análoga se genera el segundo hijo.

Parte IV

ALGORITMOS DE CLASIFICACIÓN POR VECINDAD

Capítulo 5

Algoritmos de clasificación por vecindad

5.1 Introducción

Dentro de las técnicas de *clasificación supervisada*, una de las aproximaciones más conocidas es la que se basa en criterios de vecindad (Dasarathy, 1991). Bajo esta perspectiva, los métodos de clasificación exigen la definición de una cierta medida de disimilitud (o distancia) entre los distintos elementos del espacio de representación. Es decir, precisan de la definición de una métrica que ayude a comparar las distancias entre los distintos objetos.

La ventaja mas inmediata que presentan las técnicas de clasificación basadas en criterios de vecindad con respecto a otros métodos de clasificación, hace referencia a su simplicidad conceptual, que se podría resumir del siguiente modo: la clasificación de un nuevo punto del espacio de representación se calcula en función de las clases, conocidas de antemano, de los puntos más próximos a él.

Tal y como se desprende de la afirmación anterior, la idea fundamental sobre la que se apoyan estas técnicas de clasificación, se basa en que las muestras pertenecientes a una misma clase, probablemente se encontrarán próximas en el espacio de representación. En la Figura 5.1 se observa un ejemplo de la aplicación de esta regla.

En los algoritmos basados en vecindad, la decisión de clasificar un caso $\mathbf{x}$ en la categoría θ depende de una colección de N casos previamente clasificados $(\mathbf{x}_1, \theta_1), (\mathbf{x}_2, \theta_2), ..., (\mathbf{x}_N, \theta_N)$, y el proceso de clasificación puede dar lugar a errores. Este tipo de problemas se enmarca, como ya se ha mencionado previamente, en el dominio de la clasificación supervisada, y no existe un clasificador óptimo que resuelva satisfactoriamente todos los problemas que se pueden plantear.

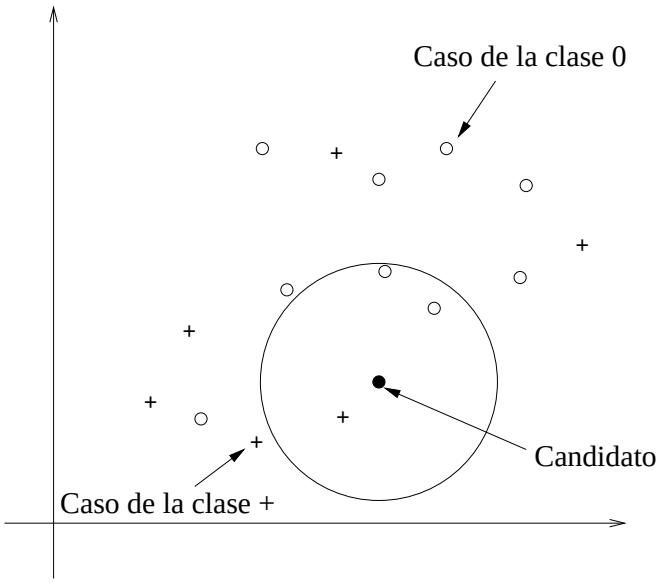


Figura 5.1: La regla NN frente a la regla K-NN (para K=3).

Si se asume que los casos ya clasificados $(\mathbf{x}_i, \theta_i)$ es hallan independiente e idénticamente clasificados respecto a la distribución de $(\mathbf{X}, \theta)$, se pueden establecer ciertos argumentos heurísticos para el desarrollo de buenos procesos de clasificación. Por ejemplo, parece razonable asumir que observaciones que se encuentran cercanas (en alguna métrica apropiada) tendrán aproximadamente la misma distribución de probabilidad a posteriori en sus respectivas clasificaciones.

De este modo, para clasificar el caso $\mathbf{x}$, podríamos querer asociar más peso a la evidencia de un caso conocido cercano $\mathbf{x}_i$. Quizás el procedimiento de decisión no paramétrico más simple es el del vecino de distancia mínima, o *nearest neighbour* (NN), que asigna a $\mathbf{x}$ la categoría de su vecino más cercano de entre los ya clasificados.

La primera formulación de una regla del estilo de NN parece haber sido debida a Fix y Hodges (1951), que investigaron el método conocido como K Nearest Neighbors (K-NN), que asigna a un caso no clasificado la clase más fuertemente representada de entre sus K vecinos más cercanos.

En general, cualquier problema de clasificación abordado con un enfoque basado en criterios de vecindad se puede caracterizar del siguiente modo:

1. Se dispone de un conjunto de N prototipos (o muestras ya clasificadas) llamado conjunto de entrenamiento.

2. Tenemos que clasificar un nuevo caso, **x**, no perteneciente al conjunto de entrenamiento.
3. Existe una métrica entre los diferentes objetos del espacio de representación.
4. No se utiliza ninguna otra información acerca de la distribución de los parámetros estadísticos asociados al conjunto de entrenamiento.

Vistas estas características, algunos autores distinguen los algoritmos de clasificación por vecindad del resto de los algoritmos de clasificación supervisada aduciendo que, mientras en el resto de los algoritmos de clasificación se realiza primero una inducción del modelo (árbol de clasificación, reglas que se utilizarán para clasificar nuevos casos, etc.), y utilizan posteriormente el modelo inducido para clasificar los nuevos casos, esto es, *inducción + deducción*, en los algoritmos basados en vecindad el modelo se halla implícito en los datos, sin llegar a explicitarse, con lo que se dice que se realiza una *transducción*.

5.1.1 Regla del vecino más próximo

En este caso, el clasificador asociará al caso **x** la clase verdadera del objeto que se encuentra más próximo a **x** dentro del espacio de representación. Para esta aproximación, todos los prototipos del conjunto de entrenamiento participan en la decisión de clasificación de la nueva muestra, esto es, el modelo de clasificación está compuesto por todos los casos que componen el conjunto de entrenamiento.

Un aspecto a tener en cuenta es que la efectividad de esta regla está condicionada a que se disponga de un número suficientemente grande de prototipos en el conjunto de entrenamiento. Desde un punto de vista práctico, este hecho puede representar un serio inconveniente en cuanto al coste computacional requerido para buscar el prototipo más próximo dentro del conjunto de entrenamiento. No obstante, este inconveniente se puede mitigar con la utilización de alguno de los procedimientos existentes para reducir el número de casos que componen el modelo, de forma que éste no esté compuesto por todos los casos que forman el conjunto de entrenamiento, sino por un subconjunto de los mismos.

5.1.2 Regla de los K vecinos más próximos (K-NN)

Por medio de esta regla, la clase asignada a un nuevo caso **x** será la clase más votada entre los K vecinos más próximos del conjunto de entrenamiento.

Sin embargo, cabe la posibilidad de que se produzca un empate en el número de votos de la clase más votada. En ese caso, para seleccionar la clase a asignar, es necesario utilizar algún método de

resolución de empates. Entre ellos, podemos encontrar los siguientes: asignar al caso a clasificar la clase que tenga el primer vecino más próximo de entre las empatadas a votos, seleccionar como clase a asignar aquella de entre las empatadas cuya media de distancias de sus vecinos sea menor, etc.

La principal ventaja de la *regla* K-NN con respecto a la *regla* NN radica en que, al utilizar varios vecinos en lugar de uno sólo para la clasificación de un nuevo caso, se aprovecha de una forma más eficiente la información que se puede extraer del conjunto de entrenamiento.

Sin embargo, esto no quiere decir que el método K-NN sea siempre más exacto que el método NN, sino que se ha comprobado empíricamente que el primero comete menos errores de clasificación que el segundo, para algún valor de la constante K. Es de destacar el hecho de que no existe monotocidad del porcentaje de bien clasificados obtenidos respecto al valor de K. En la literatura se indica que, en general, los mejores resultados se obtienen con K=3 o K=5, aunque esto depende en gran medida del problema concreto que se desea abordar.

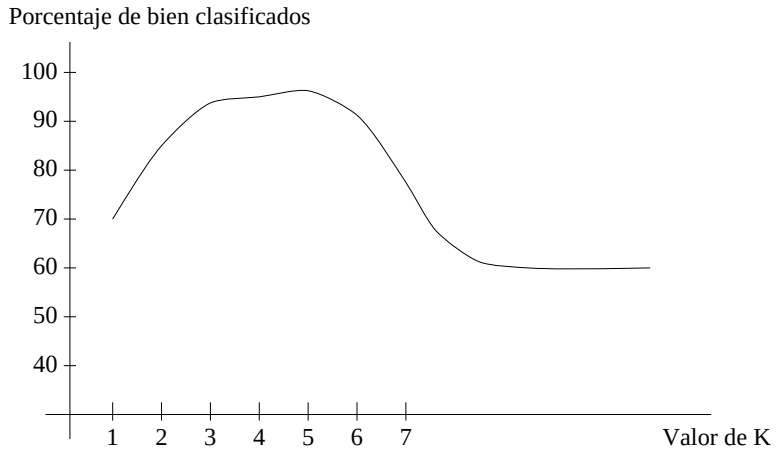


Figura 5.2: Comportamiento esperado del algoritmo K-NN respecto al valor del parámetro K.

En la Figura 5.2 se puede observar cual es el comportamiento típico del algoritmo K-NN para diferentes valores del número de vecinos a tener en cuenta, en problemas clasificatorios de dos clases. Para un número mayor de clases, la gráfica resultante es parecida, desplazada hacia la derecha. En cualquier caso esta gráfica es sólo indicativa, ya que para cada problema a abordar el comportamiento puede diferir en gran medida respecto al aquí mostrado.

5.2 Variantes del algoritmo K-NN

Dentro de los métodos de clasificación basados en criterios de vecindad, podemos encontrar varios clasificadores diferentes, algunos de los cuales se presentan en la siguiente sección. El libro de Dasarathy (1991) contiene una excelente recopilación de los artículos que constituyen las aportaciones más importantes realizadas –entre los años 1950 y 1990– en el área de las técnicas de clasificación basadas en criterios de vecindad. En los primeros artículos sobre el tema se denominaba *regla del vecino más próximo* a lo que habitualmente se conoce como algoritmo de clasificación supervisada *nearest neighbour* o simplemente NN. Es por este motivo que presentamos algunos de éstos algoritmos bajo el epíteto *regla*, sin pretender por ello entrar en consideraciones filosóficas acerca de la conveniencia o no de dicho calificativo. El contexto debe ser lo suficientemente claro como para evitar equívocos con las denominadas *reglas de clasificación*, que, como se ha visto en el Capítulo 3, constituyen un paradigma con entidad propia dentro del mundo de la clasificación supervisada.

Los algoritmos basados en vecindad basan su efectividad en la utilización de una métrica adecuada para el problema que se esté abordando. En sus primeras versiones, los algoritmos de la familia NN consideran por igual todas las variables que intervienen en el modelo a la hora de calcular la distancia, pero desde hace tiempo (Cost y Salzberg, 1993, Kohavi y Sommerfield, 1995) se intenta asociar a cada variable un peso distinto en el cálculo de las distancias dependiendo de su importancia en la determinación del valor de la variable a clasificar. Otras versiones asignan a cada caso de entre los K vecinos un peso diferente en la votación, según diferentes criterios (cercanía, información mutua, ...). Existen otra serie de versiones que lo que tratan es de reducir el número de cálculos necesarios para encontrar los K vecinos, y otras técnicas, presentadas en el siguiente capítulo, tratan de reducir el tamaño del modelo, esto es, de la Base de Datos de casos que se va a utilizar para hallar los vecinos, eligiendo una serie de prototipos que forman un subconjunto de la Base de Datos de entrenamiento y que son los que posteriormente se utilizarán para realizar las clasificaciones. Todas estas técnicas se pueden hibridar de muchas maneras, lo que nos da una idea de lo complejo que puede llegar a ser un sistema de clasificación por vecindad.

5.2.1 Regla K-NN con rechazo

Basándose en lo que ocurre con la regla de decisión de Bayes, esta variante del K-NN descarta la clasificación de aquellas muestras para las que no se obtenga una cierta garantía de que la clase asignada sea la correcta.

Aplicando esta regla, la clasificación sólo será realizada en el caso de que alguna de las clases reciba un número de votos superior a un umbral previamente fijado, por lo que si resulta que

el número de votos recibidos por la clase más votada no supera dicho umbral, el caso no será asignado a ninguna de las clases del problema. Los casos que quedan sin clasificar son aquellos dudosos, que probablemente se encuentren próximos a las fronteras de decisión.

El umbral que se establece para esta regla suele oscilar entre K/M y K, siendo M el número de clases del problema de clasificación que se desea abordar, por lo que se le conoce como umbral para la mayoría para la votación entre los K vecinos más próximos. Si ninguna clase supera este umbral, el caso a clasificar será asignado a la clase de rechazo w. En la Figura 5.3 se pueden observar dos ejemplos de la aplicación de esta regla.

Una alternativa a la regla K-NN con rechazo, pero con una clara herencia conceptual de ésta, es la de establecer un tipo de *mayoría absoluta* para el número de votos correspondientes a la clase mayoritaria. Así el nuevo caso a clasificar sólo será asignado a la clase más votada, en el caso en el que la diferencia de votos entre ésta y las demás clases supere un determinado número reflejado en un umbral absoluto.

Se busca que la aplicación de este criterio proporcione a la regla K-NN una mayor seguridad a la hora de establecer la clase de los casos correspondientes al conjunto de testeo. Para la aplicación de esta regla hay que tener en cuenta que se debe fijar un valor para K lo suficientemente grande para que el umbral absoluto establecido no conlleve el rechazo de un número elevado de casos en el proceso de clasificación, y a la vez éste valor de K no debe ser lo suficientemente grande como para incurrir en demasiados errores (ver gráfica en la Figura 5.2).

5.2.2 Regla K-NN por distancia media

Nace como una alternativa a la regla K-NN, dándose en este caso mayor relevancia a la distancia a la que se encuentran los K vecinos más próximos que al voto individual de cada vecino. A partir de los K vecinos más próximos, a un nuevo caso a clasificar le es asignada la clase cuya distancia media es menor dentro de las clases de los K vecinos más próximos.

5.2.3 Clasificador de la distancia mínima

Para la aplicación de este método, la primera tarea consiste en seleccionar un prototipo o representante para cada clase. Esta tarea tiene una gran importancia ya que los resultados obtenidos por el método van a depender de los representantes elegidos para las distintas categorías.

Una vez determinados los distintos representantes, el clasificador asignará a un nuevo caso $\mathbf{x}$ la clase cuyo representante se encuentre más próximo a ella.

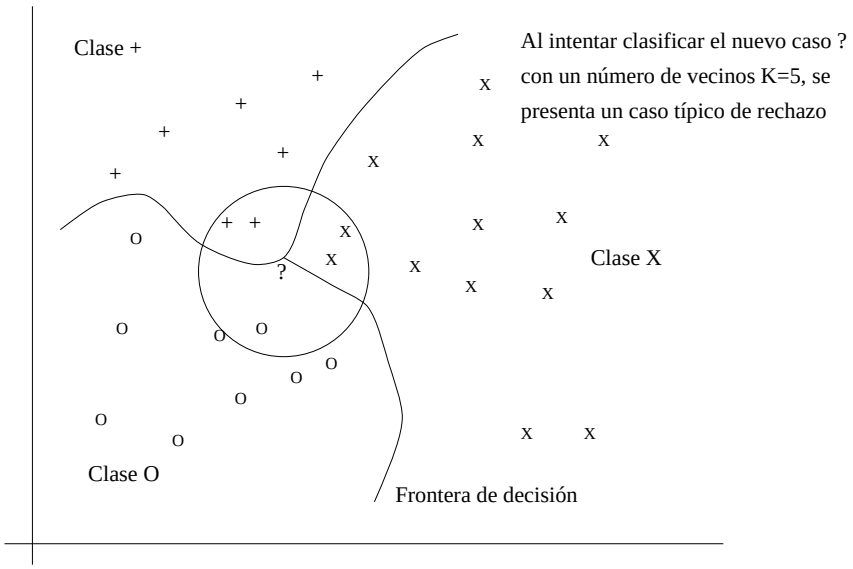


Fig 5.3a

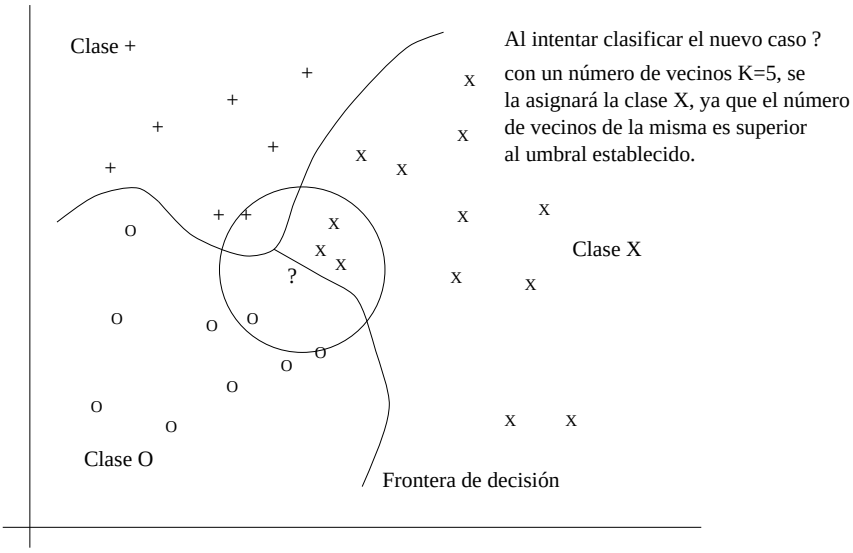


Fig 5.3b

Figura 5.3: Frontera de decisión. En la Figura 5.3a el caso sería rechazado con K=5 y un umbral fijado en 2. En la Figura 5.3b se le asignará como clasificación la clase X.

Dada su simplicidad, y la reducida necesidad de cómputo para su cálculo, la efectividad de este método se verá seriamente condicionada por la estrategia de representar cada categoría del problema por medio de un único prototipo. Concretamente, este criterio de clasificación no resultará apropiado en el caso de que exista más de un agrupamiento o clustering para una determinada clase.

Clasificador de la distancia mínima

COMIENZO
Como entrada disponemos de:
El modelo, conteniendo M casos, uno por cada clase $(\mathbf{x}_i, \theta_i), i = 1, ..., M$,
y un nuevo caso $(\mathbf{x}, \theta)$ que se desea clasificar

PARA cada caso del modelo $(\mathbf{x}_i, \theta_i)$ HACER
COMIENZO
Calcular la distancia a $\mathbf{x}$ del caso
Sea D_i dicha distancia
FIN
Como salida, dar la clase θ_j cuya distancia D_j es
la mínima de entre todas las clases
FIN

Figura 5.4: Pseudocódigo del algoritmo *clasificador de la distancia mínima.*

En la Figura 5.4 se muestra el pseudocódigo del algoritmo de clasificación por la distancia mínima. La dificultad radica en la elección del único representante de cada clase.

PEBLS

El algoritmo PEBLS (Cost y Salzsberg, 1993), asocia un peso a cada instancia, que es utilizado como coeficiente en el cálculo de las distancias entre los diferentes casos. Estos pesos reflejan el comportamiento clasificatorio histórico de cada instancia. El peso es el porcentaje de veces que el uso de la instancia ha obtenido una clasificación correcta de un caso respecto al número de veces que la instancia ha sido usada para realizar alguna clasificación. El peso definitivo que se asigna a una instancia depende del orden en que se seleccionan las mismas. PEBLS las selecciona aleatoriamente, y da varias pasadas para evitar el influjo del orden.

Nearest Centroid Neighbourhood

El concepto *Nearest Centroid Neighbourhood* (NCN), (Chaudhuri, 1996), está basado en la idea de que la vecindad de un caso dado se halla sujeta a las siguientes dos restricciones:

1. Debido a la utilización del criterio de la distancia, los K vecinos de un caso dado, $\mathbf{x}$, deben estar lo más cerca posible.

2. Debido al criterio de simetría, su centroide debe estar lo más cerca posible al caso $\mathbf{x}$.

El algoritmo K-NN toma en consideración el primer criterio, y los K vecinos no tienen porque estar distribuidos de forma simétrica respecto a $\mathbf{x}$. En la aproximación NCN, sin embargo, se utiliza un tipo de vecindad que tiene en cuenta la situación geométrica de los vecinos que se van hallando respecto al punto $\mathbf{x}$, de forma que su centroide sea lo más cercano posible al mismo. La forma de hacerlo es la siguiente:

1. El vecino más cercano al caso a clasificar $\mathbf{x}$ es el primer caso que se considera q_1.

2. El i-ésimo vecino, $q_i, i \geq 2$ es aquel que hace que el centroide de los $i-1$ vecinos hallados previamente, y él mismo, es decir, del conjunto Q_i, $Q_i = \{q_1, q_2, ..., q_{i-1}, q_i\}$, sea lo más cercano posible a $\mathbf{x}$.

Clasificador Nearest Centroid Neighbourhood

COMIENZO
  Como entrada disponemos de:
    La Base de Datos de entrenamiento D, conteniendo N casos, $(\mathbf{x}_i, \theta_i), i = 1, ..., N$,
    Un nuevo caso $\mathbf{x}$ a clasificar

  Hallar los K vecinos con centroide más cercano a $\mathbf{x}$
  Asignar a $\mathbf{x}$ la clase más representada de entre estos k vecinos
FIN

Figura 5.5: Pseudocódigo del algoritmo de clasificación *K-NCN*.

Esta idea, cuyo algoritmo asociado se muestra en la Figura 5.5, puede ser utilizada para clasificar el nuevo caso $\mathbf{x}$. En concreto, el clasificador K-NCN (Sánchez y col., 1997), trata de

aplicar esta idea para mejorar los resultados clasificatorios del algoritmo NN. Para ello, se propone predecir la clase de un nuevo caso de acuerdo con la distribución espacial de los prototipos a su alrededor. Esto significa que los casos son clasificados teniendo en cuenta, no sólo sus K vecinos más próximos, sino también la forma en que éstos se sitúan alrededor de él. A este concepto se le ha denominado *vecindad envolvente*.

5.2.4 Reducción en el cálculo de las distancias

A continuación se presentan tres algoritmos de búsqueda del vecino más próximo en espacios métricos que mediante una fase de preproceso reducen significativamente el número de distancias calculadas para hallarlo. Así los algoritmos aquí expuestos, pertenecen a la familia AESA (Approximating Eliminating Search Algorithm) y pueden estudiarse con mayor detalle en (Micó, 1996). Estos algoritmos se basan, para conseguir una mayor eficiencia en cuanto al número de distancias calculadas para hallar el vecino más próximo a un caso candidato a ser clasificado, en la desigualdad triangular propia asociada a cualquier métrica.

Estos tres algoritmos reciben el nombre de AESA, LAESA y TLAESA, habiendo sido diseñados los dos últimos con el objetivo de mejorar las prestaciones y limitaciones que presenta el AESA. Existen otras versiones de algoritmos que tienen el mismo objetivo de reducción del coste computacional asociado al cálculo de las distancias en este tipo de algoritmos. Así, en Fukunaga y Narendra, (1975), Friedman y col., (1975) y Yunk, (1976), encontramos otro tipo de aproximaciones.

AESA

El Algoritmo AESA (Approximating Eliminating Search Algorithm) es propuesto en Vidal (1986), y consta principalmente de dos fases: aproximación y eliminación. Para lograr reducir significativamente el número de distancias calculadas en la fase de búsqueda del vecino más próximo al caso a clasificar que se está tratando en este momento, utiliza una matriz triangular de $\frac{\|N\|(\|N\|-1)}{2}$ distancias entre los prototipos del conjunto de entrenamiento. Dicha matriz es calculada en una fase de preproceso.

En una primera fase de aproximación se elige un caso, (también llamado prototipo), que se encuentre lo *bastante* cercano al caso a clasificar, se calcula la distancia entre ambos y se actualiza el vecino más próximo encontrado hasta ahora. En la segunda fase, la de eliminación, se vale de la desigualdad triangular asociada a toda métrica para eliminar prototipos que no puedan estar más cercanos al caso al que se pretende asignar una clasificación, que el prototipo más cercano hallado hasta ese momento, terminando el proceso de clasificación del nuevo caso cuando el conjunto de

prototipos de entrenamiento queda vacío, bien porque han sido eliminados, bien porque se les ha seleccionado para calcular su distancia al caso a clasificar.

El principal problema que se encuentra en la aplicabilidad del AESA, es el alto coste espacial que supone el tener guardadas todas las interdistancias de los prototipos del conjunto de entrenamiento, lo que lo hace inviable en problemas con conjuntos de entrenamiento de cierta magnitud.

LAESA

El algoritmo LAESA (Linear Approximating-Eliminating Search Algorithm) es un algoritmo de búsqueda de vecinos más próximos en espacios métricos, que nace con el objetivo de suplir las limitaciones de su antecesor, el AESA, en cuanto a espacio de almacenamiento, pero manteniendo las buenas prestaciones de éste en cuanto a número de distancias promedio calculadas. En concreto, el LAESA reduce el espacio de almacenamiento requerido a cotas lineales con respecto al tamaño, N, del conjunto de entrenamiento. Para ello se guardan las distancias entre un reducido subconjunto de prototipos, llamado conjunto de prototipos base, e independiente además del tamaño de N, y todos los prototipos del conjunto N. A costa de reducir el tamaño de la matriz de distancias calculadas en la fase de preproceso se produce un empobrecimiento de la cantidad de información manejada posteriormente en la fase de búsqueda del algoritmo, que se deberá adaptar a esta circunstancia.

El LAESA sigue utilizando la información calculada y almacenada en la fase de preproceso durante la fase de búsqueda. Mediante la desigualdad triangular asociada a la métrica se podrá seguir calculando una cota inferior de la distancia de cada prototipo al nuevo caso. Esta cota inferior se utiliza durante las fases de aproximación (elección de un prototipo que se halle muy cercano al caso a clasificar con una alta probabilidad) y eliminación (de todos aquellos prototipos que no puedan estar más cercanos al caso a clasificar que el más cercano hasta el momento) del algoritmo.

Dentro de las posibles mejoras que se pueden realizar al algoritmo LAESA destacan la forma de llevar a cabo la gestión de los prototipos base.

TLAESA

El algoritmo TLAESA (Tree Linear Approximating-Eliminating Search Algorithm) se basa únicamente en las propiedades métricas del espacio de representación, y utiliza una estructura arborescente sobre la que realizar la búsqueda. Se intenta reducir así el coste computacional lineal (el asociado a encontrar el vecino más próximo mediante la eliminación de todos los prototipos del

conjunto de entrenamiento, bien mediante el cálculo de la distancia de los prototipos al nuevo caso, o mediante eliminación a través de la función cota inferior de la distancia) del algoritmo LAESA. Este nuevo algoritmo utiliza, al igual que su predecesor LAESA, un subconjunto reducido de prototipos del conjunto original, denominado conjunto de prototipos base, respecto del cual se calculan y se almacenan las distancias al resto de prototipos. En esta aproximación se pretende que el número de distancias calculadas permanezca constante con el tamaño del conjunto de entrenamiento, al igual que sucedía en LAESA.

En este caso, además de la matriz de distancias entre los prototipos base y el conjunto de todos los prototipos, se necesitará un árbol binario donde se almacenan todos los prototipos del conjunto de entrenamiento. En la fase de búsqueda del vecino más próximo se utilizará el árbol, valiéndose de la matriz de distancias para evitar la exploración de aquellas ramas descartadas para albergar en su interior al vecino más próximo.

El cálculo del número de distancias para el algoritmo TLAESA es mayor que para LAESA, ya que la aproximación y eliminación se realizan en función de los representantes de cada nodo y no para cualquier prototipo; en cambio el coste computacional asociado al TLAESA es sublineal con respecto al número de prototipos del conjunto de entrenamiento. El rendimiento de uno u otro algoritmo dependerá del problema. En problemas en los que el tamaño del conjunto de prototipos sea mediano o las distancias utilizadas sean caras, será preferible el LAESA al TLAESA y viceversa.

5.3 Pesado de casos y de atributos

En sus primeras versiones, los algoritmos basados en la vecindad de los casos no tenían en cuenta el hecho de que generalmente no todas las variables influyen de la misma manera en la predicción de la clase, ni todos los casos aportan la misma información. Dado que este tipo de algoritmos basan su modelo clasificatorio en todos los casos de entrenamiento junto con todas las variables que aparecen en los mismos, no realizan de forma implícita una selección de variables y de casos, tal y como es habitual en otros paradigmas de clasificación supervisada (árboles de clasificación, inducción de reglas, ...). Rápidamente surgieron versiones del algoritmo que tienen esto en cuenta de alguna u otra manera. En esta sección se presentan algunos de los principales métodos de pesado de casos y atributos para el algoritmo K-NN encontrados en la literatura. En Wettschereck, (1994) y en Kohavi y col., (1996) se pueden ver revisiones más profundas sobre el tema.

5.3.1 Pesado de casos

Casi desde el descubrimiento de la regla, se empezaron a realizar versiones del algoritmo K-NN que daban distinto peso a los diferentes vecinos encontrados en base a la distancia a la que se encuentran del nuevo caso a clasificar (Dudani, 1975), suponiendo que los casos más cercanos deben tener mas influencia a la hora de realizar la clasificación. De este modo, y suponiendo que los casos $(\mathbf{x}_1, \theta_1), ..., (\mathbf{x}_K, \theta_K)$ son los K vecinos más cercanos a un caso $\mathbf{x}$ a clasificar, en un problema de clasificación con M clases, al caso $\mathbf{x}$ se le asignará la clasificación más votada de entre estos vecinos, teniendo en cuenta que el voto que aporta cada uno de los vecinos viene ponderado por un peso $W_i, i = 1, ..., K$ que no tiene porque ser el mismo para todos ellos. Esto puede verse como una generalización del K-NN inicial, considerando que en éste todos los vecinos tienen el mismo peso, es decir $W_i = 1$.

El peso a asociar a cada uno de los K casos vecinos se puede tener en consideración de muy diversas maneras.

Algunas de ellas pueden ser:

- El voto que aporta un caso dado es inversamente proporcional a la distancia a que se encuentra de la instancia a clasificar

$$W_i = \frac{1}{Dist(\mathbf{x}, \mathbf{x}_i)}.$$

  Donde $Dist(\mathbf{x}, \mathbf{x}_i)$ denota la distancia entre el i-ésimo vecino, $\mathbf{x}_i$, y el caso a clasificar, $\mathbf{x}$.

- Voto fijo según el orden de vecindad. En esta aproximación de dan los pesos fijos $W_1, W_2, ..., W_K$ a los vecinos primero, segundo, ..., K-ésimo, respectivamente, suponiendo que se encuentran ordenados de menor a mayor distancia respecto al caso a clasificar, $\mathbf{x}$, pero sin tener en cuenta el valor de la distancia que los separa. Obviamente, los pesos $W_1, W_2, ..., W_K$ son decrecientes.

- Voto ponderado según las probabilidades a priori de las clases a las que pertenecen los datos en caso de empate. De este modo, en caso de empate se elige la clase menos probable, ya que es la que más peso tiene.

5.3.2 Pesado de atributos

Vamos a considerar diferentes alternativas en la ponderación a la hora de calcular las distancias[1] en función de la importancia de las variables para realizar la clasificación.

[1] La distancia métrica más comúnmente utilizada es la Euclídea. Esta es la métrica que se ha venido utilizando en la mayoría de las versiones del algoritmo K-NN. Otras distancias que se utilizan en la literatura son las de Mahalanobis, la Euclídea ponderada, la de Hamming, etc.

Diferentes aproximaciones

Análogamente a lo sucedido con los casos vecinos a uno dado, se puede tener en cuenta el hecho de que no todas las variables influyen de la misma manera sobre la clase, y que por lo tanto habría que tener alguna forma de valorar más aquellas que más influencia tengan en la clasificación. Se tiene en cuenta este diferente nivel de influencia de las variables en el cálculo de las distancias. Para ello, consideremos un problema de clasificación en el que se han tenido en cuenta n variables predictoras $X_1, X_2, ..., X_n$, y que el peso atribuido a cada una de estas n variables es $W_1, W_2, ..., W_n$ respectivamente. Entonces, la distancia entre dos casos de este problema, $\mathbf{x} = (x_1, x_2, ..., x_n)$ e $\mathbf{y} = (y_1, y_2, ..., y_n)$ se calcula mediante la aplicación de la siguiente fórmula:

$$D_w(\mathbf{x}, \mathbf{y}) = \sum_{i=1}^{n} W_i (x_i - y_i)^2.$$

El cálculo de los diferentes pesos asociados a las diferentes variables puede ser realizado de muy diversas formas. Pasamos a enumerar algunas de ellas:

- El peso asociado a cada variable X viene determinado por la medida de la información mutua entre ella y la variable a clasificar C. La información mutua entre dos variables se define como la reducción en la incertidumbre concerniente a los posibles valores de una de las variables cuando se establece el valor de la otra. Si una variable determina exactamente la clase, la información mutua entre ambas es proporcional al logaritmo del número de clases, asumiendo que las instancias de cada una de las clases son igualmente frecuentes. La siguiente fórmula computa la información mutua entre dos variables discretas:

  $$I(X, C) = H(X) - H(X \mid C) = \sum_{x,\theta} \log p(x, \theta) \frac{p(x, \theta)}{p(x)p(\theta)}.$$

  El peso a establecer para la variable será proporcional al valor obtenido.

- Determinar un conjunto $\mathcal{W}^l = \{W_1, ..., W_l\}$ de l pesos fijos distintos, $l < n$, y asociar a cada una de las variables alguno de estos pesos según el resultado de algún tipo de test. Los ejemplos más sencillos de este tipo de pesado son:

1.- ($l = 2$) Establecer 2 pesos, $W_1 = 0, W_2 = 1$, de modo que las variables a las que se asocie un peso 1 son seleccionadas para intervenir en la realización de la clasificación, mientras que aquellas variables asociadas con un peso 0 han sido descartadas para participar en el proceso de clasificación. Esto es utilizado por algunos procesos de selección de variables, en lo que se conoce como *feature subset selection* (Inza, 1999b).

2.- ($l = 3$) Establecer 3 pesos, $W_1 = 0, W_2 = 0.5, W_3 = 1$, de modo que las variables son asociadas a alguno de ellos según su importancia en el proceso de clasificación. Se eligen estos tres pesos tan sencillos para evitar caer en un sobreajuse a los datos (*overfitting*) de los pesos calculados (Kohavi, 1997).

3.- ($l = m$) Esto se puede extender a cuantos valores se desee, siendo estos normalmente equidistantes en el intervalo $[0, 1]$.

De nuevo se pueden ver estas aproximaciones como generalizaciones del algoritmo K-NN, considerando que en su versión inicial todas las variables predictoras tienen asociado un peso $W_i = 1$.

5.4 Reducción del conjunto de entrenamiento

El costo computacional asociado a los algoritmos de clasificación por vecindad viene influenciado de forma cuadrática por el tamaño de la base de datos de entrenamiento, ya que todos los casos en ella contenidos forman parte del modelo de clasificación, esto es, dado un nuevo caso a clasificar, se calcula su distancia a todos y cada uno de los casos de dicha base de datos. Es por ello que desde hace más de 30 años (Hart, 1968) se viene investigando en la utilización de modelos que no estén compuestos por la totalidad de los casos previamente clasificados de que se dispone. A los casos que forman el subconjunto resultante de la reducción se les denomina **prototipos**. El problema es denominado en algunos artículos *problema de selección de referencias*, y los algoritmos utilizados para realizar la tarea de selección se encuentran bajo la denominación *algoritmos de edición* o *reglas de edición* (Dasarathy, 1991). Los algoritmos de edición se dividen en dos grupos. El primer conjunto de algoritmos fue desarrollado en el área de *reconocimiento de patrones* en investigaciones iniciadas en la década de los años 70. El segundo conjunto ha sido presentado de forma más reciente en el área de *aprendizaje automático* o *machine learning*, en investigaciones realizadas durante la década de los 90. A pesar de que los algoritmos de los diferentes grupos no presentan muchas diferencias (algunos son idénticos) se suelen presentar de forma separada por motivos históricos. Un estudio más profundo sobre algoritmos de edición puede encontrarse en Aha (1990), en Dasarathy (1991), o en Sánchez (1997).

Los algoritmos de selección de prototipos tienen dos objetivos, que son, por otro lado, los objetivos principales de cualquier algoritmo de clasificación. El objetivo primero es el de reducir el costo computacional asociado a la aplicación de las reglas de vecindad. El segundo objetivo es el de incrementar el poder clasificatorio del algoritmo. Freund y Shapire (1996) observan que, incluso después de la aplicación de un algoritmo de edición sencillo (*Condensed Nearest Neightbour*, Hart, 1968), los prototipos seleccionados continúan mostrando una tendencia a cubrir todo el conjunto de entrenamiento.

En general, existen tres aproximaciones a la hora de realizar algoritmos de selección de prototipos: (1) filtrado de casos, (2) búsqueda estocástica, y (3) ponderación de casos. La primera de las aproximaciones es la que presenta mayor volumen de publicaciones. Se utiliza algún tipo de regla para determinar, incrementalmente, qué casos de la base de datos de entrenamiento serán seleccionados como prototipos y cuales se descartarán del modelo. Algunos de los trabajos enmarcados en esta aproximación guardan instancias mal clasificadas (Hart, 1968, Gates, 1972, Aha, 1990); otros guardan instancias típicas (Zhang, 1992); algunos guardan sólo las instancias que han sido correctamente clasificadas utilizando el resto de las instancias (Wilson, 1972); otros se basan en conocimientos sobre el dominio (Kurtzberg, 1987); y otros combinan ambas técnicas (Voisin y Devijer, 1987). Algoritmos de selección estocástica de prototipos han sido aplicados en Skalak (1994) y Cameron-Jones (1995). Hay además otro tipo de aproximaciones que realizan la selección de prototipos mediante el cálculo de porcentajes de clasificación o mediante abstracciones (Djouadi y Boucktache, 1997, Sánchez y col., 1998, Micó, 1998, Lipowezky, 1998).

Presentamos a continuación algunos de los algoritmos más conocidos de selección de prototipos. Estos son las versiones más simples que aparecen en la literatura, existiendo variaciones más complejas de casi todos ellos.

5.4.1 Selección de prototipos

Con el fin de seleccionar un subconjunto de casos representativo del conjunto de entrenamiento de partida, se han desarrollado todo un conjunto de procesos previos al proceso de clasificación. Estos procesos se aplican antes de la regla NN con el objetivo por un lado de aumentar la efectividad del clasificador mediante la eliminación del conjunto de entrenamiento de prototipos erróneamente clasificados, y por otra reducir la talla resultante del conjunto de entrenamiento. La reducción de la talla del conjunto de entrenamiento va a resultar de crucial importancia para la reducción de la carga computacional inherente a las reglas de clasificación basadas en criterios de vecindad.

Las técnicas de selección de prototipos suelen clasificarse en 2 categorías dependiendo del fin que persigan con su aplicación. Por un lado están las técnicas encaminadas a eliminar prototipos erróneamente clasificados del conjunto de entrenamiento, y a la vez, eliminar los posibles solapamientos entre regiones de clases distintas en el espacio de representación. Estas técnicas son conocidas con el nombre de técnicas de edición. Por otro lado se encuentran aquellas que se centran en seleccionar un subconjunto suficientemente representativo del conjunto de prototipos inicial, de manera que el subconjunto de prototipos resultante proporcione un comportamiento a la regla NN similar al que ofrecía la totalidad del conjunto de entrenamiento inicial. Este segundo grupo de técnicas y procedimientos son conocidas como técnicas de condensación o condensado.

5.4.2 Técnicas de edición

Como se ha comentado anteriormente las técnicas de edición buscan eliminar prototipos que se encuentran en zonas del espacio de representación diferente al de la mayoría de los casos pertenecientes a su misma clase. La aplicación de este tipo de técnicas, además de eliminar estos prototipos erróneamente clasificados del conjunto de entrenamiento original, lleva a la agrupación de los prototipos en clusters pertenecientes a la misma clase. Así, este proceso servirá para que procesos posteriores de condensado puedan aprovecharse de unas fronteras de decisión sencillas, pudiendo eliminar una gran cantidad de prototipos que son considerados *poco importantes* para una posterior clasificación.

Los algoritmos de edición comentados a continuación son una muestra de los más utilizados a la hora de reducir el conjunto de prototipos original.

Edición de Wilson

Este algoritmo (Wilson, 1972) constituye el primer intento formal de reducir el conjunto de entrenamiento mediante la eliminación de prototipos erróneamente etiquetados. Utilizando la técnica de validación Leaving-One-Out, presentada en el Capítulo 2 de esta memoria, sobre el conjunto de entrenamiento, se eliminarán del mismo todos aquellos prototipos que resulten mal clasificados utilizando de la regla K-NN. Lo más habitual es aplicar la variante más sencilla de la misma, esto es, la regla 1-NN, aunque se puede elegir arbitrariamente el valor de K a utilizar en la aplicación del algoritmo.

El principal inconveniente que presenta dicho método de clasificación se refiere al alto coste computacional que conlleva su aplicación: $O(N^2)$ siendo N el tamaño del conjunto de entrenamiento.

Edición repetitiva

Basado en el anterior, aprovecha los agrupamientos más o menos compactos que proporciona el método de edición de Wilson, para proponer la aplicación repetitiva de dicho método sobre los diferentes conjuntos de prototipos resultantes (Tomek, 1976). De este modo pretende potenciar el efecto producido por la edición de Wilson sobre el conjunto de entrenamiento resultante.

La estrategia de repetir la edición de Wilson sobre los diferentes conjuntos de entrenamientos resultantes no produce una mejora significativa, ya que el número de prototipos eliminados tras la primera iteración resulta ser mínimo, aumentando además el costo computacional asociado al algoritmo de edición, dado el carácter repetitivo de la técnica.

Edición con reetiquetado

El criterio esbozado en este método para producir agrupamientos compactos y homogéneos del conjunto de prototipos se basa en reetiquetar determinados prototipos en función de la zona de representación en la que se encuentren. Así prototipos *rodeados* por otros pertenecientes a una clase distinta a la suya, serán reasignados a la clase mayoritaria de los prototipos que se encuentran a su alrededor. Esta estrategia de reasignar prototipos dudosos a otras clases no suele presentar buenos resultados en muchos casos reales de clasificación.

Edición con rechazo

Basándose en unos criterios muy similares a los considerados para la técnica anterior, se puede obtener un esquema alternativo en el que la regla K-NN con rechazo se utiliza no sólo en el proceso de edición, sino también en la clasificación de nuevos casos. Al aplicar este método, el conjunto de entrenamiento original de M clases se convierte en un conjunto editado con $M+1$ clases (1 de ellas es la de rechazo, normalmente la última), siendo esta nueva clase de rechazo tenida en cuenta en la aplicación posterior del proceso de clasificación.

Multiedit

Este método repite la edición con rechazo tal y como el método de edición repetitiva lo hace con la edición de Wilson. Por tanto se aplica repetidamente el proceso de edición por partición pero con K=1 (1-NN).

Los algoritmos que basan sus estimaciones en particiones del conjunto de entrenamiento tales como la edición por partición muestran habitualmente un comportamiento de menor potencia clasificatoria respecto a los otros algoritmos explicados de edición, caso de la edición de Wilson, empeorándose la situación si dado un conjunto de entrenamiento pequeño se aumenta además el número de bloques por partición efectuada. Es por ello que, en la práctica, los esquemas de edición basados en particiones se suelen utilizar para conjuntos de entrenamiento amplios.

5.4.3 Técnicas de condensado

Se presentan a continuación diversos procedimientos de entre los que se utilizan con el objetivo de reducir el número de casos que formarán el modelo de entre los del conjunto de entrenamiento. Se pretende que la reducción en el número de casos no afecte a la eficacia del clasificador, eliminando aquellos prototipos del conjunto de entrenamiento cuya presencia en éste no sea crucial a la hora de preservar la eficacia del clasificador.

Condensado de Hart

Constituye la primera propuesta formal de construir un método de condensado (Condensed Nearest Neighbour, Hart, 1968). Este método define lo que se llama consistencia respecto al conjunto de entrenamiento. Se dice que un conjunto de prototipos S es consistente con respecto a otro conjunto D, si al utilizar S como conjunto de entrenamiento, es posible clasificar los casos de D correctamente. Es deseable que un conjunto condensado sea un conjunto de prototipos reducido y consistente. En la Figura 5.6 se muestra el algoritmo correspondiente.

Condensado de Hart

COMIENZO
  Como entrada disponemos de:
    La Base de Datos de entrenamiento D, conteniendo N casos,
    $(\mathbf{x}_i, \theta_i), i = 1, ..., N$,

  Inicializar a vacío el subconjunto S de prototipos
  PARA cada caso de D, $(\mathbf{x}_i, \theta_i)$ HACER
    COMIENZO
      SI la clase que le asigna el algoritmo NN
      tomando por modelo los prototipos de S es correcta
      ENTONCES
        No añadir el caso actual a S
      SINO
        Incluir el caso actual en S
    FIN
  Como salida, se obtiene el subconjunto de prototipos seleccionados, S
FIN

Figura 5.6: Pseudocódigo del algoritmo *condensado de Hart*.

El funcionamiento utilizado por este método es bien sencillo: mantiene en el conjunto de entrenamiento aquellos prototipos que resulten mal clasificados a partir del conjunto de entrenamiento formado hasta ese mismo instante. Esto es así porque se supone que estos prototipos mal clasificados están próximos a las fronteras de decisión. Este algoritmo tiene un comportamiento computacional de orden lineal en la práctica, consiguiendo conjuntos de prototipos bastante reducidos con respecto al original, siempre que el conjunto de prototipos haya sido editado antes de serle aplicado el método. Con la aplicación de este método no se puede asegurar que se obtiene siempre el conjunto minimal consistente con respecto al original, si es que éste existe.

Condensado reducido

Con este método, (Gates, 1972), se pretende eliminar del conjunto consistente obtenido a partir del condensado de Hart, aquellos prototipos que no resulten imprescindibles para preservar la mencionada propiedad de consistencia. Esto se realiza aplicando sucesivamente el condensado de Hart sobre los diferentes subconjuntos de prototipos que se van obteniendo en cada una de las ejecuciones del mismo, hasta que el algoritmo ya no elimina ningún prototipo del subconjunto que recibe como entrada. En la Figura 5.7 se puede ver el pseudocódigo correspondiente a este método.

Condensado reducido

```
COMIENZO
  Como entrada disponemos de:
    La Base de Datos de entrenamiento D, conteniendo N casos,
    (x_i, θ_i), i = 1,...,N,

  Inicializar a vacío el subconjunto S de prototipos
  Inicializar el subconjunto auxiliar T a X
  MIENTRAS T sea diferente de S HACER
    COMIENZO
    Obtener el subconjunto S de prototipos mediante el condensado de Hart sobre T
    Aux = S
    PARA cada caso de S, (x_i, θ_i) HACER
        COMIENZO
            SI la clase que asigna el algoritmo NN a todos los casos de T
                  tomando por modelo los prototipos de S \ x_i es correcta
            ENTONCES
                  Eliminar el caso actual de S
            SINO
                  Mantener el caso actual en S
        FIN
        T = Aux
    FIN
  Como salida, se obtiene el subconjunto de prototipos seleccionados, S
FIN
```

Figura 5.7: Pseudocódigo del algoritmo *condensado reducido*.

En la práctica este algoritmo tampoco asegura la consecución del algoritmo minimal consistente, obteniendo, eso sí, conjuntos de prototipos de tamaño menor o igual que el de Hart. La efectividad

conseguida por el clasificador utilizando los prototipos aquí obtenidos se halla muy pareja a la que se consigue utilizando los prototipos que selecciona el condensado de Hart.

Nearest Neighbour selectivo

En esta aproximación (Rittet y col., 1975), se extiende heurísticamente el concepto de subconjunto consistente minimal, añadiendo el requerimiento de que cada instancia de la Base de Datos de entrenamiento tenga como vecino más cercano de entre los prototipos seleccionados uno de su misma clase. Este requerimiento adicional da lugar a una relación que asocia a cada instancia con aquellas de su misma clase que se encuentran más cercanas entre sí que respecto a instancias de distinta clase. El subconjunto de prototipos que se obtiene utilizando este algoritmo *Selective Nearest Neighbour*, es el subconjunto más pequeño que contiene al menos una instancia de cada una de las relaciones de este tipo que aparecen en el conjunto de entrenamiento.

IB2 e IB3

El algoritmo IB2 (Aha, 1990) es bastante parecido al *condensado reducido*. Para predicciones sobre valores finitos y numerados, IB2 guarda sólo los casos mal clasificados utilizando el algoritmo NN. Un segundo algoritmo, IB3, (Aha, 1990), utiliza únicamente instancias *aceptables* como prototipos, esto es, se seleccionan previamente instancias no ruidosas de la base de datos, y luego se reduce el conjunto resultante. Se consideran aceptables, provisionalmente, aquellas instancias que son bien clasificadas mediante la aplicación del algoritmo IB2. Posteriormente sólo un subconjunto de éstas será utilizado en el proceso de clasificación. Este subconjunto se obtiene seleccionando de entre las instancias aceptables, aquellas cuyo poder clasificatorio es significativamente mayor entre sus instancias cercanas que sobre todo el conjunto de entrenamiento. Para determinar la significación estadística, se mantiene un registro de los porcentajes de clasificación obtenidos por las instancias aceptadas, para aplicar posteriormente un test. IB1 es la implementación realizada por Aha del algoritmo NN o 1-NN.

Búsqueda estocástica

Entre los métodos de este tipo, hay algunos simples de selección de prototipos y variables utilizando *algoritmos genéticos*. Cameron-Jones (1995) ofrece una aproximación basada en un heurístico consistente en minimizar la longitud del string que representa un conjunto de instancias bien y mal clasificadas.

Además de las técnicas de selección de prototipos convencionales de edición y condensación del conjunto de entrenamiento expuestas anteriormente, se ha optado por analizar y elaborar

una alternativa a estas técnicas, basada en la aplicación de algoritmos genéticos, tratando de conseguir una reducción de la talla del conjunto de entrenamiento, pero procurando al mismo tiempo preservar la efectividad del clasificador. La forma de llevar a cabo todo el proceso de selección de prototipos mediante algoritmos genéticos merecerá un punto aparte en esta memoria, donde se explicará con detalle.

Capítulo 6

Aportaciones realizadas a la familia de algoritmos basados en vecindad

6.1 Introducción

Se presentan en esta sección las aportaciones realizadas en el campo de los algoritmos basados en vecindad. Aparecen, por un lado, tres nuevos algoritmos de la familia propuestos en esta memoria de tesis, (K Diplomatic Nearest Neighbour (K-DNN) y Probabilistic Weighted K Nearest Neighbour (PW-K-NN), y K-NN Clases), por otro, dos nuevas técnicas de pesado de atributos, (K-NN-UN y K-NN-LO), mientras que una tercera aportación se basa en la utilización de algoritmos evolutivos para la selección de prototipos.

El algoritmo K-DNN es una extensión del clásico K-NN, en la cual se buscan, para cada caso a ser clasificado, sus K vecinos mas próximos *en cada una de las clases existentes*, asignando al nuevo caso a clasificar la clase cuya media de distancias de sus K elementos mas próximos sea mínima. El nombre quiere indicar que se realiza de una forma diplomática, esto es, sin tener en cuenta el tamaño de cada una de las clases.

El segundo algoritmo novedoso propuesto para este paradigma, PW-K-NN, asocia a cada elemento una medida de su *tipicidad*, asignándole la probabilidad que tiene de pertenecer a la clase a la que efectivamente pertenece. Para calcular estas probabilidades se utiliza una red Bayesiana inducida sobre todos los casos pertenecientes a la Base de Datos de entrenamiento, con lo que se podría considerar este algoritmo como un híbrido entre los clasificadores basados en la distancia mínima y los basados en redes Bayesianas. A la hora de clasificar un nuevo caso, se hallan sus K vecinos más próximos utilizando el algoritmo K-NN, pero se asocia a cada uno de los

K vecinos un peso en el voto equivalente al asignado por la red Bayesiana como la probabilidad de pertenecer a su clase. De este modo evitamos que casos que son excepciones en una clase influyan del mismo modo que los casos típicos en la clasificación de un nuevo caso.

El tercer algoritmo, K-NN Clases, tiene en cuenta el porcentaje de casos de cada clase que se encuentran entre los K vecinos de un caso dado a la hora de realizar la clasificación. Para ello se realiza un cálculo previo de los vecinos que tiene cada uno de los casos de la base de datos de entrenamiento.

Respecto a los métodos de ponderación de variables propuestos, están basados en lo que se conoce en inglés como técnicas *wrapper*, que se podrían traducir como técnicas envolventes, en las que se aplica el mismo algoritmo clasificatorio para medir la bondad del modelo inducido por él. En este caso se utiliza el algoritmo K-NN para medir el peso que se asociará a cada una de las variables del modelo. Si suponemos que disponemos de una Base de Datos con n variables predictoras, dos formas de hacerlo son:

1. Para cada variable predictora X_i, servirse del resto de variables $X_1, X_2, ..., X_{i-1}, X_{i+1}, ..., X_n$ para inducir un modelo clasificador, y utilizar el porcentaje de casos bien clasificados obtenido como una medida de la pérdida de poder clasificatorio del modelo al dejar de usar la variable X_i en el proceso de clasificación. Sea P_i el tanto por uno de bien clasificados que se obtiene al utilizar todas las variables excepto la i-ésima; entonces el peso que se aplicará a dicha variable en el cálculo de las distancias será

$$Peso_i = 1 - \frac{P_i}{\sum_{j=1}^{n} P_j}.$$

   A este método se le ha denominado K-NN-UN en esta memoria.

2. Utilizar únicamente la variable X_i para clasificar los casos de la Base de Datos (mediante la técnica de validación *Leave One Out*), y considerar el porcentaje de casos bien clasificados obtenido como una medida del peso a asociar a dicha variable. En este caso, y considerando de nuevo que P_i es el tanto por uno de bien clasificados que se obtiene al utilizar únicamente la variable i-ésima en el proceso de clasificación, el peso que asociamos a dicha variable es

$$Peso_i = \frac{P_i}{\sum_{j=1}^{n} P_j}$$

   A este método se le ha denominado K-NN-LO en esta memoria.

La forma en que se utilizan los algoritmos evolutivos para la selección de prototipos es explicada en el punto 6.4 de esta memoria.

6.2 Aportaciones basadas en el cálculo de la distancia

Se presentan en esta sección las aportaciones que en el ámbito del cálculo de las distancias se han propuesto en el desarrollo de esta tesis. Se pueden considerar los siguientes algoritmos como variantes o extensiones novedosas del algoritmo K-NN.

6.2.1 Algoritmo K *Diplomatic Nearest Neighbour*

En esta sección presentamos una modificación del método K-NN que halla los K vecinos más cercanos en todas y cada una de las clases tenidas en cuenta en el problema en cuestión, y asigna al caso a clasificar aquella clase cuyos K vecinos más próximos tienen la menor distancia media respecto al caso a clasificar (Sierra y col., 1999a). La idea está basada en la sospecha de que los valores de las variables predictoras (componentes de $\mathbf{x}$) pudieran ser diferentes en cada una de las clases, al menos en lo que a su distribución se refiere.

El método K *Diplomatic Nearest Neighbour*

Cuando se debe de afrontar un problema que presenta incertidumbres mediante la aplicación de algún método proveniente de la clasificación supervisada, se tiene la sospecha de que existe alguna relación entre los componentes de la observación $\mathbf{x}$, denominadas variables predictoras, y la clase θ a la que corresponde el caso a clasificar, esto es, se supone que las variables predictoras, como su nombre parece indicar, dan cierta información sobre la clase real del caso a clasificar. En otras palabras, si la observación $\mathbf{x}$ se halla compuesta por n variables predictoras, $X_1, X_2, ..., X_n$, se puede asumir que exista cierta dependencia entre estas n variables y la clase real θ.

En otros paradigmas clasificatorios (árboles de clasificación, inducción de reglas,...) se realiza una progresiva selección de variables predictoras hasta que la decisión sobre la clase de pertenencia del caso a tratar es llevada a cabo. El espacio de búsqueda se divide teniendo en cuenta los valores de algunas de las variables predictoras, y de éste modo se presume que algunos de éstos valores aparecen con mayor frecuencia para una clase θ, esto es, los valores de las variables podrían tener diferente distribución de probabilidad para cada clase tenida en consideración.

Teniendo en cuenta que se espera que la distribución de los valores de las variables predictoras sea específico para cada clase, se propone una modificación sobre el algoritmo K-NN en el sentido señalado, que tomará en cuenta de forma un tanto distinta los valores de las variables a la hora de asignar una clase al nuevo caso $\mathbf{x}$ a clasificar.

El nuevo método propuesto, K *Diplomatic Nearest Neighbour* (K-DNN), es mostrado en forma algorítmica en la Figura 6.1. Es una extensión simple del algoritmo K-NN que ha obtenido buenos

resultados en los experimentos realizados –véase el Capítulo 9–. A pesar de que el costo computacional es elevado ($O(K \cdot M \cdot (N/M)^2)$, siendo M el número de clases consideradas, este método ha obtenido resultados que igualan o superan a los algoritmos estándares de clasificación supervisada (árboles de clasificación, inducción de reglas, aprendizaje basado en casos,...) implementados en la librería MLC++ (Kohavi y col., 1997).

K Diplomatic Nearest Neighbour

COMIENZO K-DNN
  Como entrada disponemos de:
    la Base de Datos de ejemplo, conteniendo N casos
  $(\mathbf{x}_i, \theta_i), i = 1, ..., N$,
  el valor del parámetro K
  y un nuevo caso $(\mathbf{x}, \theta)$ que se desea clasificar
  PARA cada clase C HACER
    COMIENZO
      Seleccionar los K vecinos más cercanos a $\mathbf{x}$,
      de entre los que pertenecen a la clase C en la Base de Datos
      Calcular la distancia media a $\mathbf{x}$ de estos K casos D_C
    FIN
  Como salida, dar la clase c_l^* cuya distancia media D_l^* es la mínima de entre todas las clases
FIN K-DNN

Figura 6.1: Pseudocódigo del algoritmo K *Diplomatic Nearest Neighbour*.

Del pseudocódigo del algoritmo mostrado en la Figura 6.1, se deduce que éste procede de la siguiente manera:
Dado un conjunto de N casos correctamente clasificados, $(\mathbf{x}_1, \theta_1), (\mathbf{x}_2, \theta_2), ..., (\mathbf{x}_N, \theta_N)$, en un problema en el que se contemplan M categorías o clases diferentes, siendo $1, 2, ..., M$ los códigos respectivos de cada una de estas clases, y dado un nuevo caso $(\mathbf{x}, \theta)$, del cual queremos establecer a que clase pertenece, esto es, el valor de θ es desconocido, e indicando al algoritmo el número K de vecinos que deben ser tenidos en cuenta para realizar la clasificación, el algoritmo lleva a cabo el siguiente proceso clasificatorio:

Para cada clase c, $c \in \{1, 2, ..., M\}$, hallar los K casos más cercanos a $\mathbf{x}$ de entre los pertenecientes a dicha clase c. Sean $\mathbf{x}_{c1}, \mathbf{x}_{c2}, ..., \mathbf{x}_{cK}$ dichos K casos, con distancias respectivas al punto $\mathbf{x}$, $d_{c1}, d_{c2}, ..., d_{cK}$. Calcular la media de estas K distancias D_C aplicando la fórmula siguiente:

$$D_C = \sum_{i=1}^{K} \frac{d_{Ci}}{K}$$

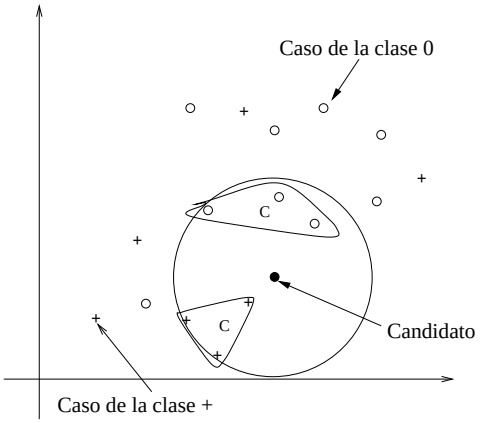


Figura 6.2a

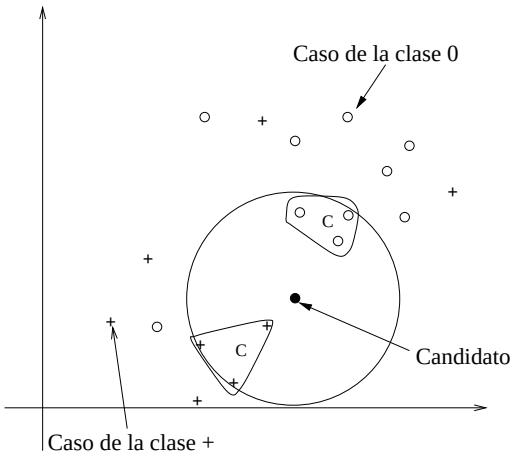


Figura 6.2b

Figura 6.2: La regla de decisión 3-DNN comparada con 6-NN. En la Figura 6.2a el resultado, tanto de 3-DNN como de 6-NN es un empate, mientras que en la Figura 6.2b el resultado que produce la aplicación de la regla 3-DNN es la clase +, obteniéndose un empate mediante la aplicación de 6-NN.

Asignar a $\mathbf{x}$ como clasificación aquella clase c_l^* para la cual la distancia media D_l^* de sus K casos más cercanos a $\mathbf{x}$ es la mínima entre todas las clases, esto es,

$$\theta = C_l^* / D_{C_l^*} = min(D_{C_j}) j = 1, ..., M.$$

Del mismo modo que sucede con el algoritmo K-NN, es necesario establecer algún método de resolución de empates, el cual puede ser, por ejemplo, elegir la clase más frecuente de entre las que se hallan empatadas, la que tenga un caso a menor distancia, etc.

En la Figura 6.2 se muestra un ejemplo del comportamiento del algoritmo 3-DNN en comparación con la regla de clasificación 6-NN en un problema con dos clases. Aunque el costo computacional del nuevo algoritmo es superior, su comportamiento clasificatorio puede tratar de forma más clara la información discriminatoria que se deduce de las variables predictoras. Esto se aprecia en mejor medida en problemas con más de dos clases.

6.2.2 Algoritmo *Probabilistic-Weighted K Nearest Neighbour*

En esta sección se presenta un nuevo algoritmo propuesto durante el desarrollo de esta tesis que tiene en cuenta el hecho, habitual en las Bases de Datos de casos con las que se suele trabajar, de

que no todas las instancias que en ellas aparecen son típicas de la clase a la que pertenecen, sino que algunas de ellas constituyen excepciones. Se trata de establecer una medida de la excepcionalidad de los casos para que, de esta forma, podamos ponderarlos a la hora de la votación de modo que un caso típico de una clase influya en mayor medida que uno poco típico. En esta aproximación, a cada caso de la Base de Datos se le asocia una medida de su tipicidad, utilizando para ello una red Bayesiana inducida sobre todos los casos de la misma, que, siendo instanciada con los valores de las variables de un caso, y tras realizar la propagación, nos da una medida de la probabilidad de ése caso de pertenecer a la clase a la que realmente pertenece.

El modelo de red Bayesiana utilizado puede variar. En principio parece lógico utilizar un modelo inducido sobre los datos con objetivo clasificatorio, pero esto no tiene necesariamente que ser así. Se puede utilizar el clasificador más simple, naive Bayes, o alguno más complicado, como el *Markov blanket* de la variable a clasificar, o se pueden utilizar redes Bayesianas inducidas mediante la utilización de otro tipo de métrica, como puede ser K2, BIC, etc.

El método *Probabilistic-Weighted K Nearest Neigbour*

Teniendo en cuenta que la distribución de las variables predictoras se asume específica para cada clase, se propone una modificación del algoritmo K-NN que tenga, de alguna manera, esto en consideración, obteniendo mayor poder clasificatorio al poder identificar la probabilidad de cada caso de pertenecer a cada una de las clases. Se obtiene en cierto modo una medida de la excepcionalidad de cada caso.

Un ejemplo de aplicación es mostrado en la Figura 6.3. Este nuevo método que se propone se presenta de forma algorítmica en la Figura 6.4.

Tal y como se desprende del algoritmo presentado en la Figura 6.4, el método PW-K-NN se aplica de la siguiente manera: dado un conjunto de N casos correctamente clasificados, $(\mathbf{x}_1, \theta_1)$, $(\mathbf{x}_2, \theta_2)$, ..., $(\mathbf{x}_N, \theta_N)$, en un problema de clasificación en el que se han tenido en consideración M clases diferentes, que han sido codificadas de la forma $1, 2, ..., M$, y dados un nuevo caso a clasificar, $(\mathbf{x}, \theta)$, en el cual la clase θ es desconocida, y un número de vecinos a tener en consideración, K, se realiza el siguiente proceso de clasificación:

En primer lugar, se debe inducir una red Bayesiana a partir de los N casos o instancias que se encuentran en la base de datos de entrenamiento. Para ello podemos utiliza la técnica explicada en el Capítulo 4 de esta memoria que obtiene el *Markov blanquet* de la variable a clasificar, utilizando los algoritmos genéticos como heurístico de búsqueda y el porcentaje de bien clasificados como medida de la bondad de un individuo dado del algoritmo, que será una estructura válida de red Bayesiana en la que los nodos forman el *Markov blanket* de la variable que representa a la clase.

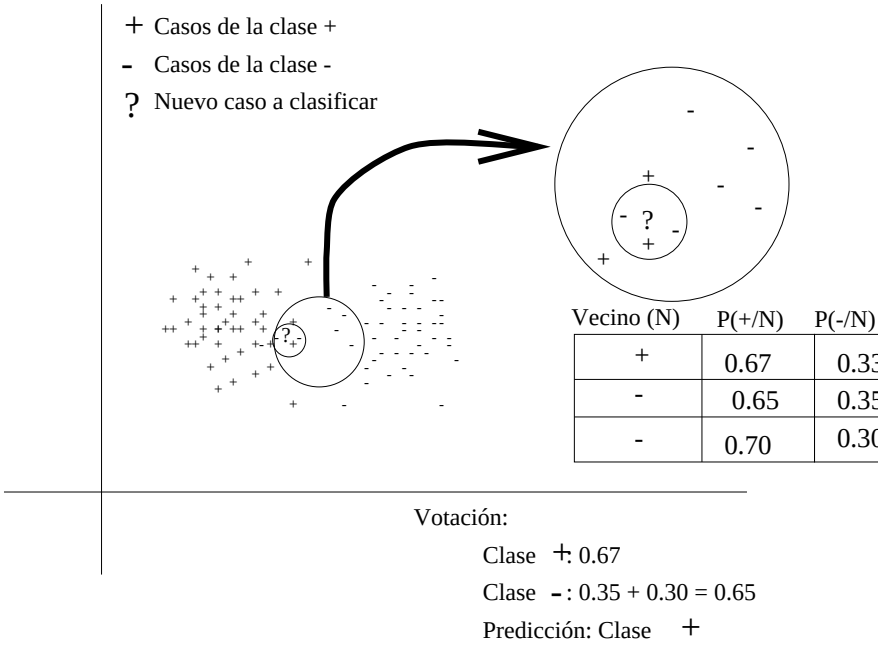


Figura 6.3: Un ejemplo de la aplicación de la regla de decisión *Probabilistic-Weighted 3 Nearest Neighbour*.

Una vez aprendida la red Bayesiana, se utiliza para asignar a cada uno de los N casos correctamente clasificados de que se dispone, la probabilidad a posteriori de pertenecer a la clase a la que realmente pertenecen, $(\mathbf{x}_i, \theta_i)$ y $W_i = P(\theta_i \mid \mathbf{x}_i) \Rightarrow (\mathbf{x}_i, \theta_i, W_i)$. A continuación, dado el nuevo caso que se desea clasificar, se hallan sus K vecinos más próximos, $(\mathbf{x}_{N1}, \theta_{N1}, W_{N1}), \ldots, (\mathbf{x}_{NK}, \theta_{NK}, W_{NK})$, y se calcula el peso del voto asociado a cada clase utilizando los pesos asociados a las clases que se han calculado mediante la utilización de la red Bayesiana en el paso previo. Predecir como clasificación del nuevo caso aquella clase θ que ha obtenido un mayor peso al finalizar el proceso de votación.

De nuevo es necesario establecer algún método de resolución de empates, aunque dadas las características de la aproximación propuesta, es poco probable que se produzcan.

6.2.3 El método K *Nearest Neighbour Clases*

Esta es una variante del K-NN que amplía el concepto de vecindad entre las instancias de la base de datos, aprovechando la información proporcionada por todos y cada uno de los K vecinos a un caso dado. Antes de llevar a cabo el proceso de clasificación de nuevos casos, requiere una fase de preproceso en el conjunto de entrenamiento, en la que, a partir de las instancias existentes en el

Probabilistic-Weighted K Nearest Neighbour

Comienzo PW-K-NN
Como entrada recibe el fichero de casos de entrenamiento TR,
conteniendo N casos $(\mathbf{x}_i, \theta_i), i = 1, ..., N$,
El valor del parámetro K,
y un nuevo caso $(\mathbf{x}, \theta)$ al que se desea predecir la clase de pertenencia
PARA cada caso perteneciente a TR HACER
COMIENZO
Sea $(\mathbf{x}_i, \theta_i)$ el caso actual
Estimar $W_i = P(\theta_i \mid \mathbf{x}_i)$ a partir de una red Bayesiana en $TR \setminus \{(\mathbf{x}_i, \theta_i)\}$
Añadir este peso estimado a la información del caso en TR, $(\mathbf{x}_i, \theta_i, W_i)$
FIN
Buscar los K vecinos más próximos de $(\mathbf{y}, \theta)$ en TR
Inicializar a cero los pesos de todas las clases $WC_i = 0$
PARA cada uno de los K vecinos más próximos HACER
COMIENZO
Sea $(\mathbf{x}_i, \theta_i, W_i)$ el vecino actual
Actualizar el peso de la clase c_i como se indica a continuación:
$WC_i \leftarrow WC_i + W_i$
FIN
Como salida, dar la clase c^* cuyo peso Wc^* es el máximo de entre todas las clases
Fin PW-K-NN

Figura 6.4: El pseudocódigo del algoritmo *Probabilistic-Weighted* K *Nearest Neighbour*.

mismo, se realiza un proceso de reclasificación, guardando los porcentajes de pertenencia a cada clase de los prototipos del conjunto de entrenamiento a partir de la aplicación de la técnica de validación Leaving-One-Out sobre todos los casos del conjunto de entrenamiento. A cada caso de la base de datos de entrenamiento se le asigna el tanto por ciento de las clases de los distintos vecinos encontrados en el área de vecindad definido para el problema a través del valor del parámetro K.

Ejemplo:

- Suponiendo un valor 5 para el número de vecinos a considerar, (K=5), y un total de 4 clases (c1, c2, c3, c4).
- Elegir un caso cualquiera del conjunto de entrenamiento para ser clasificado a partir de los restantes $N-1$ casos. (Leaving-One-Out)
- Si resulta que la distribución de los K vecinos (por clases) al caso a clasificar resulta ser:

$c1$	$c2$	$c3$	$c4$
3	1	0	1

- La nueva clasificación del caso será:

$c1$	$c2$	$c3$	$c4$
60%	20%	0%	20%

Tras redefinir la clasificación de cada instancia del conjunto de entrenamiento, se podrá pasar inmediatamente a operar con el conjunto de entrenamiento resultante. A la hora de clasificar las instancias del conjunto de testeo se tendrán en cuenta los porcentajes de pertenencia a clases de los K casos más cercanos del conjunto de entrenamiento. Así, un nuevo caso se asignará a la clase que acumule un porcentaje más elevado de vecinos a partir de la suma de los tantos por ciento aportados por los K vecinos más próximos al caso a clasificar para las diferentes clases del problema.

Ejemplo:

- Se clasifica un caso **x** del conjunto de testeo a partir del conjunto de entrenamiento resultado de aplicar el preproceso de cálculo de porcentajes de pertenencia a las clases a las instancias del conjunto de entrenamiento.

  En la Tabla 6.1 se pueden observar los datos de los K vecinos más próximos a **x**.

Tabla 6.1: K vecinos más próximos a **x** del conjunto de entrenamiento.

Caso	*c1*	*c2*	*c3*	*c4*
1	40%	20%	40%	0%
2	80%	0%	20%	0%
3	20%	40%	20%	20%
4	60%	40%	0%	0%
5	40%	60%	0%	0%

- El nuevo caso será clasificado como perteneciente a la clase que acumula un porcentaje mayor de las clases del problema. Así el caso **x** se clasificará en la clase 1 ya que es la que acumula un tanto por ciento de votos mayor de todas las clases.

Mediante la ampliación del concepto de vecindad, se pretende recoger en los prototipos del conjunto de entrenamiento toda la información asociada a su clasificación a partir de los demás componentes del conjunto de entrenamiento. Los K prototipos más próximos utilizados para la asignación de una clase a la nueva instancia, al no contener una única clase de pertenencia sino el tanto por ciento de pertenencia a las diferentes clases del problema, puede ayudar a clasificar correctamente nuevos casos próximos a las fronteras de decisión.

La información adicional utilizada en el problema pretende recoger más fielmente la situación de cada prototipo del conjunto de entrenamiento en el espacio de representación del problema, información que puede servir de gran ayuda sobre todo a la hora de clasificar instancias próximas a las fronteras de decisión entre clases del problema, ya que los prototipos de entrenamiento no solamente aportan información sobre su clasificación particular, sino que la información aportada por cada prototipo se refiere más bien a su situación particular en el espacio de representación del problema. La información sobre la situación particular de cada prototipo del conjunto de entrenamiento del problema, será utilizada por el clasificador para asignar la clase más probable de pertenencia de la nueva instancia a clasificar a partir de su estatus propio en el espacio de representación del problema.

La parte negativa asociada a la utilización de este nuevo método que amplía el concepto de vecindad para la clasificación de nuevos casos, reside en la carga computacional que añade el cálculo de estos porcentajes de pertenencias a clases para los diferentes prototipos del conjunto de entrenamiento, además del espacio adicional requerido para la representación de esta información adicional. Para la aplicación del K-NN tradicional sólo es necesario guardar la clase para cada prototipo; en cambio con la ampliación del concepto de vecindad, se necesitará guardar todos los porcentajes de pertenencia a todas las clases del problema para cada prototipo del conjunto de entrenamiento.

6.3 Aportaciones sobre el pesado de atributos

Se presentan en esta sección cuatro nuevos métodos de pesados de atributos propuestos, basados, dos de ellos, en lo que se conoce como técnicas envolventes o *wrapper* (faja). En los mismos, la idea básica consiste en utilizar el mismo algoritmo para el pesado de atributos que el que posteriormente se utilizará en la clasificación de los nuevos casos. Los otros dos utilizan otro algoritmo clasificatorio, en este caso el naive-Bayes, para calcular el peso asociado a cada variable. Estas técnicas aquí presentadas pueden ser aplicadas en otro tipo de paradigmas de la clasificación supervisada.

La idea se basa en medir la importancia de las variables en el proceso de clasificación. Se pretende obtener una idea del poder clasificatorio de cada variable, para luego utilizarlo en el pesado de los atributos.

Si aplicamos un peso a los atributos, en el algoritmo K-NN se produce un cambio en el cálculo de las distancias. Sean $X_1, ..., X_n$ las variables predictoras del problema que nos ocupa, sean $W_1, ..., W_n$ los pesos respectivos de cada una de éstas n variables, y sean dos casos $\mathbf{y}$ y $\mathbf{z}$, con componentes $y_1, ..., y_n$ y $z_1, ..., z_n$ respectivamente; la distancia entre ambos se calcula mediante la aplicación de la siguiente formula:

$$D_{YZ} = \sum_{i=1}^{n} W_i (y_i - z_i)^2.$$

Existen diferentes formas de calcular los pesos W_i (Wettschereck y col., 1997). Proponemos a continuación dos nuevas aproximaciones.

6.3.1 El método de pesado K-NN-UN

Este método (Sierra y col., 1999a) mide la influencia que el hecho de quitar la variable i-ésima del modelo tiene sobre el porcentaje de bien clasificados que se obtiene. De nuevo, consideremos que $X_1, ...X_n$ son las variables predictoras del modelo; para cada variable, X_i, realizamos el siguiente proceso: construimos una base de datos de entrenamiento que contiene todas las variables, excepto la propia X_i. Medimos a continuación el porcentaje de bien clasificados que se obtiene con esta base de datos y el método naive-Bayes de clasificación. De este modo, obtenemos un valor P_i que indica el porcentaje de éxito clasificatorio obtenido mediante la utilización en el modelo de todas las variables excepto X_i.

A continuación, calculamos el peso asociado a cada una de las variables predictoras de la siguiente manera:

$$W_i = 1 - \frac{P_i}{\sum_{j=1}^{n} P_j}; i = 1, ..., n, 0 \leq W_i \leq 1.$$

El método de pesado K-NN-UN1

Este es una variante del método anterior, en la que se utiliza el propio algoritmo K-NN (más concretamente, el 1-NN), en lugar del naive-Bayes, para obtener el porcentaje de bien clasificados en los subconjuntos de entrenamiento que se obtienen, aplicando la técnica de validación Leave-One-Out.

6.3.2 El método de pesado K-NN-LO

Este método mide el poder clasificatorio de cada una de las variables X_i utilizándolas una a una para construir un modelo clasificatorio, y midiendo el porcentaje de bien clasificados que obtiene cada una de ellas mediante el algoritmo de clasificación naive-Bayes. Sean $X_1, ...X_n$ las variables predictoras del modelo. Para cada variable, X_i, realizamos el siguiente proceso: construimos una base de datos de entrenamiento que contiene únicamente la propia variable X_i y la asociada a la clase C. Medimos a continuación el porcentaje de bien clasificados que se obtiene con esta base de datos y el método naive-Bayes de clasificación. De este modo, obtenemos un valor P_i que indica el porcentaje de éxito clasificatorio obtenido mediante la utilización en el modelo asociado a la variable X_i.

A continuación, calculamos el peso asociado a cada una de las variables predictoras de la siguiente manera:

$$W_i = \frac{P_i}{\sum_{j=1}^{n} P_j}; i = 1, ..., n, 0 \leq W_i \leq 1.$$

El método de pesado K-NN-LO1

Este método es una variante del anterior que utiliza el algoritmo 1-NN, mediante la técnica de validación Leave-One-Out, para medir el poder clasificatorio de cada una de las variables.

6.4 Aportaciones sobre selección de prototipos

Tanto la selección de prototipos como el pesado de atributos son problemas que tienen una complejidad computacional muy elevada, lo que los hace inabordables mediante un algoritmo exhaustivo de búsqueda. Es por ello que las aproximaciones que se utilizan para abordar este problema están basadas en heurísticos. Como ya se ha indicado en el Capítulo 2 de esta memoria, en problemas de este tipo se han venido aplicando con éxito técnicas de computación evolutiva, como pueden

ser los *algoritmos genéticos* (AAGG), (Holland, 1975), o los algoritmos que se han desarrollado más recientemente, *estimation of distribution algorithm*, (EDA), algoritmos de estimación de distribuciones de probabilidad (Mühlenbein y Paaß, 1996).

En el desarrollo de esta tesis se han aplicado versiones de ambas aproximaciones para la selección de un subconjunto de prototipos del conjunto de entrenamiento. Es de destacar, así mismo, que al tratar conjuntos de entrenamientos grandes, la complejidad del algoritmo basado en la computación evolutiva es excesiva, con lo que se ha optado por dividir previamente el conjunto de entrenamiento inicial utilizando técnicas de clasificación no supervisada, en concreto, el algoritmo K-Means (MacQueen, 1967). Se presentan a continuación las aproximaciones utilizadas.

6.4.1 Algoritmos genéticos para selección de prototipos

La idea de aplicar algún tipo de algoritmo genético para la selección de prototipos en el conjunto de entrenamiento, se plantea como alternativa a los tradicionales métodos de edición y condensación utilizados para tal fin.

En términos generales, se puede decir que la edición provoca la eliminación de prototipos erróneamente etiquetados o prototipos *outliers* del conjunto de entrenamiento, mientras que la condensación selecciona sólo aquellos prototipos suficientemente representativos del total, provocando una reducción del conjunto de entrenamiento sin que por ello se vea afectada la eficacia del clasificador.

La aplicación de algoritmos genéticos sobre el conjunto de entrenamiento debe por tanto aunar las ventajas de la edición y la condensación. Viendo los dos métodos tradicionales de selección de prototipos como si de un único proceso se tratara, se puede comprobar que básicamente lo que pretenden es reducir todo lo posible la talla del conjunto de entrenamiento y mantener las tasas de acierto que se obtendrían con el conjunto de entrenamiento original. Por tanto, el algoritmo genético que se construya deberá cumplir dos objetivos fundamentales para que sirva como alternativa a los procesos de selección de prototipos tradicionales: reducción significativa de la talla del conjunto de entrenamiento y obtención de unas tasas de acierto elevadas en el proceso de clasificación.

Diseño del algoritmo genético

A continuación se expondrán los pasos seguidos para la construcción del algoritmo genético para la selección de prototipos del conjunto de entrenamiento. En concreto, se va a comentar como se han realizado cada uno de los siguientes apartados: codificación de los individuos, construcción de la función objetivo, operadores de selección, cruce y mutación, reducción de las poblaciones, y por último veremos como queda la estructura general del algoritmo.

Codificación

Para la codificación de los individuos se ha utilizado un alfabeto binario {0,1}. La longitud de las ristras binarias que representan a un individuo de la población es N, siendo N el número de prototipos del conjunto de entrenamiento. Cada gen de los individuos corresponde a un prototipo determinado del conjunto de entrenamiento; si el gen que representa a un prototipo está a"1" en un individuo, significa que el prototipo correspondiente se ha elegido para formar parte del conjunto de entrenamiento tras la selección (en la solución representada por el individuo), análogamente, si un gen se encuentra a "0", el prototipo correspondiente no forma parte del conjunto de prototipos elegidos tras la selección.

Cabe señalar que los prototipos no elegidos del conjunto de entrenamiento servirán en cada momento para estimar la bondad de la solución que representa un individuo, sirviendo estos prototipos como conjunto de testeo para los prototipos elegidos, como se verá más adelante.

Ejemplo: Suponiendo una correspondencia Gen j $\rightarrow$ Prototipo j en los Individuos, y teniendo un conjunto de entrenamiento en el que hay 10 instancias, $\{P_1, P_2, P_3, P_4, P_5, P_6, P_7, P_8, P_9, P_{10}\}$, el siguiente individuo

Individuo i= | 0 1 1 0 0 1 1 0 1 0 |

representa una solución en la que los prototipos seleccionados son $\{P_2, P_3, P_6, P_7, P_9\}$.

Construcción de la función objetivo

Como se ha comentado en el apartado dedicado a la introducción de los algoritmos genéticos, la función objetivo establece la bondad de la solución representada por un individuo al problema combinatorio que se está tratando. En el caso que nos ocupa, la selección de prototipos del conjunto de entrenamiento, una solución será considerada como buena si, además de eliminar un gran número de prototipos del conjunto original mantiene los niveles de acierto conseguidos con éste. En la definición de la función objetivo para este problema habrá que centrar el diseño en dos cuestiones fundamentales: minimizar la talla del conjunto de prototipos y conseguir una tasa de aciertos elevada a partir del nuevo.

Para la primera de las premisas a tener en cuenta en la construcción de la función objetivo, es necesario saber el número de prototipos seleccionados en los diferentes individuos/soluciones. A partir de la codificación elaborada para este problema, la talla del conjunto de entrenamiento se obtiene en base al número de genes que se encuentran a "1" en el individuo. Este dato será el que se tenga en cuenta en la función objetivo para cubrir el primero de los dos aspectos fundamentales establecidos para la definición de la misma.

La segunda premisa a considerar en la función objetivo, se concentra en los aciertos obtenidos a partir de una solución dada, representada por un individuo. Una forma de saber el número de aciertos cosechados por un individuo/solución del algoritmo genético será contabilizar el porcentaje de aciertos que se obtienen utilizando los genes que se encuentran a "1" como conjunto de entrenamiento y los genes que se encuentran a "0" como conjunto de testeo. Este dato será tenido en cuenta a la hora de evaluar la bondad de un individuo a través de la función objetivo. Cabe señalar también que la contabilización del porcentaje de aciertos cada vez y para cada individuo resulta muy costosa en cuanto a carga computacional. Al mismo tiempo es obvio que se repetirá el cálculo de un gran número de distancias entre prototipos iguales a lo largo de la ejecución del algoritmo genético. Para paliar el excesivo coste computacional que conlleva el cálculo repetido de distancias entre prototipos iguales, se creará una matriz de orden $O(N(N-1)/2)$ que almacenará en una fase de preproceso las distancias entre todos los prototipos del conjunto de entrenamiento. Esto conllevará en su vertiente negativa un gran coste espacial, limitando su utilización en conjuntos de entrenamientos grandes. En cualquier caso, es preferible este coste espacial al coste computacional asociado sin la utilización de esta matriz, que hace impracticable la utilización de una función objetivo que tenga en cuenta los aciertos de los diferentes individuos/soluciones y por lo tanto la utilización de algoritmos genéticos para la resolución del problema de la selección de prototipos del conjunto de entrenamiento.

Señalar también que se ha dado prioridad al porcentaje de aciertos obtenido sobre el porcentaje de prototipos seleccionados en cada individuo, para la construcción de la función objetivo, ya que se considera más importante la obtención de un buen porcentaje de aciertos a costa de un porcentaje de individuos seleccionados algo más elevado.

Habida cuenta todo lo anterior, la función objetivo tendrá la siguiente forma para el individuo i:

$$F(i) = Max\left[\%Aciertosi - (\%Aciertosi - \%PrototiposSeleccionadosi), 0\right].$$

Esto es, a cada individuo se le asigna un valor que depende del porcentaje de aciertos que se obtiene utilizándolo como modelo de clasificación y del número de prototipos que selecciona para formar el modelo clasificatorio, tratando de mantener un equilibrio entre comportamiento y tamaño del modelo. Por ello, el valor será el porcentaje de aciertos obtenido, menos la diferencia entre este mismo porcentaje y el porcentaje de prototipos seleccionados, siempre que esta resta no de como resultado un numero negativo, en cuyo caso el valor asociado al individuo será 0.

Operador de selección

La función de selección utilizada ha sido la función de selección proporcional a la función objetivo, mediante la cual, la probabilidad de que cada individuo sea seleccionado es proporcional al valor que se le asigna en su evaluación, por lo que para individuos cuya relación entre %Aciertos y %Prototipos seleccionados sea mejor serán favorecidos con respecto a los demás.

Operador de cruce

Para el caso concreto de la selección de prototipos del conjunto de entrenamiento, el cruce basado en un punto será un operador de cruce un tanto "tosco" ya que no intenta aprovechar, al menos a priori, ninguna buena propiedad de sus progenitores. En cambio si el cruce se hiciera para cada clase individualmente, cruzando los genes pertenecientes a prototipos de una determinada clase entre si, es de esperar que se pudiesen obtener individuos que obtuvieran mejores prestaciones al aprovechar las buenas características de los individuos para cada clase. Esta idea ha sido precisamente la que ha dado lugar al operador de cruce del problema. Para su aplicación será necesario una ordenación de los prototipos que representan los genes por clases. Aplicando a continuación el cruce basado en un punto por clases (es decir entre los genes que representan a los prototipos de una misma clases) a los 2 progenitores para dar lugar a los 2 descendientes correspondientes.

Veamos el funcionamiento de este operador mediante un ejemplo:

Prototipos=$\{P_1, P_2, P_3, P_4, P_5, P_6, P_7, P_8, P_9, P_{10}\}$
Clases del problema=$\{c_1, c_2, c_3\}$
Clases de pertenencia de los prototipos:

$Clasec_1$: P_1, P_3, P_8

$Clasec_2$: P_4, P_5, P_9, P_{10}

$Clasec_3$: P_2, P_6, P_7.

Formato de los individuos para el cruce: $\{P_1, P_3, P_8, P_4, P_5, P_9, P_{10}, P_2, P_6, P_7\}$ (esto es, ordenados por clases)

En la figura 6.5 se puede ver un ejemplo del funcionamiento de este operador de cruce.

Operador de mutación

La aplicación del operador de mutación a los individuos del problema se hará por cada gen, mutando el valor del gen i al valor dado por la función Not (gen i). Así la probabilidad de mutación de cada gen i será muy pequeña, y diferente si el valor del gen es un "0" o un "1".

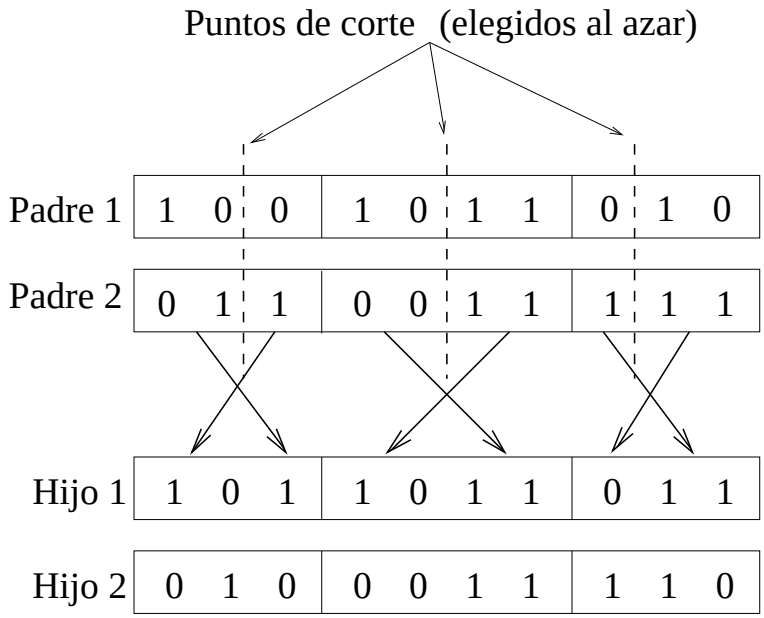


Figura 6.5: Operador de cruce. Se realiza un corte por cada clase.

Siendo este valor proporcional al número de prototipos seleccionados en el individuo (número de genes a "1" en el individuo alto, probabilidad de mutación para los genes a "0" baja y alta para los genes a "1" y viceversa). Mediante este criterio se pretende que al mismo tiempo que se produce un pequeño cambio aleatorio en el individuo se produzca con más probabilidad para aquellos valores de los genes en los que pueda estar "más necesitado" el individuo. Finalmente la probabilidad de mutación para los genes de cada individuo i será:

Genes "0" $\rightarrow \frac{(100-\%Prototipos seleccionados i)}{1000}$

Genes "1" $\rightarrow [100 - \frac{(100-\%Prototipos seleccionados i))}{5000}]$

Reducción de las poblaciones

Las poblaciones se han reducido siguiendo el criterio de la reducción elitista de grado N, siendo N el número de individuos de la población en cada generación (tamaño de la población). Mediante este criterio se eligen de entre los N padres y N hijos correspondientes, los N individuos que mejor valor de función objetivo presentan tras una iteración del algoritmo, los cuales pasan a formar parte de la siguiente generación.

Generación de la población inicial

La población inicial se genera mediante la activación de un número de genes por individuo que oscila entre 1% y % Máximo Prototipos al finalizar el algoritmo genético, eligiendo este número al azar. Este % máximo de prototipos al finalizar el algoritmo genético, es un parámetro más del algoritmo a introducir por el usuario y hace referencia al número de prototipos que se quiere seleccionar como máximo al finalizar el algoritmo genético del total de prototipos del conjunto de entrenamiento. La activación de los genes se realizará de forma aleatoria entre todos los genes del individuo que un principio estarán a "0". Al finalizar se tendrán individuos que como mínimo tendrán un 1 al finalizar el AG.

Estructura General del Algoritmo

La estructura general del algoritmo – veáse Figura 6.6 – es muy similar a la del *algoritmo genético simple*, si bien tiene algunas peculiaridades propias del problema concreto abordado.

6.4.2 Algoritmo PBIL en la selección de prototipos

De la misma manera que sucede con los AAGG, se pueden utilizar algoritmos de la familia EDA para realizar la selección de prototipos[1]. En el trabajo desarrollado en esta tesis, se ha utilizado una versión del algoritmo Population Based Incremental Learning (PBIL, Baluja 1994), que pasa a describirse a continuación.

El algoritmo PBIL

Es este uno de los algoritmos de la familia EDA, que asume un modelo probabilístico muy simple, en el que todos los atributos de la base de datos son independientes[2] entre si. En el algoritmo PBIL, cada uno de los bits que componen un individuo dado es examinado independientemente (recordar que un individuo tendrá tantos bits como casos tenga la base de datos). La distribución de probabilidad que se utiliza para muestrear cada uno de los bits que compondrán los individuos de la nueva población es aprendida de la siguiente forma:

$$p_l(\mathbf{x}_i) = (1-\alpha)\cdot p_{l-1}(\mathbf{x}_i|D_{l-1}) + \alpha \cdot p_{l-1}(\mathbf{x}_i|D^s_{l-1})$$

[1] En el problema de selección de prototipos, cada variable representa un caso, instancia o prototipo de la base de datos de entrenamiento. Se consideran dos valores para cada una de las variables que componen la solución candidata: '0' denota la ausencia del prototipo asociado en el subconjunto seleccionado, mientras que el valor '1' denota la presencia de la instancia en el subconjunto elegido.

[2] Asumiendo la siguiente factorización: $p_l(\mathbf{x}) = \prod_{i=1}^{N} p_l(\mathbf{x}_i)$.

Algoritmo genético para selección de prototipos

```
Comienzo AG
  Como entrada, se recibe la base de datos de entrenamiento

  Calcular la matriz de distancias entre los casos de la base de datos
  Generar Población Inicial de Individuos
  Computar Función Objetivo (F) de cada individuo
  Recuento de Genes a '1' por individuo (Datos para Evaluación-Mutación)
  MIENTRAS (No se cumpla el criterio de parada) HACER
    NumIteración = NumIteración + 1
    PARA (Tamaño de la población/2) HACER
        Selección 2 Padres para el cruce (Selección Proporcional a F)
        Cruce los 2 individuos
    Fin Para
    Mutación de todos los Descendientes con cierta probabilidad
    Computar Función Objetivo(F) de todos los descendientes
    Generar Población siguiente (Elitista de grado Tamaño Población)
    Recuento Genes a '1' por individuo(Datos para Evaluación-Mutación)
    Actualizar MejorIndividuo
    SI (NumIteracionActual > NumMaxIteraciones) ENTONCES
        Fin = True
    Sino
        Si (NumIndividuosInfractores > = Tamaño Población) ENTONCES
            Fin = True
        Fin Si
    Fin Si
  Fin Mientras
  Devolver Mejor Solución
Fin AG
```

Figura 6.6: Algoritmo genético para la selección de prototipos en K-NN.

- $p_{l-1}(\mathbf{x}_i|D_{l-1})$ es la distribución de probabilidad estimada para el bit i-ésimo en la población actual.
- $p_{l-1}(\mathbf{x}_i|D^s_{l-1})$ es la distribución de probabilidad estimada para el bit i-ésimo en los individuos seleccionados para dar origen a la nueva población.
- α es un parámetro introducido por el usuario para garantizar la evolución.

En lugar de descartar todos los individuos de la población actual, el mejor individuo obtenido hasta el momento se guarda para ser mantenido en la siguiente población, y se muestrean $N^* - 1$

nuevos individuos, siendo N^* el tamaño de la misma. De este modo se garantiza que el mejor individuo que devuelva el algoritmo va a ser realmente el mejor de los que ha encontrado a lo largo de la ejecución del mismo. Un criterio más elitista todavía consiste en seleccionar para formar parte de la siguiente población a los N^* mejores individuos de entre la población anterior y los $N^* - 1$ nuevos generados. Esto implica rápidas convergencias, pero puede llevar a mínimos locales. En cualquier caso, esta ha sido la aproximación utilizada para abordar el problema de selección de prototipos mediante el algoritmo PBIL.

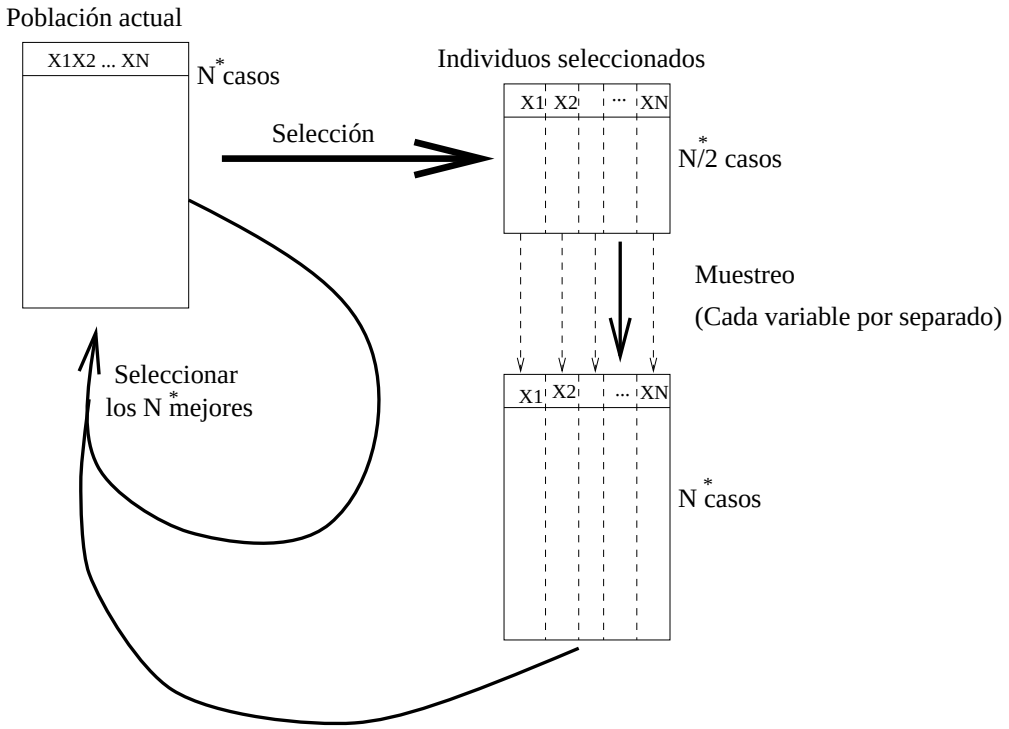


Figura 6.7: Funcionamiento general del algoritmo *Population Based Incremental Learning* (PBIL).

En la Figura 6.7 se puede ver una representación esquemática del funcionamiento del algoritmo. En este caso, la función de evaluación utilizada tiene en consideración tanto el porcentaje de bien clasificados que se obtiene utilizando los prototipos que señala el individuo cuyo valor se está calculando, como el porcentaje de prototipos que elige sobre el total de casos de la base de datos. De este modo, el valor de un individuo concreto vendrá dado por la ecuación:

$$Valor(Ind_i) = Aciertos_i + (CasosTot - CasosInd_i)/20.$$

Esto es, el valor de un individuo dado se calcula sumando al número de casos que clasifica bien el número de prototipos que no selecciona dividido entre 20.

6.4.3 Combinación con K-Means

Como ya se ha mencionado en la introducción de este capítulo, para abordar de forma efectiva la selección de prototipos mediante algoritmos evolutivos, es necesario que el tamaño de la base de datos de entrenamiento no sea excesivamente grande. En caso de que esto suceda, se propone la utilización de técnicas de clasificación no supervisada para dividir el conjunto de entrenamiento en una serie de subconjuntos más pequeños que sí puedan ser abordados satisfactoriamente por los algoritmos evolutivos presentados.

En concreto, se han realizado experimentos con el algoritmo K-Means (MacQueen, 1967), de la siguiente manera:

Dado un fichero de casos lo suficientemente grande, conteniendo N casos, dividirlo en s subficheros mediante la aplicación del algoritmo K-Means con el parámetro K = s.

Método de K medias de MacQueen (1967)

Comienzo K-Means
El algoritmo recibe como entrada la base de datos de casos
y el número de conglomerados, fijado de antemano
Paso 1: tomar los K primeros elementos del fichero de casos
como K conglomerados de un sólo objeto
Paso 2: asignar en el orden del fichero cada uno de los $N - K$ elementos restantes
al conglomerado con centroide más próximo. Después de cada
asignación computar de nuevo el centroide
Paso 3: después de que todos los elementos hayan sido asignados
en el Paso 2, tomar los centroides de los conglomerados obtenidos como
puntos gérmenes fijos y asignar cada objeto al punto gérmen más próximo
Paso 4: repetir los Pasos 2 y 3 hasta alcanzar la convergencia
(no se produzcan cambios en la reasignación)
Fin K-Means

Figura 6.8: Algoritmo *K-Means*.

Una vez realizada esta división, aplicar los algoritmos evolutivos que se desee utilizar a cada uno de los subficheros obtenidos, para posteriormente utilizar como prototipos la unión de los prototipos obtenidos para cada uno de los s subficheros. Señalar que el algoritmo K-Means devuelve s subconjuntos disjuntos. Ver el algoritmo en la Figura 6.8.

Parte V

COMBINACIÓN DE CLASIFICADORES

Capítulo 7

Combinación de clasificadores

7.1 Introducción

En el área de la clasificación supervisada, uno de los campos de investigación se centra en el modo en que se pueden combinar, unir o ensamblar varios clasificadores para obtener uno más potente, esto es, un clasificador con mejores resultados desde el punto de vista clasificatorio. Al clasificador que se obtiene de esta unión se le da el nombre genérico de *multiclasificador* (Xu, 1992). Un ensamblaje de clasificadores puede mejorar el resultado de los clasificadores individuales que lo componen únicamente si existen desavenencias entre ellos, esto es, si existen casos a los que distintos clasificadores asignan distinta clasificación (Hand, 1997). Se busca, por lo tanto, diversidad en los clasificadores componentes de un multiclasificador.

Para aclarar el concepto, supongamos que en un problema de clasificación supervisada disponemos de una base de datos de entrenamiento de la forma $\{(\mathbf{x}_1, \theta_1), (\mathbf{x}_2, \theta_2), \ldots, (\mathbf{x}_N, \theta_N)\}$ para un problema que puede ser visto de la forma $\Theta = f(\mathbf{x})$, esto es, se supone que la clase real Θ es función de las variables predictoras X. Normalmente la función es desconocida. Los valores $\mathbf{x}_i$ son vectores de la forma $< x_{i1}, x_{i2}, ..., x_{in} >$, siendo los componentes valores discretos, continuos o simbólicos. A partir de estos valores $\mathbf{x}_i$, un paradigma de clasificación supervisada trata de obtener una función $Y' = f'(\mathbf{x})$, de modo que se aproxime en lo posible a la función verdadera $\Theta = f(\mathbf{x})$, para que se pueda utilizar en su lugar. Denotamos este paradigma como h.

Supongamos que tenemos tres clasificadores, $\{h_1, h_2, h_3\}$ y consideremos un nuevo caso $\mathbf{x}_{N+1}$. Podemos tener varias posibilidades:

1. La predicción realizada por cada uno de los tres clasificadores es idéntica, e incorrecta[1].

[1] Suponemos un problema clasificatorio con dos clases. La extensión a un mayor número de clases es inmediata.

2. La predicción realizada por cada uno de los tres clasificadores es idéntica, y correcta.

3. Dos clasificadores predicen la clase errónea y el otro realiza la predicción correcta.

4. Dos clasificadores predicen la clase correcta y el otro realiza la predicción errónea.

Existen muchas formas de realizar la combinación de clasificadores. Algunas de ellas serán presentadas a continuación. La más sencilla puede ser la que se denomina *principio del voto de la mayoría*, que en este caso clasificaría correctamente el segundo y cuarto de los casos presentados, e incorrectamente los otros dos. En el primero y segundo caso un ensamblaje no variará la clasificación, mientras que para el tercero y el cuarto existen varias alternativas de combinación. En Dietterich (1997) se presentan de modo extenso distintas formas de construir un multiclasificador.

La razón en la que se sustentan los clasificadores múltiples para realizar combinaciones subyace en las diferentes características de los paradigmas de clasificación, que se basan en distintas teorías y metodologías. Dado un problema clasificatorio, diversas aproximaciones pueden obtener niveles de acierto distintos, aunque ninguno será, seguramente, el acierto pleno. Incluso siendo similares los niveles de acierto de los clasificadores, existirán instancias clasificadas de forma distinta, esto es, no clasificarán de la misma forma todo el espacio clasificatorio. Esta característica hace que existan formas de combinación que mejoren el comportamiento de cada uno de los clasificadores componentes.

Entre los temas a considerar para construir un clasificador múltiple están la cantidad de clasificadores componentes que se van a utilizar, junto con los recursos computacionales que ello requerirá, la diversidad de los mismos, el poder predictivo de los clasificadores componentes y la eficiencia del multiclasificador construido.

Existen distintas aproximaciones para la construcción de un clasificador múltiple. A continuación se presentan algunas de las formas de composición más utilizadas. Distinguimos, por un lado, el acoplamiento de varios clasificadores en uno solo, formando lo que se denomina *clasificador híbrido*, y por otro la conjunción de varios clasificadores, manteniendo cada uno de ellos su identidad, a lo que denominaremos propiamente *multiclasificador*.

7.2 Clasificadores híbridos

Una de las primeras formas que surgió para combinar clasificadores es hacerlo de modo que varios clasificadores, o más bien, sus ideas principales, formen un nuevo clasificador. Existen muchas posibilidades para realizar un algoritmo que sea híbrido de dos o más clasificadores. Para ello se adoptan ideas provenientes del razonamiento simbólico, la lógica difusa, el posibilismo o la teoría

de la probabilidad, y se mezclan con algoritmos genéticos, enfriamiento estadístico, búsqueda tabú u otra técnica de búsqueda para combinarla con las ideas provenientes de algún algoritmo de machine learning, redes neuronales o redes Bayesianas, obteniendo un único clasificador. Un ejemplo se puede ver en el software Robust Bayesian Classifier (Ramoni, 1999).

Dos ejemplos de clasificadores híbridos son los clasificadores Naive-Bayes Tree y PW-K-NN.

7.2.1 *Naive-Bayes Tree*

Este algoritmo, conocido como NBTree (Kohavi, 1996), está basado en la idea *divide y vencerás*.

Con el fin de incrementar la potencia clasificatoria del algoritmo *naive-Bayes*, se realiza una partición del espacio de búsqueda, de modo que el algoritmo NB propiamente dicho se aplica en cada una de las particiones realizadas. Para realizar la partición se utiliza un árbol de clasificación, en el que se efectúa una prepoda, esto es, se desarrolla hasta que en los casos de las hojas no son *significativamente* mejorados al dividirlos expandiendo el árbol. En este método, la división se realiza si la reducción del error es mayor que un 5% y además hay más de 30 instancias o casos en el subconjunto asociado al nodo.

De este modo, podemos ver el NBTree como un clasificador híbrido, en el cual se utilizan las ideas provenientes de los árboles de clasificación para obtener subconjuntos de la base de datos de entrenamiento, y luego se aplica el algoritmo NB en cada uno de los nodos terminales u hojas.

7.2.2 Probabilistic Weighted K Nearest Neighbour

Tal y como se vio en el capítulo dedicado a los algoritmos de vecindad, este algoritmo es un híbrido en el que se utilizan las ideas clasificatorias del algoritmo K-NN, pero utilizando un tipo de voto especial que viene dado por la tipicidad de los casos vecinos dada por una red Bayesiana.

Recordemos que los pasos a seguir en la ejecución de este algoritmo son los siguientes:

1. Seleccionar los K vecinos más próximos del caso a clasificar. Sean estos $(\mathbf{x}_1, \theta_1)$, $(\mathbf{x}_2, \theta_2)$, ..., $(\mathbf{x}_K, \theta_K)$.

2. Asignar a cada uno de estos K vecinos un peso en la votación que será el que la red Bayesiana a utilizar le da a la clase a la que realmente pertenece cada una de las instancias. Tenemos entonces $(\mathbf{x}_1, \theta_1, Peso_1), (\mathbf{x}_2, \theta_2, Peso_2), \ldots, (\mathbf{x}_K, \theta_K, Peso_K)$.

3. Asignar al caso a clasificar la clase cuya suma de pesos entre las presentes en los vecinos seleccionados sea mayor.

7.3 Combinación de clasificadores

En el área de *machine learning* se ha demostrado que la combinación de clasificadores es un sistema efectivo para la resolución de problemas complejos. Mediante la combinación, permitimos que los clasificadores que participan –clasificadores componentes– compensen sus deficiencias individuales, integrando el conocimiento proveniente de diversas fuentes. Es una forma de obtener clasificadores con un alto poder clasificatorio mediante la utilización de clasificadores menos potentes, integrando los resultados de los mismos para tratar de alcanzar un resultado global mejor.

Dietterich (1997) cita tres causas para la menor potencia clasificatoria de los clasificadores individuales, que son superadas, en cierto modo, mediante la combinación de los mismos:

1. Un algoritmo de *machine learning* busca en el espacio de posibles hipótesis $\mathcal{H}$ tratando de encontrar la más acertada. Si el espacio de búsqueda es muy grande, se necesita un conjunto de entrenamiento de gran tamaño para realizar una buena aproximación. Respecto a los clasificadores individuales, las hipótesis más acertadas encontradas por cada uno de ellos en la construcción del modelo son utilizadas por la combinación de los mismos, de modo que, al menos la mejor, sea igualada.

2. Los clasificadores individuales pueden no ser capaces de solucionar problemas clasificatorios de mucha complejidad. Los algoritmos de aprendizaje de modelos utilizan heurísticos de búsqueda en el espacio de posibles hipótesis, con el ánimo de consumir menos tiempo en la construcción del modelo clasificatorio. Esto hace que la hipótesis encontrada no sea, en general, la mejor.

3. El espacio de búsqueda que forman las hipótesis por las que se mueve un clasificador individual es, generalmente, un subconjunto de todas las asociadas al problema. Es posible que la mejor no esté incluida en ese subconjunto. Cada algoritmo tiene tipos de hipótesis que le son imposibles de encontrar. Esto se puede observar fácilmente en los algoritmos de aprendizaje de árboles de clasificación, como son *ID3* o *C4.5*: la linea de separación entre las clasificaciones que se realizan viene determinada por el valor de uno o varios atributos, de modo que la forma que tienen las lineas de separación entre las clases toma una forma escalonada. En la Figura 7.1 se presenta un ejemplo de esto. La imagen de la izquierda muestra el límite real, diagonal, entre las distintas clases, junto con una aproximación escalonada al mismo, que representa la forma en que se aproxima este límite mediante la utilización de árboles de clasificación. La imagen de la derecha muestra la separación obtenida mediante la decisión que se toma al utilizar varios clasificadores, resultando una aproximación mejor a la diagonal que muestra la separación real.

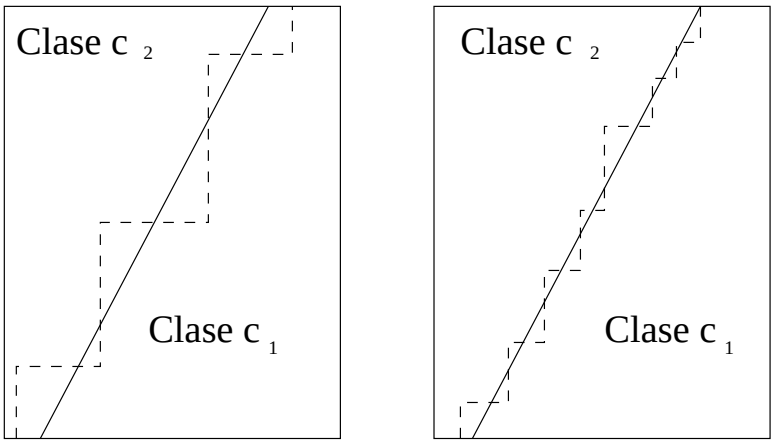


Figura 7.1: Comportamiento de un clasificador individual (izquierda) y uno múltiple (derecha). La diagonal representa la división real entre las clases.

Del mismo modo, el paradigma *naive Bayes* no supera a la clasificación aleatoria en un problema clasificatorio en el que los casos se encuentren distribuidos de forma cruzada, tal y como se muestra en la Figura 7.2. Esto es debido a que las probabilidades de la clase c_1 y de la clase c_2 son casi iguales para cualquier valor de los atributos X_1 y X_2. Como consecuencia, la asunción de independencia realizada sobre los atributos no aporta información sobre la probabilidad de cada clase.

Lu (1996) cita 3 configuraciones para combinar clasificadores: cascada, paralelo y jerárquica (Ver la Figura 7.3). Algunas de sus características son:

- En un *sistema en cascada*, los resultados de la clasificación generados por cada clasificador son utilizados, normalmente, para dirigir los procesos de clasificación de los paradigmas utilizados a continuación. El problema principal de esta aproximación es que los errores cometidos por los clasificadores del inicio del proceso son de difícil recuperación para los clasificadores restantes.

- En un *sistema paralelo*, se realiza una integración real del conocimiento de los diversos paradigmas, generados independientemente. Si el proceso de decisión se diseña bien, puede alcanzar un comportamiento que supere al de los clasificadores individuales que lo forman.

- En un *sistema jerárquico*, la estrategia de control es una combinación entre los procesos de cascada y paralelo.

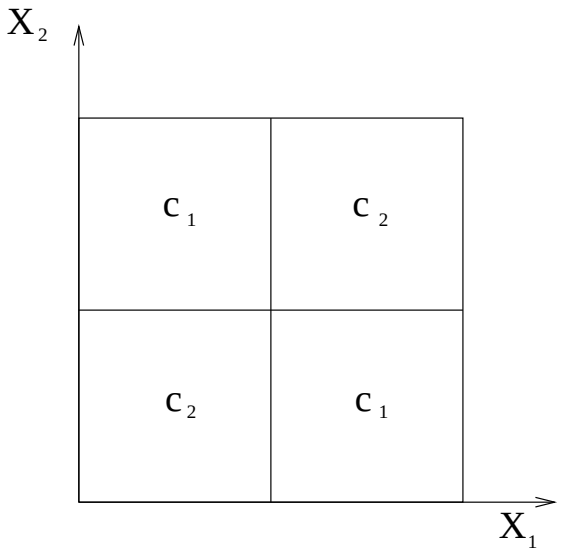


Figura 7.2: Este tipo de información uniformemente cruzada no puede ser adecuadamente manejado mediante el clasificador *naive Bayes*.

Otros trabajos sobre la combinación de clasificadores han sido realizados por Dietterich (1997) y por Ting (1997).

Centraremos nuestra atención en los métodos de combinación paralelos. A continuación se presentan algunas de las formas típicas de construcción de clasificadores múltiples con arquitectura paralela.

7.3.1 Principio del voto por la mayoría

Esta técnica consiste en aplicar a un caso a clasificar varios paradigmas, y asignarle como clasificación aquella que se halle más fuertemente representada de entre los resultados obtenidos. En su forma más básica, suponiendo que utilizamos K clasificadores, se asignará al nuevo caso la clase mayoritaria de entre los K votos.

Existen muchas alternativas a considerar, dando lugar a numerosas variantes en la aplicación de este método, de manera análoga a lo que se pudo entrever en el capítulo dedicado a los algoritmos de clasificación por vecindad.

Quizás lo más reseñable sea el uso generalizado de una clase extra, la $M+1$ en un problema de M clases, que indica que el proceso de votación no ha sido capaz de determinar una clase como

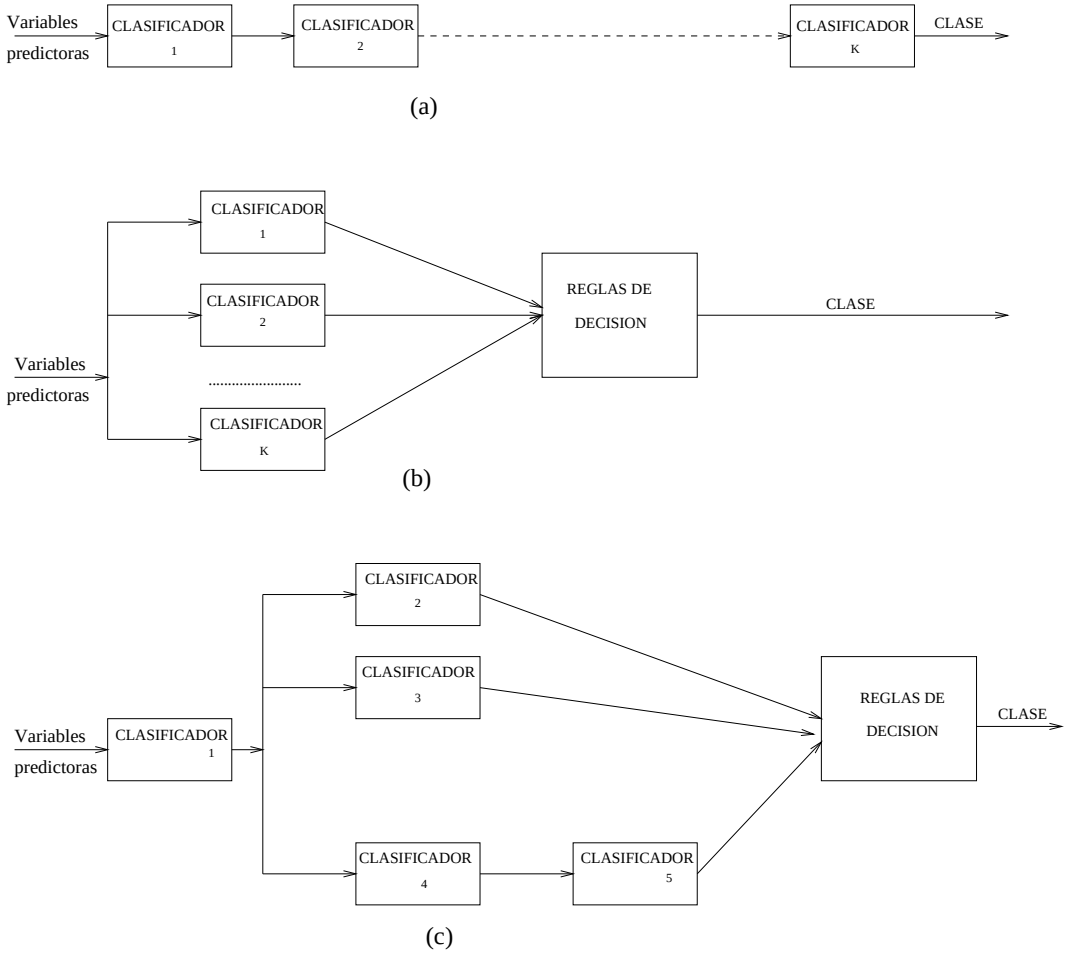


Figura 7.3: Ejemplos de las tres arquitecturas de un clasificador múltiple: (a) Cascada (b) Paralelismo (c) Jerárquico.

vencedora. Si el multiclasificador devuelve la clase $M+1$ se deben investigar otras técnicas de clasificación para asignar una clase al caso a tratar.

7.3.2 El formalismo Bayesiano

Suponiendo de nuevo que utilizamos K clasificadores, $h_1, h_2, ..., h_K$ el principio de la mayoría trata por igual los resultados que ofrece cada uno de ellos al clasificar un caso $\mathbf{x}$, $h_l(\mathbf{x}) = \theta_j$; $l = 1, \ldots, K$, sin considerar los errores que cada clasificador h_l ha cometido en la fase de entrenamiento ni la potencia clasificatoria de cada uno de ellos. El formalismo Bayesiano, por el contrario, tiene en consideración el comportamiento mostrado por cada uno de los K paradigmas utilizados con los casos de la base de datos de entrenamiento. Para ello se basa en la matriz de confusión, que tiene el siguiente aspecto para un problema de M clases y para cada clasificador h_l:

$$PT^l = \begin{pmatrix} n^l_{11} & n^l_{12} & . & . & . & n^l_{1M} \\ n^l_{21} & n^l_{22} & . & . & . & n^l_{2M} \\ . & . & . & . & . & . \\ n^l_{M1} & n^l_{M2} & . & . & . & n^l_{MM} \end{pmatrix}$$

para $l = 1, \ldots, K$. En esta matriz cada fila i corresponde a la clase real θ_i, mientras que cada columna j corresponde a la predicción realizada por el clasificador, $c_l(\mathbf{x}) = \theta_j$ para el ejemplo $\mathbf{x}$. De este modo, el elemento (i, j) de la matriz PT^l, denota que n^l_{ij} ejemplos de la clase θ_i han sido considerados por el clasificador c_l como pertenecientes a la clase θ_j.

La matriz de confusión PT^l puede ser vista como algo semejante al conocimiento previo que un experto tiene sobre el problema. Con los datos de PT^l, podemos expresar mediante un número real $bel(.)$, denominado *valor de creencia*, la incertidumbre que se tiene sobre cada una de las M posibles predicciones a realizar $\mathbf{x} \in \theta_i$, $\forall i \in 1, \ldots, M$:

$$bel(\mathbf{x} \in \theta_i | h_j(\mathbf{x})) = P(\mathbf{x} \in \theta_i | h_j(\mathbf{x}) = \theta_j)$$

Este *valor de creencia* es calculado bajo la condición de que el clasificador h_j realice la predicción $h_j(\mathbf{x}) = \theta_j$.

Al disponer de K clasificadores, contamos con K matrices de confusión, $PT^1, PT^2, ..., PT^K$. Al obtener las K predicciones que estos clasificadores asignan al caso $\mathbf{x}$, podemos combinar e integrar este conocimiento de la forma indicada, expresando de cierta manera la creencia que los K clasificadores tienen de que el caso $\mathbf{x}$ pertenezca a la clase θ_i.

$$bel(\mathbf{x} \in \theta_i) = \prod_{l=1}^{K} P(\mathbf{x} \in \theta_i | h_l(\mathbf{x}) = \theta_j)$$

Por último, dependiendo de estos valores $bel(\mathbf{x} \in \theta_i)$, se puede clasificar el caso $\mathbf{x}$ en una única clase aplicando la siguiente regla de decisión:

$$CreenciaFinal(\mathbf{x}) = \theta_j,$$

siendo $bel(\mathbf{x} \in \theta_j) = max_{i \in \wedge}$ y $\wedge$ el conjunto correspondiente a las M clases posibles.

7.3.3 Clasificadores multicapa

Un método para la combinación de clasificadores es el que ha sido denominado *stacked generalization* (Wolpert, 1992), y que consiste en la creación de clasificadores por capas. Cada una de las capas está formada por uno o más clasificadores, cuyas predicciones son utilizadas por los clasificadores de la capa inmediatamente superior para ser combinadas. La última capa está formada por un único clasificador, que es el que toma la decisión final.

La información que circula de una capa a la superior puede ser en forma de vectores que incluyan la predicción realizada por cada uno de los clasificadores, niveles de confianza y cualquier otro tipo de información que pueda resultar útil. Al ser esta una introducción nos centraremos en aquellos casos en los que la información enviada a la capa superior es exclusivamente la clasificación asignada por los paradigmas de la capa actual.

En su forma más básica, consiste en una serie de clasificadores que forman la primera capa. Wolpert denomina a este conjunto *clasificadores de nivel 0*, mientras que los de la capa inmediatamente superior son los de nivel 1 y así sucesivamente. En la Figura 7.4 se observa el esquema de un multiclasificador con dos capas.

Esta técnica de combinación de clasificadores es un intento de minimizar los errores de los clasificadores de las capas inmediatamente inferiores, utilizando una serie de capas que permitan determinar o aprender los tipos de errores asociados a cada paradigma. Esto está relacionado, en cierto modo, con técnicas de validación, como pueden ser la validación cruzada o el *bootstrapping*.

Una extensión a este paradigma, investigada por Shaffer (1994), consiste en una combinación multicapa en la que las capas superiores tienen acceso a los valores de las variables pasados como parámetro a los clasificadores de la capa 0, y no sólo a las predicciones realizadas por los clasificadores de la capa inferior. La idea es utilizar esta información para determinar de un modo adecuado la forma de combinar las opiniones aportadas por cada clasificador. A esta aproximación se le ha dado el nombre de *bi-level stacking*.

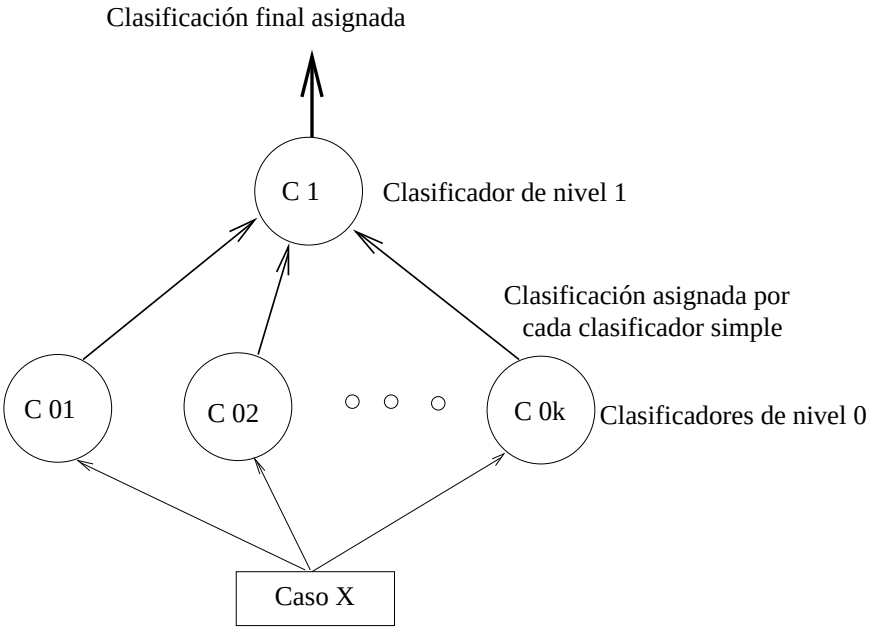


Figura 7.4: Esquema de un clasificador bi-capa.

7.4 Particionamiento recursivo

El *particionamiento recursivo* es otra forma de construir clasificadores múltiples. También conocido como *divide y vencerás*, utiliza una estrategia para particionar el espacio de instancias en regiones que contienen instancias pertenecientes a una única clase. Es una técnica de combinación de clasificadores que puede ser vista también como un híbrido entre varios paradigmas clasificatorios.

Existen varias opciones para realizar este tipo de clasificadores múltiples. La técnica denominada *mode class selection* (Brodley, 1994) crea recursivamente un clasificador múltiple en forma de árbol, que utiliza una combinación de árboles de clasificación, clasificadores basados en la regresión lineal y clasificadores de la familia del aprendizaje basado en casos. En la construcción del multiclasificador, se utiliza información heurística sobre el comportamiento de los diversos paradigmas utilizados. En la figura 7.5 se presenta el esquema asociado a esta idea.

Una ventaja que presentan este tipo de aproximaciones es que la elección del clasificador se realiza en base a las características de la región que se desea dividir, esto es, de alguna manera se utiliza un clasificador propio para cada región. Otra ventaja reside en la eficiencia de la aplicación del método, no es necesario aplicar todos los clasificadores a todas las instancias, tal y como sucede en los clasificadores multicapa.

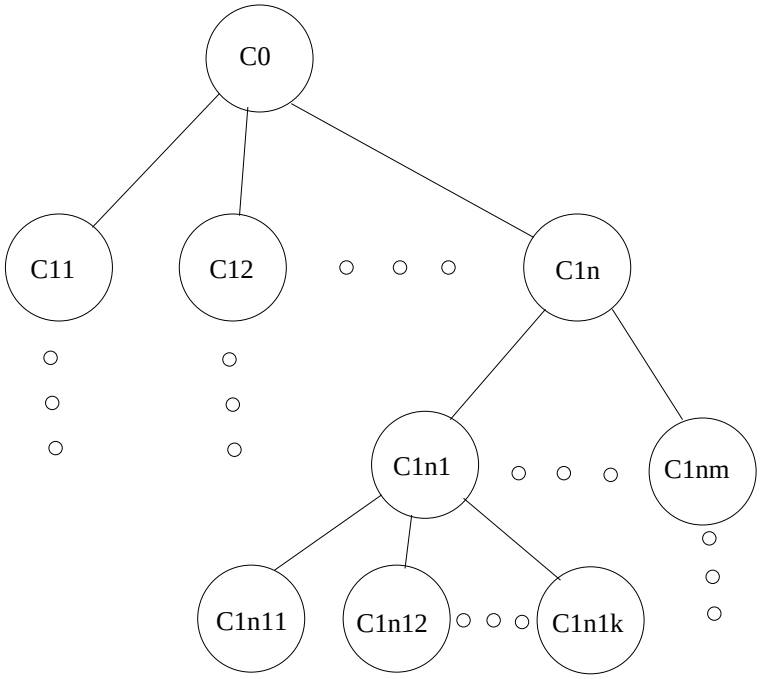


Figura 7.5: Esquema del particionamiento recursivo.

En cuanto a las desventajas, al dividir el espacio se cuenta con un menor número de instancias en cada región, lo cual puede repercutir en el proceso de aprendizaje del modelo de clasificación que se va a utilizar en ellas. Un menor número de casos en el fichero de entrenamiento puede llevar a un clasificador con pobres resultados. Otra desventaja reside en el hecho de que sólo se utiliza un clasificador en cada instancia, el correspondiente a la región a que pertenece, lo cual impide que se aprovechen opiniones provenientes de diferentes paradigmas.

7.5 Otro tipo de aproximaciones

Al igual que en el Capítulo 2, donde nos centrábamos en los paradigmas de clasificación supervisada con un número de clases finito, indicando que existen problemas donde la clase puede ser un número real, en el caso de los clasificadores múltiples no hemos mencionado el hecho de que puede que la combinación se realice en base a otros criterios, en vez de basarse únicamente en la clase asignada por los clasificadores componentes. Algunas opciones son las siguientes:

- Combinar los clasificadores teniendo en cuenta el orden que asignan a cada clase. En este caso, los clasificadores indican el orden de preferencia de las clases, de modo que la combinación se complica, disminuyendo a su vez el número de empates que se producen.

- Es posible que los clasificadores aporten información de otro tipo, indicando para cada clase el nivel de confianza que se supone tiene. Un ejemplo de esto son los clasificadores Bayesianos, que dan un valor de probabilidad a cada una de las clases. Con este tipo de clasificadores la construcción de un multiclasificador puede complicarse aun más.

Capítulo 8

Aportaciones realizadas en el área de los clasificadores múltiples

8.1 Introducción

Durante los últimos años, y en una gran cantidad de dominios de aplicación, los investigadores del área de *machine learning, reconocimiento de patrones, teoría de la computación* y la *estadística* han renovado el ánimo de tratar de descubrir cómo combinar y ensamblar varios clasificadores, haciéndolo además en problemas reales.

Combinar las predicciones de varios clasificadores ha demostrado obtener unos niveles de clasificación superiores al mejor de los clasificadores individuales que se combinan en una gran variedad de problemas de clasificación (Ho y col., 1994, Inza y col., 1998).

8.2 Esquema del multiclasificador Bayesiano

Durante el desarrollo de esta tesis, se ha diseñado un multiclasificador de dos capas (Sierra y col., 1999b), en el cual se utilizan una serie de clasificadores estándares de *machine learning* como elementos que forman la primera capa, esto es, como clasificadores *simples*, mientras que la segunda y última capa la forma una red Bayesiana inducida sobre las predicciones realizadas por los clasificadores de la capa previa. De este modo, la RB actúa, en cierta forma, como elemento consensuador de las diferentes *opiniones* que aportan los diferentes clasificadores estándares utilizados (Sierra y col., 1999b, 2000b). Una idea parecida se utiliza en Inza y col., (1999a) para ver las interrelaciones existentes entre diversos tipos de clasificadores en diferentes problemas.

La idea principal del multiclasificador es la siguiente: se dispone de una serie de l clasificadores individuales, $h_1, h_2, \ldots, h_l$ formando la primera capa; con los resultados obtenidos por éstos sobre

una base de datos D, conteniendo N_1 casos, se construye otra base de datos de entrenamiento $D2$, en la que cada una de las primeras l columnas se corresponde con los resultados obtenidos por cada uno de los l clasificadores sobre los N_1 casos, mientras que la columna $l+1$, la última, se corresponde con la clase real de los casos de la base de datos D. A partir de la base de datos D_1 se realiza el aprendizaje estructural de una red Bayesiana con objetivos clasificatorios, en la que existe una variable asociada a cada uno de los clasificadores, mas una variable asociada a la clase real. Esta RB será la que se utilize en la segunda capa para realizar el voto consensuado de las predicciones dadas por los l clasificadores utilizados, correspondiendo sus predicciones a instanciaciones de los nodos de la red Bayesiana.

Una vez construido, el multiclasificador realiza el siguiente proceso para asignar una clase a un caso que se le presente: en primer lugar, el nuevo caso es planteado a cada uno de los clasificadores individuales que forman la primera capa, para que cada uno de ellos aporte su opinión respecto a la clase de pertenencia que se debe predecir. Una vez obtenidos los resultados de estos clasificadores, la clase predicha por cada uno de ellos se toma como valor de la variable asignada a los mismos en la RB que forma la segunda capa, instanciando de este modo todas las variables de la RB excepto la correspondiente a la clase. Una vez instanciadas las variables, se realiza la propagación de la evidencia a través de la RB, y se obtienen las probabilidades a posteriori de cada clase, asignando finalmente al nuevo caso la clase que tenga una probabilidad de pertenencia mayor, esto es, la de mayor valor en el nodo asociado a la clase de la RB.

El nuevo método aquí presentado está basado en la técnica de combinación de clasificadores denominada *stacked generalization* (Wolpert, 1992), presentada en el capítulo anterior.

8.2.1 Estructura del multiclasificador

La estructura del multiclasificador diseñado puede verse en la Figura 8.1. Se ha buscado diversidad y eficacia o potencia clasificatoria en detrimento de la eficiencia y el consumo de recursos computacionales.

En los clasificadores multicapa se utilizan los resultados de la capa inferior por los clasificadores de la capa superior, realizándose de este modo una combinación ascendente. En el ejemplo utilizado, disponemos de nueve clasificadores estándares en la capa inferior, mientras que la segunda y última está formada por un único clasificador, una red Bayesiana que sirve para combinar las opiniones de los clasificadores de la primera capa. De este modo, se realiza una especie de *voto consensuado* sobre las predicciones realizadas por los clasificadores estándares, asumiendo como buena la idea de tener en consideración las diferentes relaciones que existen entre las predicciones realizadas sobre los casos de la base de datos de entrenamiento por los distintos clasificadores utilizados. La intención es que estas relaciones se vean reflejadas en la estructura de la red Bayesiana,

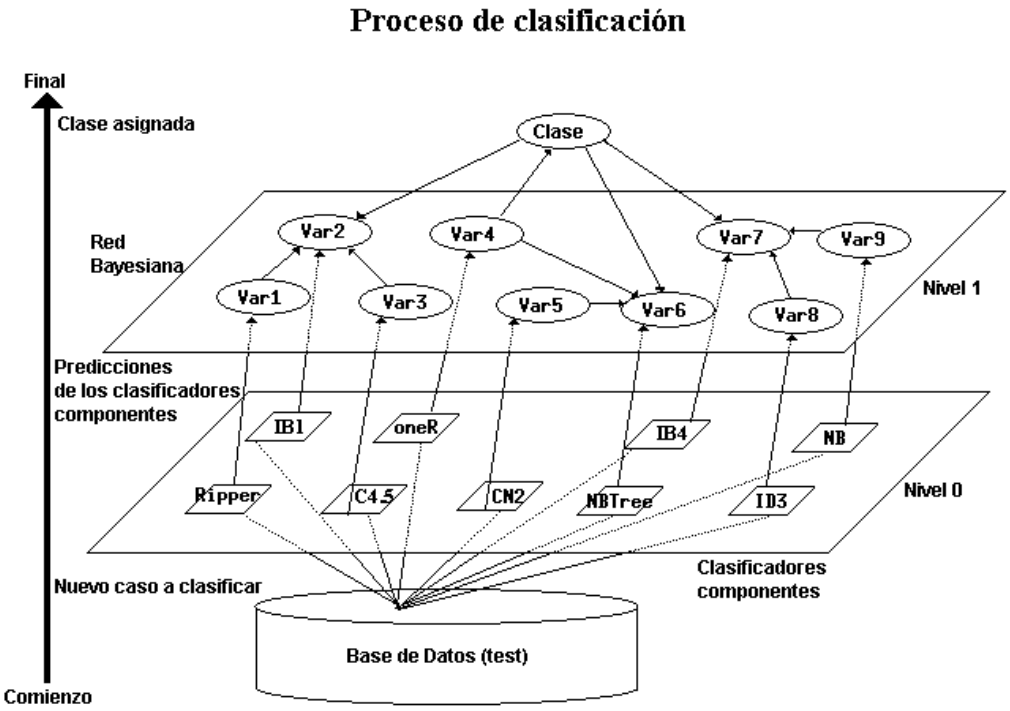


Figura 8.1: Estructura del multiclasificador

de modo que se identifiquen las dependencias e independencias –condicionales o no– existentes entre los paradigmas utilizados, para el caso de estudio tratado, dándonos de este modo una idea de la forma en que es abarcado el espacio de búsqueda. Mediante esta red Bayesiana realizamos el último paso de la clasificación.

8.2.2 Construcción del multiclasificador

Se presenta en esta sección la metodología utilizada para la construcción del *multiclasificador*. En la Figura 8.2 se puede ver este proceso de forma esquemática.

Para cada clasificador estándar, se aprende el modelo clasificatorio correspondiente. Para ello se puede utilizar cualquier software disponible, como por ejemplo la librería de libre distribución MLC++ (Kohavi y col., 1996).

Para aprender el modelo de red Bayesiana que se utilizará en la segunda capa del multiclasificador, es necesario disponer de una segunda base de datos de entrenamiento, distinta a la anterior,

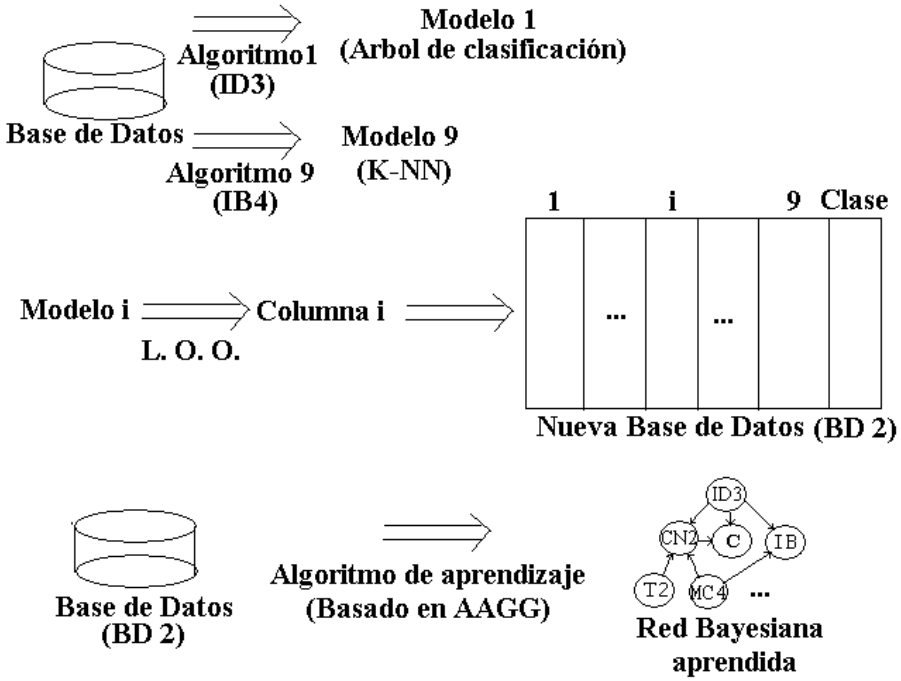


Figura 8.2: Construcción del multiclasificador.

en la que las variables representen la clasificación que a un caso le ha asignado un clasificador determinado. En esta segunda base de datos, contamos por lo tanto con tantas variables predictoras como clasificadores hemos utilizado en el primer nivel, más la variable asociada al valor real de la clase, y además se da la circunstancia de que todas las variables toman un conjunto de posibles valores idéntico, el correspondiente al número de clases que se han tenido en cuenta en el problema clasificatorio planteado.

La forma de obtener esta segunda base de datos varía según el tamaño del conjunto de datos inicial. Dependiendo de ello, los pasos a seguir son los siguientes:

Base de datos inicial de pequeño tamaño

Se ejecuta una secuencia de validación *leave-one-out* para cada uno de los clasificadores utilizados en la primera capa, obteniéndose para cada uno de ellos una salida que constituirá la columna correspondiente de la base de datos de entrenamiento de la que se inducirá la red Bayesiana.

Base de datos inicial de tamaño grande

Se divide la base de datos inicial en dos, de modo que una sirve de entrenamiento para los clasificadores de la primera capa, mientras que la segunda parte sirve para obtener la segunda base de datos al ser los casos que contiene clasificados por todos y cada uno de los paradigmas de la capa inferior. De este modo se obtiene la segunda base de datos, con casos no utilizados previamente, pero de los que se conoce su clase real.

8.2.3 Clasificadores de nivel 0

Tal y como se desprende de la tesis de Skalak (1994, 1997), el objetivo principal de la selección de los clasificadores que van a formar parte de un multiclasificador consiste en obtener la mayor diversidad posible en las predicciones realizadas por cada uno de ellos, manteniendo, eso si, buenos niveles de predicción para el problema que se esté tratando. Debido a esto, en la construcción del multiclasificador propuesto se utilizan diversos algoritmos estándares provenientes del área *machine learning*:

- **Árboles de clasificación**
  Se utilizan dos algorimos de aprendizaje de árboles de clasificación muy conocidos: ID3 (Quinlan, 1986) y C4.5 (Quinlan 1993).

- **Aprendizaje Basado en Casos: Instance-Based Learning**
  Se utilizan dos algoritmos de esta familia: IB1, un algoritmo implementado en la librería de algoritmos estándares MLC++ (Kohavi y col, 1997), basado en los trabajos de Aha (1992) y de Wettschereck (1994). IB4 es una extensión del anterior que incluye la asignación durante el proceso de aprendizaje de diferentes pesos a los atributos en base a su importancia relativa en el proceso de clasificación.

- **Inducción de reglas**
  De esta familia se utilizan los algoritmos CN2 (Clark y Nibblet, 1989), oneR (Holte, 1994) y Ripper (Cohen, 1995) para inducción automática de reglas.

- **Naive Bayes**
  En el multiclasificador se utiliza el naive-Bayes y el clasificador naive Bayes Tree, que, como se ha mencionado previamente, es un híbrido entre el NB y los árboles de clasificación.

En total se utilizan nueve algoritmos de nivel 0.

8.2.4 Clasificador de nivel 1: red Bayesiana

Como clasificador de nivel 1 utilizamos una red Bayesiana aprendida utilizando el concepto de *Markov blanket* sobre el conjunto de entrenamiento compuesto por una segunda base de datos, obtenida a partir de los resultados asignados por los clasificadores de nivel 0 sobre una serie de datos de los que se conoce la clasificación real. De este modo pretendemos obtener información sobre las dependencias e independencias –condicionales o no– que los diferentes clasificadores muestran entre si para el problema en estudio, y conseguir que la combinación se realice de un modo más adecuado.

Teniendo en cuenta que, en una red Bayesiana, cada variable es influenciada por lo que se denomina su manto de Markov o *Markov blanket* (MB), que está formado por sus nodos padres, sus nodos hijos y los nodos padres de sus hijos (ver Figura 8.3), consideramos que esta estructura puede ser adecuada para representar las influencias que los diferentes clasificadores aportan a la clase real, haciendo que todos ellos formen el MB del nodo que representa a la clase real.

Realizamos el aprendizaje de la RB mediante *algoritmos genéticos* en los que cada individuo representa un posible MB de la variable a clasificar.

Otra posibilidad sería realizar un aprendizaje con objetivos clasificatorios de una estructura general, ya que de este modo podríamos realizar al mismo tiempo una selección de los clasificadores que forman parte del modelo, ya que los que no participen en el MB de la variable asociada a la clase real no serán considerados en la clasificación.

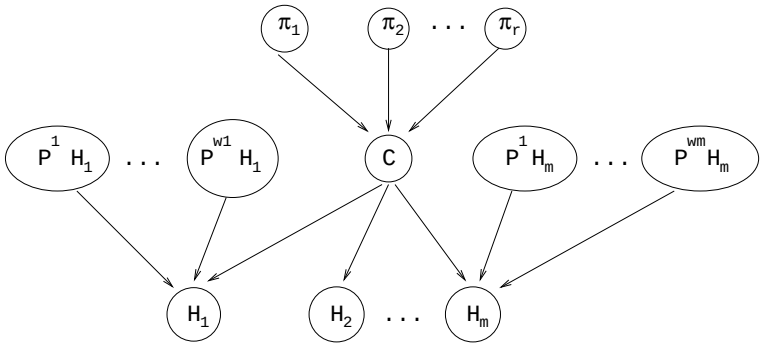


Figura 8.3: Estructura *Markov Blanket* general.

Parte VI

RESULTADOS EXPERIMENTALES

Capítulo 9

Problemas reales provenientes del mundo médico

9.1 Introducción

Se presentan en este capítulo los resultados experimentales que a lo largo del desarrollo de esta tesis se han obtenido al aplicar técnicas novedosas de clasificación supervisada en problemas médicos reales.

9.2 Predicción de la supervivencia en pacientes con melanoma maligno

9.2.1 El melanoma maligno

A pesar de los avances realizados en los últimos años en el tratamiento del cáncer, la prognosis sobre pacientes afectos de melanoma maligno, conocido como cáncer de piel, apenas ha experimentado avances. La incidencia de la enfermedad ha crecido continuamente en la última década, aumentando año a año. Según los expertos en el área, si el agujero de la capa de ozono sigue creciendo, la incidencia de la enfermedad se incrementará en mayor medida.

Los datos experimentales y los resultados obtenidos en estudios epidemiológicos sugieren que dos son las causas principales de riesgo: exposición solar, y ciertas características genéticas del individuo. De este modo, por ejemplo, la exposición solar continuada representa un factor de riesgo de 9, mientras que la exposición intermitente aguda tiene asociado un ratio de 5,7.

De acuerdo con el registro de cáncer de la Comunidad Autónoma Vasca, el melanoma maligno constituye entre un 8 y un 10 por ciento de los cánceres que afectan a la piel, lo que representa aproximadamente 2,2 personas de cada 100.000.

En los experimentos que hemos realizado disponíamos de una base de datos conteniendo 311 casos diagnosticados en el Instituto Oncológico de Gipuzkoa en el periodo comprendido entre enero de 1988 y diciembre de 1995. Para cada uno de los casos, se dispone de información sobre ocho variables, cinco de las cuales son consideradas predictoras en el sistema clasificatorio. Estas son:

1. Sexo. Dos categorías.
2. Edad recodificada. Cinco categorías.
3. Estadío recodificado. Cuatro categorías.
4. Grosor del tumor. Cuatro categorías.
5. Número de nodos positivos. Dos categorías.

La variable a predecir tiene dos categorías, correspondientes a la supervivencia del paciente, y se consideran los plazos de uno, tres y cinco años desde que se le pronostica melanoma maligno para realizar las predicciones.

9.2.2 Modelos utilizados

Se han tenido en consideración cuatro modelos diferentes a la hora de realizar los experimentos:

1. Búsqueda mediante algoritmos genéticos de la estructura de red Bayesiana que maximiza la métrica de Cooper y Herskovits, sin tener en cuenta la existencia de un orden total entre los nodos (CH-GA).
2. Búsqueda mediante algoritmos genéticos de la estructura *Markov Blanket* de la variable a clasificar que maximice el porcentaje de bien clasificados obtenido (MB-GA), teniendo en cuenta que todas las variables deben formar parte de este manto de Markov.
3. Idéntico al anterior, pero con un *Markov blanket* relajado, en el que no es necesario que todas las variables participen, y además se ponen ciertas restricciones en la estructura para reducir su complejidad (RMB-GA).
4. El modelo naive-Bayes, en el que todas las variables predictoras son hijas de la correspondiente a la clase (NB).

En todos los casos se utiliza la técnica de validación 10-Fold Cross-validation (Stone, 1974). La propagación de la evidencia se ha realizado utilizando el software HUGIN (Andreassen y col., 1989)

En la Figura 9.1 se puede observar la estructura de red Bayesiana obtenida para el caso de supervivencia a cinco año con el modelo CH-GA.

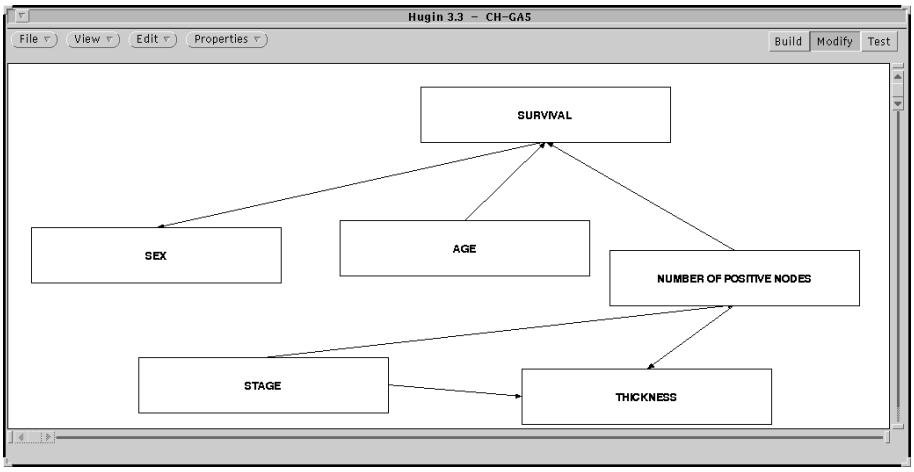


Figura 9.1: La estructura más probable a posteriori encontrada por el algoritmo genético CH-GA para el caso de cinco años.

Se puede apreciar en la Figura 9.1 que se han encontrado, entre otras, las siguientes dependencias e independencias condicionales entre las variables que participan en el modelo:

- I({Estadío recodificado, Grosor del tumor}, Clase | Número de nodos positivos)
- D(Edad, Clase | $\emptyset$)

La Figura 9.2 muestra la estructura de red Bayesiana correspondiente al clasificador NB.

En la Tabla 9.1 se dan las estimaciones de los porcentajes de casos bien clasificados obtenidos por cada uno de los modelos. La Tabla 9.2 muestra la distribución de las clases en los ficheros originales.

Tal y como se aprecia en la Tabla 9.1, el modelo MB-GA es el que mejores resultados (validados) obtiene para los casos de predicción a 1, 3 y a 5 años. Los expertos médicos con los que colaboramos en la realización de este experimento daban mucha importancia a la predicción a 5 años, con lo que el modelo elegido para la predicción en este periodo de tiempo se nos antoja el más interesante de esta experimentación. Destacar el comportamiento del modelo obtenido mediante el paradigma

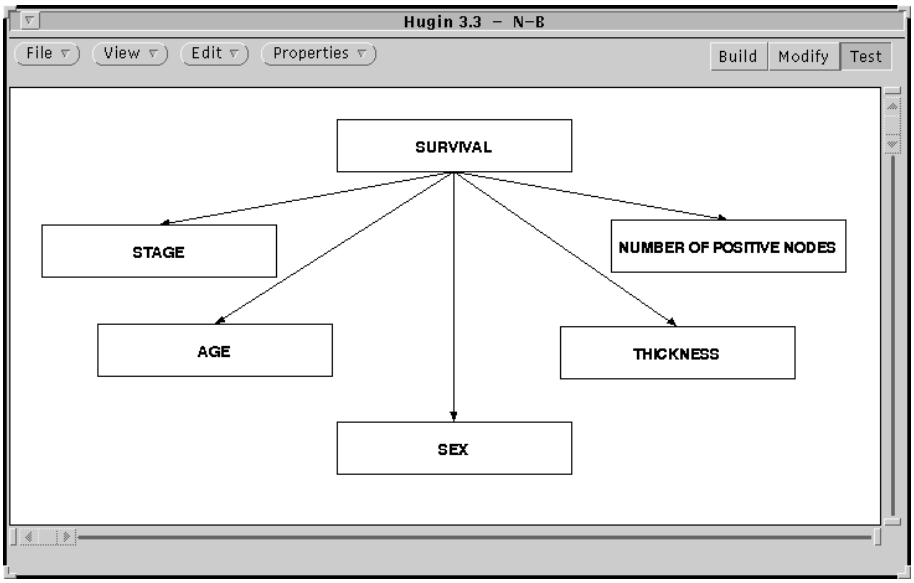


Figura 9.2: El clasificador *naive Bayes*.

Tabla 9.1: Comportamiento de cada uno de los modelos para 1, 3 y 5 años.

Supervivencia Melanoma Maligno			
	1 año	3 años	5 años
CH-GA	93.06	81.95	69.57
MB-GA	94.28	83.90	78.88
RMB-GA	93.47	83.85	74.53
NB	91.43	79.02	71.43

RMB-GA, que resulta ser el segundo mejor en todos los casos, a pesar de la reducción en la complejidad.

En la Tabla 9.2 se aprecian las probabilidades a priori de las diferentes clases en las bases de datos de que se dispone.

Tabla 9.2: Probabilidades a priori de la variable clase para 1, 3 y 5 años.

	Si	No
1 Año	93.06	06.94
3 Años	81.95	18.05
5 Años	67.28	32.72

9.3 Predicción de la supervivencia en pacientes afectas de cáncer de mama

9.3.1 Cáncer de mama

El cáncer de mama, también denominado oncomama, es el más frecuente de los que afectan a la población femenina en Europa. En España, más de un cuarto de los casos de cáncer en pacientes de sexo femenino son de este tipo. De acuerdo al registro de cáncer del País Vasco, (Izarzugaza, 1994), en 1990 el ratio de incidencia en Gipuzkoa era de 52,5 por cada 100.000 personas.

Hay diversos factores de riesgo involucrados en la actividad y patogénesis de la enfermedad. Ninguno de ellos puede, de forma aislada, predecir la apariencia de la misma, siendo la prognosis poco eficiente en nuestros días. Uno de los problemas consiste en la predicción de la supervivencia de los pacientes uno, tres o cinco años después del diagnóstico.

En este experimento analizamos la supervivencia al cáncer de mama en Gipuzkoa, donde, desde 1983, existe un registro de cáncer que recoge información sobre los tipos de tumores que afectan a la población, independientemente de donde sean diagnosticados o tratados. Los casos de cáncer de mama tratados van de enero de 1983 a diciembre de 1988.

La base de datos contiene 1000 casos, y cada uno de ellos contiene la siguiente información sobre los pacientes:

1. Edad recodificada. Cinco categorías.

2. Fase de la enfermedad. Cuatro categorías.
3. Tamaño del tumor. Cuatro categorías.
4. Número de nodos positivos. Cuatro categorías.
5. Supervivencia a 1 año. Dos categorías.
6. Supervivencia a 3 años. Dos categorías.
7. Supervivencia a 5 años. Dos categorías.

9.3.2 Modelos utilizados

Los modelos utilizados son los siguientes:

- CH-GA.
- TAN-GA.
- MB-GA.
- NB.
- Inducción de reglas (CN2).
- Regresión logística.

La Figura 9.3 muestra la estructura de red Bayesiana obtenida por el algoritmo CH-GA para el caso de un año.

La Figura 9.4 muestra el clasificador Naive-Bayes asociado al problema que se está tratando. En este caso la estructura es fija.

Una extensión del clasificador NB consiste en añadir arcos entre los hijos de la variable a clasificar, esto es, entre las variables predictoras, de modo que se tenga en cuenta, en cierta forma, las interdependencias que entre ellas aparecen. En este caso, de nuevo es necesario realizar una búsqueda entre las posibles estructuras TAN. Al realizarse la misma mediante algoritmos genéticos, se obtiene el algoritmo TAN-GA. La Figura 9.5 muestra la estructura devuelta por el algoritmo para el caso de 3 años.

Los otros dos modelos aplicados son algoritmos estándares, uno proveniente de la estadística clásica, denominado *regresión logística*, y que ha sido ejecutado utilizando el paquete estadístico SPSS, y el otro proveniente de *machine learning*, un algoritmo de inducción de reglas denominado CN2, y que ha sido ejecutado utilizando la librería MLC++.

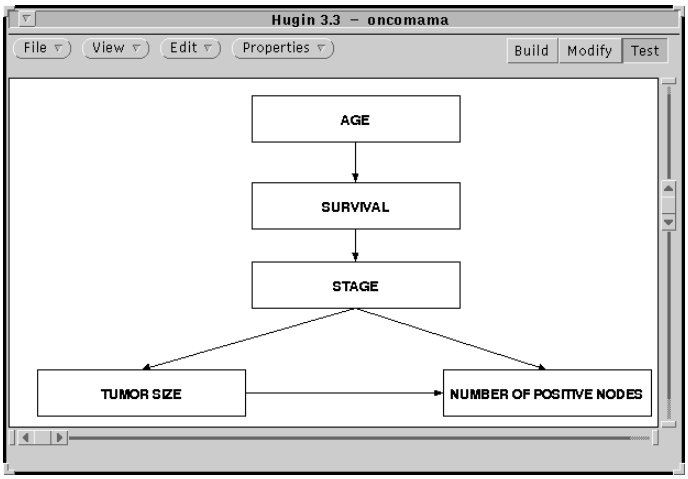


Figura 9.3: La estructura más probable a posteriori encontrada por el método CH-GA.

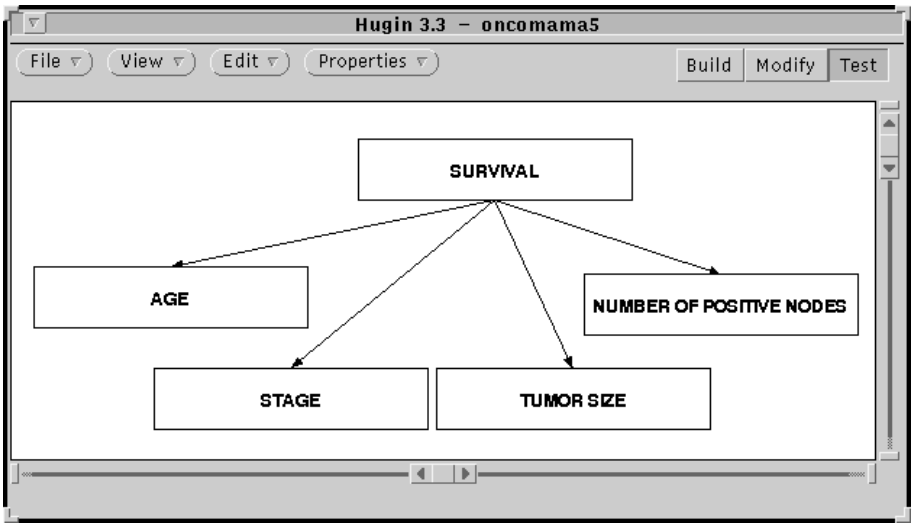


Figura 9.4: Estructura *Naive Bayes* para el problema del cáncer de mama.

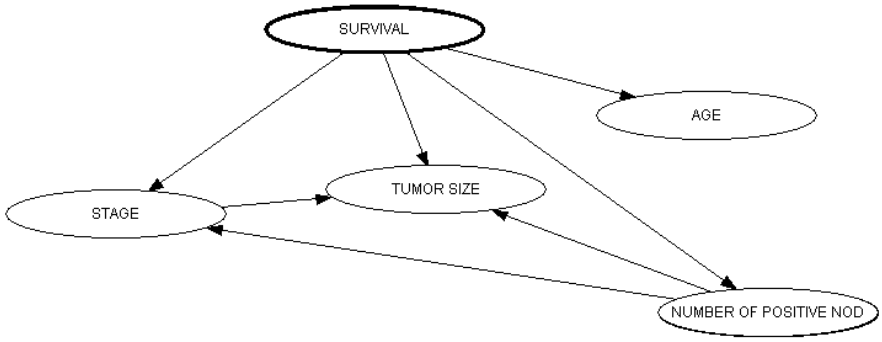


Figura 9.5: Estructura obtenida por el modelo *Tree Augmented Network*, TAN-GA.

En todos los modelos se utiliza la técnica de validación 10-Fold Cross-validation. La Tabla 9.3 muestra los resultados obtenidos por cada paradigma para cada uno de los tres problemas planteados.

Supervivencia Cáncer de Mama			
	1 año	3 años	5 años
CH-GA	94.4	80.4	72.0
TAN-GA	93.7	79.0	70.9
MB-GA	92.0	78.8	71.5
Naive Bayes	93.7	79.0	70.9
Inducción de reglas	93.7	78.9	70.2
Regresión logística	95.0	80.0	69.0

Tabla 9.3: Resultado de cada uno de los modelos para 1, 3 y 5 años.

Es de destacar en estos resultados el buen comportamiento clasificatorio que presenta el modelo inducido por el paradigma CH-GA, a pesar de ser una técnica de aprendizaje estructural que no tiene en cuenta la existencia de una variable especial. Por otro lado, se puede comprobar que los paradigmas utilizados se comportan de un modo muy parecido.

9.4 Seguimiento del estado de pacientes de una Unidad de Cuidados Intensivos (U.C.I.)

9.4.1 Introducción

En el mundo médico, hay varios métodos que se aplican a los pacientes de las Unidades de Cuidados Intensivos (UCI) en el momento de su admisión a las mismas con el objetivo de predecir la supervivencia de los mismos a 24 o a 48 horas. Algunos de los más utilizados son *Acute Physiology and Chronic Health Evaluation II y III* (APACHE II y APACHE III) (Knaus y col., 1985), *Mortality Probability Model II* (MPM II) (Lemeshow y col., 1993), y *Simplified Acute Physiology Score II* (SAPS II) (Le Gall y col., 1993). Dichos métodos parecen estar bien calibrados en la predicción de la supervivencia de pacientes de la UCI. Utilizamos las probabilidades y las medidas devueltas por estos modelos médicos con el ánimo de superar su poder predictivo mediante la combinación adecuada de sus resultados.

Los datos utilizados en este estudio fueron obtenidos en la UCI del Hospital Universitario de Canarias. Disponemos de información sobre 1210 pacientes, en los que se han medido los valores devueltos por los métodos médicos mencionados, además de otro tipo de información relativa a los mismos. Estos datos fueron utilizados con anterioridad por Serrano y col. (1998).

Los experimentos que realizamos consisten, en este caso, en utilizar métodos de *machine learning* para combinar las predicciones dadas por estos modelos del mundo médico, considerándolos como variables predictoras en una base de datos. Obviamente son variables con un gran poder predictivo.

9.4.2 Ficheros de datos

En estos experimentos se han utilizado cuatro ficheros de datos. Todos ellos contienen datos acerca de pacientes de la Unidad de Cuidados Intensivos del Hospital Universitario de Canarias. Las únicas diferencias entre los ficheros utilizados consisten en la utilización o no de la información relativa al paciente junto con los valores de probabilidad dados por los métodos APACHE II, SAPS II y MPM II o los valores devueltos por APACHE II, APACHE III y MPM II. Utilizamos dos ficheros con los valores de probabilidad (uno con datos adicionales del paciente y otro que tiene únicamente los valores de probabilidad como variables predictoras) y otros dos ficheros similares con las medidas que dan como resultado los métodos del mundo médico.

Los datos adicionales que se utilizan son los siguientes:

- Edad.

- Sexo.
- Procedencia: de donde viene el paciente. Puede ser del exterior o del interior del hospital, con varias categorías.
- Admisión: (programada, urgencia, etc.).
- Readmisión: (misma enfermedad, otra distinta, primera vez).
- Causa: (traumatología, coronaria, etc.).
- Días ingreso: cuantos días lleva en el hospital antes de entrar en la UCI.
- Código de diagnóstico: (paro cardíaco, etc.).
- Subcódigo de diagnóstico: información complementaria sobre el diagnóstico.

En la Tabla 9.4 se muestran los resultados clasificatorios de los distintos métodos utilizados por el colectivo médico a la hora de clasificar los casos de la base de datos de que se dispone. Destacar que estos datos no son comparables con los obtenidos por los clasificadores, ya que no están validados, lo que hace suponer que sean demasiado optimistas. Se dan sólo como dato orientativo.

Tabla 9.4: Nivel de comportamiento de los distintos métodos usados por el colectivo médico con diferentes umbrales.

Umbral	*0.35*	*0.40*	*0.45*	*0.50*	*0.55*	*0.60*	*0.65*	*0.70*
APACHE II	79.42	83.14	84.05	85.37	86.45	86.86	87.11	86.44
MPM II	83.64	83.47	83.80	85.12	85.70	85.79	86.61	86.52
SAPS II	78.43	80.00	81.16	82.07	83.06	84.63	85.15	85.62

Se han utilizado diferentes umbrales para ver el comportamiento de estos métodos, ya que no todos predicen de la misma manera. La utilización del umbral indica que se ha utilizado como límite en la clasificación, esto es, los datos que tienen un valor de probabilidad dado por el método menor o igual que el umbral se clasifican como *Supervivencia* y el resto como *No Supervivencia*.

En la Tabla 9.5 se muestran los mismos datos para el método APACHE III. Los umbrales varían debido a que este método devuelve como resultado un valor, en lugar de una probabilidad como ocurre con los anteriores.

En los experimentos que se realizan, se utiliza la técnica de validación 10-Fold Crossvalidation (Stone, 1974). En la Tabla 9.6 se muestran detalles sobre los cuatro ficheros de datos utilizados.

Tabla 9.5: Porcentaje de bien clasificados obtenido por el método APACHE III con diferentes umbrales.

Umbral	*50*	*60*	*70*	*75*	*80*	*85*	*90*
APACHE III	82.89	85.95	87.69	87.27	86.69	86.03	84.04

Tabla 9.6: Características de los ficheros utilizados.

Ficheros de datos	*Casos*	*Métodos del mundo médico*	*Información sobre el paciente*
Sólo probabilidades	1210	APACHE II, SAPS II y MPM II	No
Probabilidades + Datos del paciente	1210	APACHE II, SAPS II y MPM II	Si
Sólo medidas	1210	APACHE II, APACHE III y MPM II	No
Medidas + Datos del paciente	1210	APACHE II, APACHE III y MPM II	Si

Tabla 9.7: Resultados experimentales obtenidos con algoritmos estándar de *machine learning*.

Modelo	*Sólo probabilidades*	*Probabilidades + Datos del paciente*	*Sólo medidas*	*Medidas + Datos del paciente*
ID3	82.07	81.49	81.25	81.07
C4.5	86.12	86.61	86.94	87.11
NB	85.70	85.21	85.87	84.63
NBTree	87.60	84.05	87.77	84.79
IB	81.82	79.34	81.07	78.76
oneR	83.97	86.12	84.28	87.77
CN2	85.65	85.71	84.80	85.94

Los experimentos se han realizado sobre estos cuatro ficheros de datos. La Tabla 9.7 muestra los resultados experimentales obtenidos con paradigmas estándar y validación cruzada.

Como puede observarse, los mejores resultados han sido obtenidos por el clasificador NBTree y por oneR.

9.4.3 Algoritmos de clasificación por vecindad

La experimentación continúa mediante la ejecución de los algoritmos K-NN y D-K-NN con el método de pesado K-NN-LO (pesando mediante el algoritmo Naive Bayes) y los datos discretizados mediante el algoritmo *Discretize* que proporciona la librería MLC++. En la tabla 9.8 se muestran los resultados obtenidos por el algoritmo K-NN-LO para diferentes valores de K.

Tabla 9.8: Porcentaje de bien clasificados obtenido por el algoritmo K-NN-LO para las cuatro bases de datos utilizando diferentes valores de K.

Valor de K	*1*	*2*	*3*	*4*	*5*	*6*	*7*	*8*	*9*	*10*
Sólo probabilidades	82.47	82.72	84.96	84.55	85.29	85.04	84.88	84.88	84.47	85.21
Probabilidades + Datos del paciente	98.31	98.26	98.43	98.35	98.51	98.51	98.51	98.51	98.51	98.35
Sólo medidas	82.31	82.31	82.31	82.31	82.31	82.31	82.31	82.31	82.31	82.31
Medidas + Datos del paciente	97.27	97.36	97.36	97.69	97.93	97.93	98.10	97.77	97.77	97.77

La tabla 9.9 muestra los resultados obtenidos al utilizar el algoritmo D-K-NN, al que se añade la misma técnica de pesado de atributos (D-KNN-LO), con los mismos datos discretizados.

Tabla 9.9: Porcentaje de bien clasificados obtenido por el algoritmo D-K-NN-LO para las cuatro bases de datos utilizando diferentes valores de K.

Valor de K	*1*	*2*	*3*	*4*	*5*	*6*	*7*	*8*	*9*	*10*
Solo probabilidades	83.36	89.58	91.07	91.32	92.23	92.23	91.90	91.65	91.32	91.24
Probabilidades + Datos del paciente	98.43	98.76	98.67	98.35	98.01	97.60	97.19	97.11	96.53	96.53
Solo Medidas	82.21	83.36	88.51	88.51	88.51	88.51	88.51	88.51	88.51	88.35
Medidas + Datos del paciente	97.27	97.44	97.68	97.77	97.85	97.93	97.85	97.93	97.85	96.85

Se puede apreciar que el nuevo método propuesto obtiene resultados mejores que los paradigmas anteriores para este caso de estudio. Quizás la idea subyacente sea que la combinación de diferentes métodos clasificatorios, en este caso, de diferentes aproximaciones utilizadas por el colectivo médico, puede mejorar al mejor de los clasificadores individuales utilizados. Puede observarse que los resultados que se obtienen mejoran al mejor obtenido por los métodos del colectivo médico, incluso al comparar los resultados obtenidos con los que se muestran en las Tablas 9.4 y 9.5, que no están validados.

9.4.4 Resultados experimentales obtenidos con el multiclasificador

El nuevo paradigma de construcción de multiclasificadores presentado en el capítulo anterior, se utilizó también con los datos de los pacientes de la UCI. Con el objetivo de dar una perspectiva más real a los experimentos realizados, se utiliza la técnica de validación 10-Fold Cross-validation (Stone, 1974) en todos los experimentos. Los datos han sido recogidos en la UCI del hospital Universitario de Canarias, y fueron utilizados previamente en otro tipo de análisis médicos (Serrano y col., 1998).

La base de datos utilizada en estos experimentos difiere un poco de la utilizada en la experimentación anterior. Se recogen las variables correspondientes a los métodos estándar de medicina que devuelven como resultado un valor de probabilidad, dejando de lado los que devuelven un score. En la Tabla 9.10 se muestran las variables utilizadas.

Tabla 9.10: Variables de la base de datos UCI.

Variable	*Tipo*	Información
APACHE II	Continua	Método estándar en medicina (probabilidad)
SAPS II	Continua	Método estándar en medicina (probabilidad)
MPM II	Continua	Método estándar en medicina (probabilidad)
Edad	Discreta	Años
Sexo	Discreta	Sexo del paciente
Procedencia	Discreta	Desde donde viene a la UCI
Admisión	Discreta	Fecha
ReAdmisión	Discreta	Fecha
Causa	Discrete	Código interno del Hospital
Días ingreso	Continua	Días en el hospital antes de entrar en la UCI
Código de diagnóstico	Discreta	Código interno del hospital
Subcódigo de diagnóstico	Discreta	Código interno del hospital

Al aplicar el algoritmo genético de aprendizaje de estructuras de red Bayesiana que formen el *Markov blanket* de la variable a clasificar, se obtuvo la estructura que se muestra en la Figura 9.6.

Tal y como se desprende de la Figura 9.6, el aprendizaje estructural ha tenido como resultado una estructura en la que se pueden apreciar, entre otras, las siguientes relaciones de dependencia e independencia condicional entre los clasificadores utilizados:

- I(Ripper, {*El resto de variables*} | {Clase, IB}).

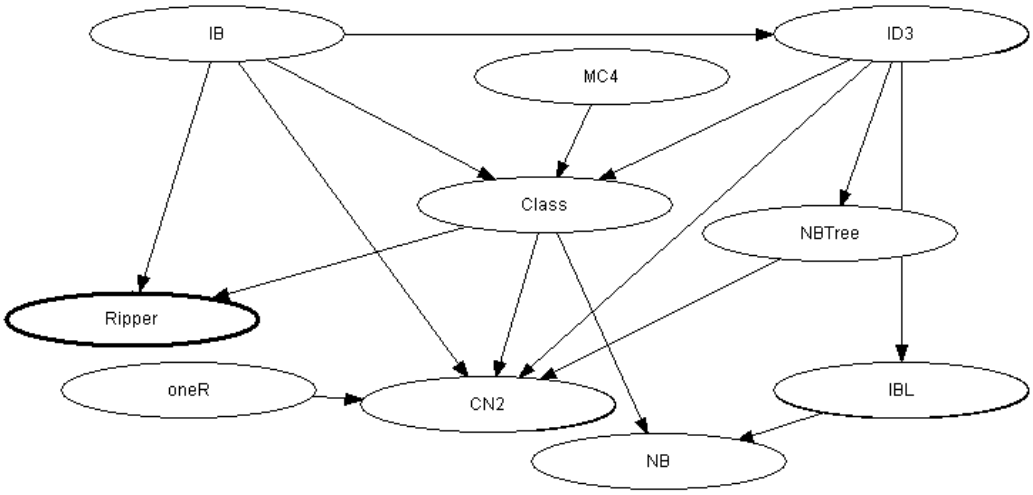


Figura 9.6: Estructura obtenida para la red Bayesiana a partir de los datos.

- I(NB, {*El resto de variables*} | {Clase, IBL}).
- D(MC4, Clase | $\emptyset$).

Hemos realizado los experimentos con esta base de datos utilizando todos los clasificadores de nivel 0. En la Tabla 9.11 se pueden observar los resultados obtenidos por cada paradigma y por el multiclasificador.

Cabe señalar que los mejores resultados (validados mediante la validación cruzada) de entre los métodos estándar de *machine learning* son obtenidos por el paradigma oneR (Holte, 1994). Esto es debido principalmente a que tres de las variables de entrada son resultados de otras técnicas de clasificación, siendo la mejor de ellas, en este caso, la correspondiente al método APACHE II.

Tabla 9.11: Porcentaje de bien clasificados obtenido por los algoritmos estándar de *machine learning* y por el multiclasificador.

Inductor	*Estimación mediante 10-Fold Cross-validation*
ID3	73.64
MC4	79.59
NB	75.64
NBTree	62.64
IB1	64.30
oneR	84.55
CN2	77.52
Ripper	80.81
IBL	63.63
MULTICLASIFICADOR	87.27

Capítulo 10

Problemas reales provenientes del mundo empresarial

10.1 Introducción

Este capítulo presenta la aplicación de métodos provenientes tanto de la Estadística como del Aprendizaje Automático (conocido habitualmente como *machine learning* dentro de la Inteligencia Artificial) en el problema de la predicción del fracaso empresarial. Los paradigmas comparados empíricamente sobre una muestra de 120 empresas españolas (60 de ellas fueron a bancarrota y el resto no) fueron: análisis discriminante, regresión logística, árboles de clasificación, inducción de reglas y redes Bayesianas. Dos técnicas provenientes de la Inteligencia Artificial –el Principio de Voto por la Mayoría y el Formalismo Bayesiano– fueron implementadas para mejorar la predicción de los modelos individuales. Las variables predictoras medidas para cada compañía a lo largo de los tres años previos a la recogida de datos son ratios financieros.

La predicción del fracaso empresarial, a través de la clasificación de casos conocidos y generalizando a otros casos, ha sido tema de estudio en los últimos 30 años. Una certera predicción del fracaso empresarial es importante para inversores, auditores y acreedores. También puede ayudar a los accionistas e incluso al Gobierno a evitar grandes pérdidas mediante el control de la bancarrota. Esto es, a través de los datos de los informes financieros de la empresa y usando herramientas analíticas, se puede llegar a evaluar y predecir el futuro estado financiero de la empresa.

Aunque la idea de la bancarrota está asociada habitualmente con la desaparición de la empresa, antes de que esto llegue a suceder la empresa pasa por un largo período de crisis con varios estados distinguibles. Muchos autores resumen esta serie de estados en dos grandes grupos usando los dos significados del concepto de fracaso: económico y financiero. El fracaso económico empieza cuando la rentabilidad del capital invertido está por debajo de sus costos de oportunidad, esto es, una

inversión en la empresa implica menor rentabilidad que otras alternativas con el mismo riesgo. Según el fracaso económico avanza y se asienta en la compañía, los ingresos comienzan a ser más bajos que los gastos, luego aparecen los primeros resultados negativos.

Si el deterioro producido durante el proceso de fracaso económico no es corregido, este llevará a la compañía a una situación de insolvencia técnica. Este es el primer estadio de lo que es conocido como fracaso financiero. En esta situación la empresa no tiene suficiente capital líquido para hacer frente a los pagos según estos crecen.

Este ruinoso proceso llevará a la compañía a una situación en la que no sólo no puede hacer frente a los débitos sino que se encontrará en una situación negativa de patrimonio neto. Esto significa que sus deudas son mayores que el valor de sus posesiones, y puede conducir a una pronta desaparición de la compañía.

El estudio del fracaso empresarial debe buscar siempre las causas de la situación que nosotros analizamos a través de los síntomas visibles. Tal y como Argenti (1976) propone, es interesante conocer las causas por las que otras compañías fracasaron para así evitarlas en la nuestra. No obstante, el descubrir las causas es imposible si no es a través de los síntomas. Algunas de esas causas son las siguientes: fracaso administrativo, deficiencia en los sistemas de contabilidad, incapacidad para la adaptación a los cambios del entorno, acometer grandes proyectos, abuso de la financiación a través de deudas, el riesgo del mundo empresarial, etc... Para los síntomas, Argenti acepta como significativos el deterioro sufrido por los ratios financieros según la compañía se acerca al fracaso empresarial, y merece ser mencionado que la manipulación de la contabilidad puede ser un signo de fracaso.

El análisis del estado de la contabilidad es parte del proceso de información, cuyo objetivo es la provisión de datos para la toma de decisiones. La idea del fracaso, y más concretamente, la idea de insolvencia ha permanecido conectada a la técnica de ratios contables. Se pensaba que los ratios empeoraban según la compañía avanzaba en el proceso de crisis, de esta forma el deterioro sufrido por la compañía podía ser medido.

Debido a la compleja comprensión de la información de los datos del estado financiero, el análisis de los ratios financieros ha sido la técnica más usada. El gran interés en la comparación entre diferentes compañías (sector industrial, tamaño, etc...) ha influido en su uso. Existen, sin embargo, dos dificultades principales relacionadas con los ratios financieros como son, su creación y su interpretación. Otra dificultad añadida a las anteriores es que el mismo valor de un ratio para dos compañías de diferentes sectores puede representar situaciones diferentes. La información de los ratios debe por tanto ser homogeneizada, así podrá ser usada para la descripción y predicción del fracaso empresarial. La segunda tarea está directamente relacionada con el uso de la Estadística.

Aunque ignoradas durante medio siglo por los analistas, hoy en día el uso de técnicas estadísticas se ha convertido en una provechosa herramienta comúnmente usada, ya que dota al análisis de objetividad. Beaver (1966) fue uno de los pioneros en usar técnicas estadísticas en el análisis de ratios financieros para predecir el fracaso empresarial. En su trabajo, comenzando con 30 ratios tomados de 79 compañías de las cuales unas fracasaron y otras no, 6 ratios fueron seleccionados. Los datos correspondían a los últimos 5 años antes del fracaso. Un análisis de los ratios está basado en la comparación de las medias de los valores para dichos ratios en cada grupo (fracaso y no fracaso) y en observar las diferencias existentes entre los mismos.

Sin embargo el modelo univariable de Beaver contrasta con el carácter multivariable inherente a la información de la situación financiera. Por lo tanto, para valorar correctamente la información arriba mencionada, esta debe ser interpretada desde una perspectiva que permita pensar sobre varios aspectos financieros de la empresa como un todo. La búsqueda de esta perspectiva ha sido la razón por la cual muchos investigadores han usado técnicas estadísticas multivariables para la predicción del fracaso empresarial.

Altman (1968) fue el pionero en la aplicación del análisis discriminante al problema anteriormente mencionado, obteniendo resultados sorprendentes. La combinación lineal de cinco ratios constituyó un predictor capaz de discriminar entre estados *saludables* y *fracasos* con un alto porcentaje de éxito en los dos años previos al fracaso.

La necesidad de una alternativa estadística para evitar los problemas relacionados con el análisis discriminante condujeron al uso de modelos basados en la probabilidad condicional, logit y probit, con requerimientos más flexibles. Ohlson (1980) está considerado el primer autor que publicó un método para la predicción del fracaso empresarial basado en modelos que usaban la probabilidad condicional.

10.2 Descripción del problema

Comenzando con la hipótesis de que el patrón de la información contable de las empresas que responden a fracasos y no fracasos es diferente, el objetivo fundamental de este capítulo es mostrar mediante un ejemplo como crear modelos capaces de predecir con antelación (1 año, 2 años y 3 años) el fracaso empresarial. Estos modelos podrían ser considerados sistemas normativos ya que están basados en una teoría matemática sólida, en este caso la teoría probabilística.

Siguiendo el progresivo desarrollo en Inteligencia Artificial, dos técnicas han sido implementadas y usadas para la integración de modelos individuales en uno, mejorando así la capacidad predictiva individual.

La muestra de datos usada es la misma que Lizarraga (1996) recogió en distintas Oficinas de Registros Mercantiles para la comparación que llevó a cabo en su tesis doctoral. A continuación se explica como fue obtenida la muestra de 120 compañías.

La necesidad de determinar el concepto de fracaso a usar fue el primer problema metodológico a resolver. Finalmente, se eligió el concepto de suspensión de pagos, ya que está relacionado no con un problema financiero específico sino con una situación de profunda crisis económica. Este concepto presenta tres ventajas fundamentales: objetividad, una fecha significativa del momento de fracaso y un gran número de empresas que podían formar parte de la muestra. Por último, la disponibilidad de la contabilidad anual depositada en las distintas Oficinas del Registro Mercantil Provincial fue otro de los aspectos que ayudó en la obtención de la información.

La muestra era de 120 compañías, la mitad de las cuales pertenecían al grupo de compañías clasificadas como *fracasos* y la otra mitad al grupo clasificada como *saludables*. La selección fue llevada a cabo mediante un proceso de emparejamiento. Seleccionando una lista de *fracasos*, se emparejó cada una de las empresas de esa lista con otra del mismo tamaño y sector industrial perteneciente al grupo de las empresas denominadas *saludables*. Dado que el acceso a todas las Oficinas del Registro Mercantil Provincial no era posible, Lizarraga decidió ceñirse a las 10 provincias con mayor número de registros de suspensión de pagos registrados durante el período de estudio. La información fue obtenida a través del Boletín Oficial de la Oficina del Registro Mercantil. Para cada compañía del estudio fueron obtenidos los datos económicos y financieros de los últimos 3 años.

10.3 Resultados experimentales

En esta sección se explican los resultados de aplicar los métodos mencionados al problema del fracaso empresarial. Se mostrarán los modelos para cada uno de los cinco paradigmas usados en este trabajo, y para un periodo de tiempo anterior al fallo (1 año, 2 años y 3 años). Los resultados son representados en la Tabla 10.1, mediante el porcentaje de compañías bien clasificadas para cada paradigma y periodo.

En este caso es interesante mencionar el buen comportamiento de paradigmas como el CN2 y las redes Bayesianas, como también los árboles de clasificación.

El objetivo de cualquier algoritmo diseñado para integrar los resultados de clasificadores individuales es generar resultados más certeros, precisos y exactos. Dos experimentos fueron llevados a cabo para comparar el rendimiento de los métodos individuales mencionados y las técnicas de combinación. La Tabla 10.2 presenta los resultados obtenidos aplicando el método de combinación

Tabla 10.1: Resultados, validados por 5-*fold cross-validation*, obtenidos por los clasificadores individuales.

Años antes del fracaso	*AD*	*RL*	*CART*	*CN2*	*Redes Bayesianas*
1	78.33	82.50	79.17	80.00	60.83
2	69.17	69.16	60.00	66.66	62.00
3	55.00	55.00	45.00	57.50	60.83

Tabla 10.2: Resultados obtenidos según el Principio de Voto por la Mayoría validados mediante 5-*fold cross-validation*.

Años antes del fracaso	*Voto por la Mayoría*
1	88.33
2	79.17
3	73.33

Principio de Voto por la Mayoría a los resultados de los clasificadores individuales, usando como método de validación el 5-fold cross-validation.

Como era de esperar el Principio de Voto por la Mayoría mejoró los resultados de los clasificadores individuales, siendo así mejor que el mejor clasificador individual para cualquiera de los 3 años. Estos buenos resultados pueden ser comparados con los que se obtuvieron aplicando el Formalismo Bayesiano como método de combinación, del cual los resultados son presentados en la Tabla 10.3.

Tabla 10.3: Resultados obtenidos con el formalismo Bayesiano validados con 5-*fold cross-validation*.

Años antes del fracaso	*Formalismo Bayesiano*
1	78.33
2	75.83
3	62.50

Como se desprende de la Tabla 10.3, el formalismo Bayesiano presenta mejores resultados que el mejor de los clasificadores individuales para los años 3 y 2 antes del fracaso, pero en el caso de 1 año antes del fracaso sólo mejora al peor de los clasificadores individuales.

Capítulo 11

Problemas estándares

11.1 Introducción

En el mundo de la clasificación supervisada existen varios conjuntos o repositorios de casos de ejemplo que son utilizados por los investigadores del área para realizar comparaciones relativas al comportamiento de los diferentes algoritmos que se proponen.

11.2 Experimentos con el algoritmo *Diplomatic K Nearest Neighbour*

Se ha aplicado el nuevo método D-K-NN, presentado en el Capítulo 6, a diferentes ficheros estándar, y se han comparado los resultados obtenidos con los que se logran al aplicar a los mismos datos algoritmos estándar de *machine learning*.

11.2.1 Clasificadores estándar de *machine learning*

Se han elegido diez paradigmas provenientes de diferentes familias del mundo de la clasificación supervisada (Dietterich, 1997) con el objeto de comparar los resultados que obtienen con los resultados del método D-K-NN:

- *ID3*. Algoritmo para la inducción de árboles de clasificación (Quinlan, 1986). No se realiza la poda del árbol expandido.
- *C4.5*. Árboles de clasificación (Quinlan, 1993). Realiza la poda en base a un algoritmo denominado *error based pruning*.

- *OC1.* Árbol de clasificación oblicuo (Murthy y Salzberg, 1994). Construye un hiperplano que tiene en cuenta todas las variables predictoras en cada uno de los nodos, separando instancias de clases diferentes.
- *T2.* Árbol de clasificación de dos niveles (Auer y col., 1995). Requiere una gran cantidad de memoria para problemas con un número no muy grande de clases.
- *Naive Bayes (NB).* (Cestnik, 1990). Asume independencia entre las variables predictoras dada la clase.
- *Naive Bayes Tree (NBTree).* Algoritmo híbrido (Kohavi, 1996). Construye un árbol de clasificación en el cual las hojas no especifican una única clase, sino que se decide la clase a predecir de un caso que llega a las mismas mediante el algoritmo NB.
- *IB.* Inductor basado en casos implementado a partir de los trabajos de Aha (1992) y de Wettschereck (1994). *IB* es un algoritmo similar al IB4 de Aha (1991).
- *PEBLS.* Inductor basado en casos (Cost y Salzberg, 1993). Incorpora una métrica especial para manejar datos simbólicos.
- *OneR.* Algoritmo que genera como salida una sola regla de clasificación, basada además en un solo atributo (Holte 1993).
- *CN2.* Algoritmo de inducción de reglas, basado en el trabajo de Clark y Nibblet (1989). Utiliza test estadísticos para inducir las reglas.

11.2.2 Bases de datos

Se han utilizado siete bases de datos para realizar los experimentos. Seis de ellas se han obtenido de la página WEB del proyecto Statlog, en el *UCI Machine Learning Repository* (Murphy, 1994). La otra, denominada Nettalk, se encuentra en el repositorio asociado a la librería MLC++ (Kohavi y col., 1997). Todas las bases de datos tienen por un lado una serie de casos de entrenamiento y por el otro una serie de casos de test. En la Tabla 11.1 se muestran las características de las siete fuentes de datos utilizadas.

Se utilizan en todos los casos las bases de datos de entrenamiento para generar el modelo clasificatorio, y los datos de test para establecer la potencia clasificadora del modelo inducido. En la Tabla 11.2 se pueden ver los resultados experimentales obtenidos utilizando paradigmas estándares de *machine learning*.

En los experimentos realizados, se ha ejecutado el algoritmo D-K-NN con diferentes valores del parámetro K, y comparado los resultados obtenidos con los que se muestran en la tabla anterior.

Tabla 11.1: Características de las bases de datos utilizadas.

Dominio	*Casos de entrenamiento*	*Casos de test*	*Clases*	*Atributos*
Iris	100	50	3	4
Letters	15.000	5.000	26	16
Nettalk	7.229	7.242	324	203
Pima	200	332	2	7
Satimage	4.435	2.000	7	36
Shuttle	43.500	14.500	7	9
Vote	300	135	2	16

Tabla 11.2: Porcentajes de bien clasificados obtenidos en el conjunto de testeo por los algoritmos estándares de *machine learning*. CRASH indica que el algoritmo no fue capaz de abordar el problema.

Inductor	*Iris*	*Letters*	*Nettalk*	*Pima*	*Satimage*	*Shuttle*	*Vote*
ID3	94.00	76.65	72.52	71.71	84.80	99.99	94.17
C4.5	92.00	75.57	71.50	75.39	85.40	99.95	97.04
OC1	96.00	68.30	66.13	75.78	86.25	99.94	96.30
T2	94.00	40.35	17.93	73.56	CRASH	*CHASH*	96.29
NB	96.00	73.97	60.99	78.01	79.65	87.61	91.85
NBTree	96.00	84.30	65.85	78.91	81.65	99.98	95.53
IB	96.00	88.70	62.56	69.75	88.80	99.93	94.81
PEBLS	96.00	95.30	70.50	71.50	87.90	*CRASH*	97.00
oneR	94.00	16.67	12.48	73.82	58.80	94.67	97.04
CN2	94.00	64.00	*CRASH*	75.80	71.30	99.40	95.60

Tabla 11.3: Porcentaje de bien clasificados obtenido en el conjunto de testeo por el algoritmo D-K-NN para distintos valores de K.

Valor de K	*Iris*	*Letters*	*Nettalk*	*Pima*	*Satimage*	*Shuttle*	*Vote*
1	96.00	95.62	51.95	68.37	89.45	**99.88**	**99.33**
2	96.00	96.02	60.72	73.80	90.65	99.86	92.59
3	**98.00**	96.28	64.25	75.90	90.85	99.85	92.59
4	98.00	**96.28**	65.24	**76.51**	90.60	99.85	93.33
5	96.00	96.24	66.24	75.90	**90.75**	99.85	93.33
6	96.00	95.96	66.29	75.90	90.55	99.82	93.33
7	98.00	95.80	**66.42**	75.90	90.15	99.79	92.59
8	96.00	95.76	66.11	75.90	90.00	99.79	93.33
9	92.00	95.42	65.71	75.90	89.95	99.77	93.33
10	92.00	95.14	65.67	76.20	90.00	99.77	93.33

En la Tabla 11.3 se muestran los resultados obtenidos utilizando esta nueva técnica. Obviamente, cuando K vale uno, el método es el 1-NN o, simplemente, NN.

En los resultados obtenidos se puede apreciar que el nuevo algoritmo propuesto tiene un buen comportamiento como clasificador, superando en algunas de las bases de datos al resto de paradigmas con los que ha sido comparado en este experimento para, algún valor del parámetro K, (ficheros Iris, Letters y Satimage) y manteniendo un buen nivel de aciertos con el resto de bases de datos.

11.3 Resultados obtenidos por el multiclasificador sobre ficheros estándares

Existen dos formas de abordar un problema con el paradigma multiclasificador presentado en esta tesis, dependiendo de si la base de datos de la que se dispone contiene un número elevado de casos o no. Si son pocos casos, se utiliza la técnica de validación *leave one out* para obtener una segunda base de datos, necesaria para realizar el aprendizaje estructural de la red Bayesiana. En el caso de que los datos sean suficientes, una segunda forma de abordar el problema es dividir la base de datos de entrenamiento en dos, DBE1 y DBE2, de modo que la primera de ellas (DBE1) sirva para aprender los modelos de los paradigmas que forman la primera capa, esto es, los nueve clasificadores de nivel 0, mientras que la segunda (DBE2) se utiliza para generar, con los resultados obtenidos por estos clasificadores sobre los casos que la componen, tomándolos como valores de

las variables asociadas al clasificador correspondiente, la base de datos necesaria para aprender la estructura de la RB.

Para esta segunda posibilidad utilizamos dos ficheros de datos:

1. *Wave-21.*
   En total, dispone de 5.000 casos, con los que realizamos la siguiente división: los primeros 1.000 casos se utilizarán como primera base de datos de entrenamiento, los segundos 1.000, como segunda base de datos, esto es, como la base de datos que, al ser utilizada como test por los modelos inducidos con la primera, sirve para generar las columnas que formarán la base de datos de entrenamiento para la RB; los últimos 3.000 casos forman la base de datos de test.

2. *Chess.*
   En total, dispone de 3.196 casos, con los que realizamos la siguiente división: los primeros 1.065 casos se utilizarán como primera base de datos de entrenamiento, los segundos 1.065, como segunda base de datos, y los últimos 1.066 casos forman la base de datos de test.

11.3.1 Resultados

Hemos realizado los experimentos con estas bases de datos utilizando todos los clasificadores de nivel 0. En la Tabla 11.4 se pueden observar los resultados obtenidos por cada paradigma utilizando, con la base de datos *Wave-21* los 2.000 primeros casos como datos de entrenamiento y los últimos 3.000 como casos de test, y con la base de datos *Chess* los primeros 2.130 casos para entrenar los modelos y los 1.066 restantes para testearlos. El motivo de mostrar estos datos es que el hecho de dividir la base de datos de entrenamiento para construir el multiclasificador no implica que en el resto de paradigmas sea necesario hacerlo, con lo que dispondrán de todos los casos de entrenamiento para inducir el modelo. Los resultados que se muestran indican el número de casos acertados, en vez del porcentaje de aciertos como venía siendo habitual en el resto de las Tablas. Ello es debido a que en ambos ficheros se obtienen muy buenos porcentajes con los clasificadores individuales, con lo que la mejora porcentual es muy pequeña.

Tal y como se aprecia en la Tabla 11.4, para la primera base de datos, el mejor de los paradigmas individuales es NBTree, con un acierto en 2.435 casos de los 3.000 de test en la base de datos *Wave-21*, mientras que para la segunda base de datos utilizada el modelo que mejor comportamiento presenta es el árbol de clasificación aprendido mediante C4.5, el cual clasifica bien 1.061 sobre los 1.066 del fichero de test de la base de datos *Chess*. Por otro lado, el peor, en este caso, resulta ser oneR, con un acierto en 1.614 y 723 casos respectivamente. Ello puede ser debido a que todas las variables predictoras (20 en *Wave-21* y 35 en *Chess*) aportan información, y no ocurre como

Tabla 11.4: Número de casos bien clasificados obtenido por los paradigmas de *machine learning* en los casos test.

Inducer	*Wave-21*	*Chess*
ID3	2.231	1.052
C4.5	2.244	1.061
NB	2.431	929
NBTree	2.435	1.059
IB1	2.250	1.013
oneR	1.614	723
CN2	2.016	1.046
Ripper	2.338	1.046
IB4	2.250	1.010

en una experimentación anterior realizada con el multiclasificador, en el que una de ellas tenía un gran poder clasificatorio por si sola, al ser la asociada a un clasificador utilizado en el mundo médico.

Con estos modelos inducidos se obtiene una segunda base de datos que consta de 10 columnas. Las primeras nueve de ellas se corresponden con los resultados que obtiene cada uno de los paradigmas de nivel 0 en cada uno de los segundos 1.000 y 1.065 casos respectivamente, mientras que la décima columna indica la clase real a la que estos casos pertenecen. Como se puede apreciar, esta base de datos tiene tantas variables predictoras como clasificadores utilizados en el nivel 0, independientemente de las variables predictoras de la base de datos original (21 y 35 en los casos estudiados).

Esta segunda base de datos se utiliza para aprender la estructura de la RB mediante el modelo MB-GA, esto es, buscando mediante algoritmos genéticos el *Markov blanket* de la variable a clasificar que maximiza el porcentaje de casos bien clasificados en la base de datos de entrenamiento. En la Figura 11.1 se muestra la estructura obtenida para la base de datos *Wave-21*.

En la Tabla 11.5 se muestra el número de casos que acierta el multiclasificador para los casos de test de ambas bases de datos.

Se puede observar que en ambos casos el multiclasificador supera el número de casos bien clasificados obtenido por el mejor de los clasificadores individuales. En el caso de la base de datos *Chess*, el incremento de aciertos es mínimo (1.062 frente a 1.061), pero hay que tener en cuenta

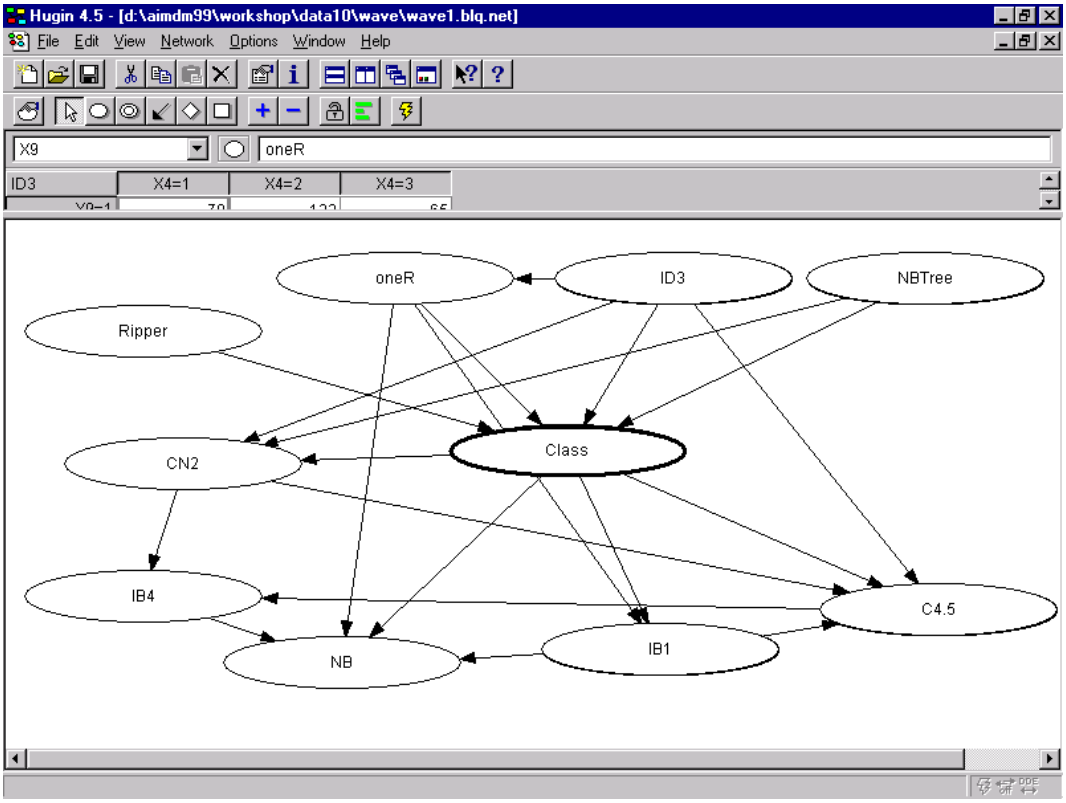


Figura 11.1: Estructura *Markov blanket* inducida para el problema *Wave-21*.

Tabla 11.5: Número de casos bien clasificados obtenido por el multiclasificador para los casos de test.

Inducer	*Wave-21*	*Chess*
MULTICLASIFICADOR	2.477	1.062

que el porcentaje de aciertos es muy elevado, y que un incremento a estos niveles de acierto es más difícil de conseguir.

11.4 Redes Bayesianas

Se han realizado experimentos utilizando las redes Bayesianas como paradigmas de clasificación supervisada con ficheros estándar del repositorio de la Universidad de Irvine, UCI (Murphy, 1994). En Sierra y Larrañaga (1997b) se presentan con más detalle los resultados obtenidos.

11.4.1 Bases de datos utilizadas

Se han elegido cinco bases de datos del repositorio, con el objetivo de realizar experimentos para comprobar el poder clasificatorio de las redes Bayesianas. Las bases de datos son:

1. *crx*: Datos sobre tarjetas de crédito.
2. *flare*: Datos sobre llamaradas solares.
3. *heart*: Datos sobre enfermedades del corazón.
4. *hepatitis*: Datos sobre pacientes de hepatitis.
5. *pima*: Datos sobre pacientes de diabetes.

En la Tabla 11.6 se muestran las características de las bases de datos utilizadas en los experimentos.

Tabla 11.6: Bases de datos utilizadas.

Base de Datos	Casos	Variables
crx	690	14
flare	1067	12
heart	180	13
hepatitis	155	19
pima	768	8

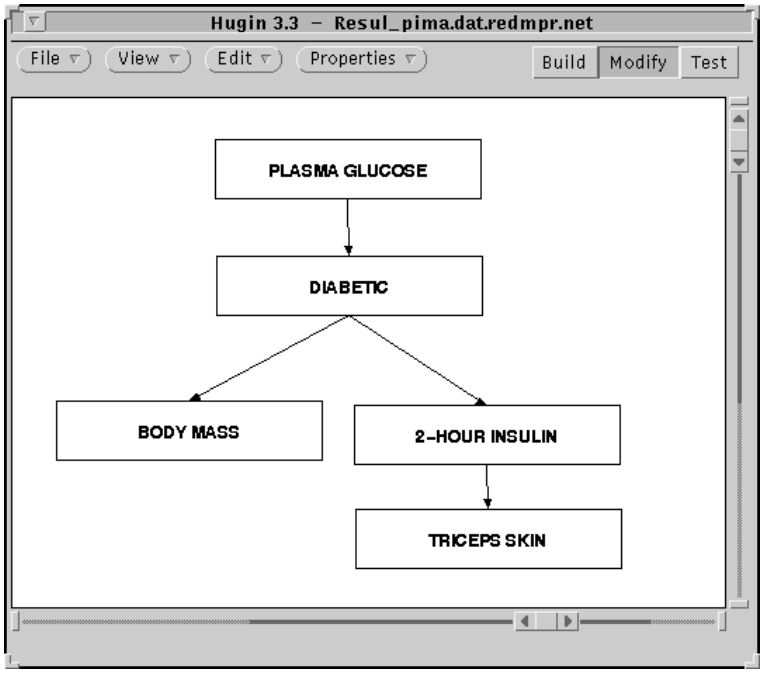


Figura 11.2: Estructura más probable a posteriori encontrada para el ejemplo *pima* mediante el método CH-GA. El resto de variables no están conectadas a la clase.

11.4.2 Resultados obtenidos

Los cuatro modelos aplicados a todas las bases de datos son: CH-GA, TAN-GA, MB-GA y NB.

En la Figura 11.2 se muestra la estructura de red Bayesiana encontrada al tratar de encontrar la estructura más probable a posteriori para el fichero de datos *pima* mediante la aproximación CH-GA.

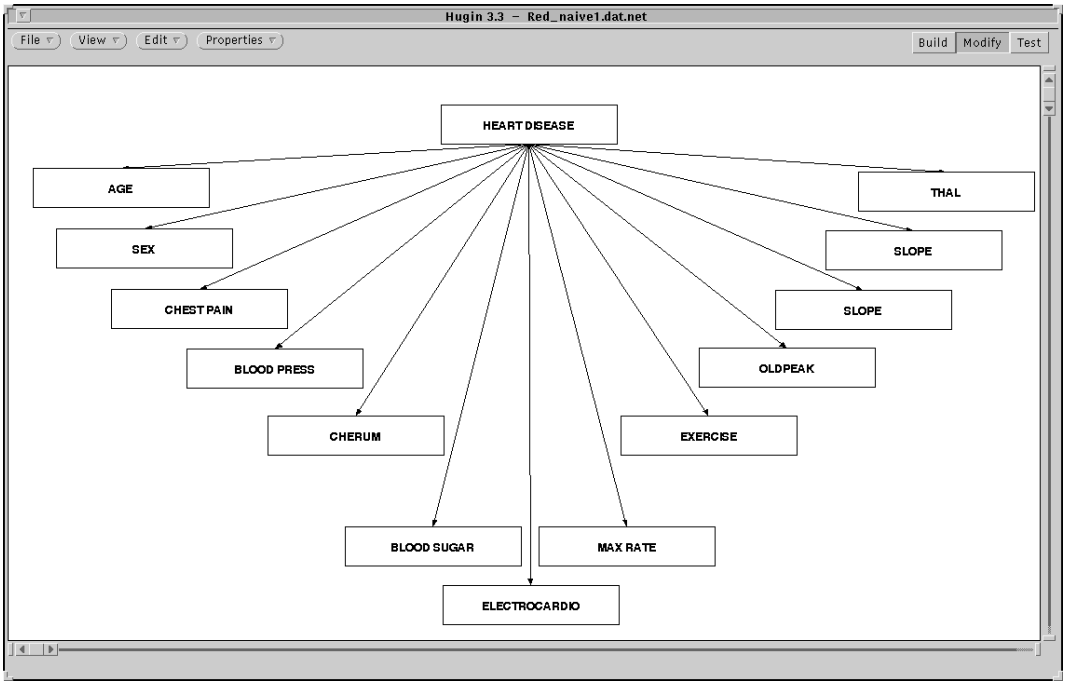


Figura 11.3: Estructura *naive Bayes* para el ejemplo *heart*.

La Figura 11.3 presenta la estructura naive-Bayes para el fichero *heart*, mientras que la Figura 11.4 muestra la estructura obtenida por el modelo TAN-GA para el mismo fichero.

La Figura 11.5 muestra la estructura obtenida tras la aplicación del algoritmo MB-GA para la base de datos *pima*.

Se ha aplicado un proceso estándar de discretización de variables en base a los percentiles de los valores de las variables, habiéndose discretizado todas aquellas variables que en el fichero original tenían más de seis valores diferentes.

En la Tabla 11.7 se pueden ver los porcentajes de aciertos estimados para cada uno de los modelos, estimado con la técnica de validación 10-fold cross-validation (Stone 1974).

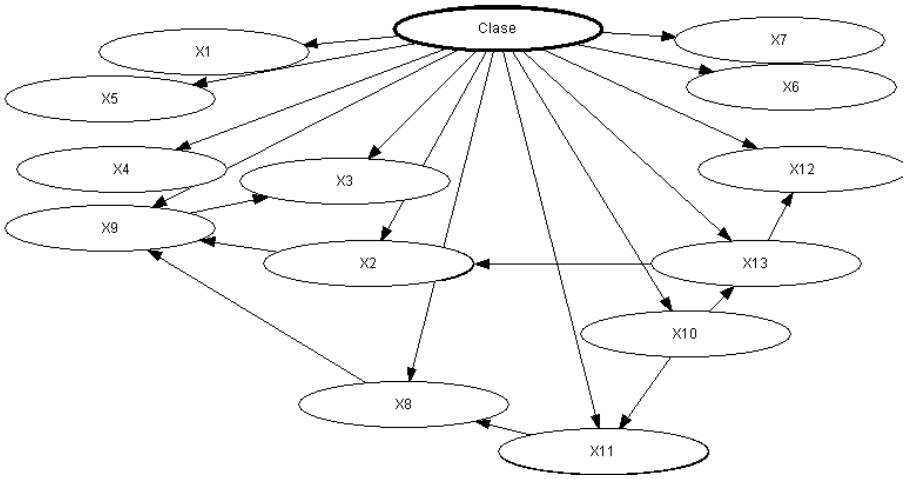


Figura 11.4: Estructura obtenida por el algoritmo TAN-GA para la base de datos *heart*.

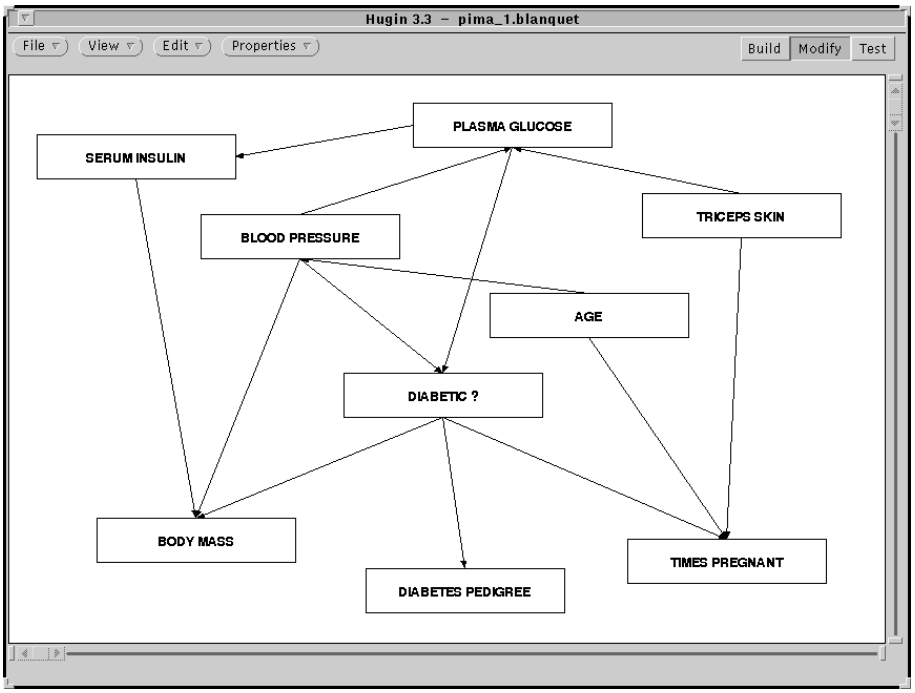


Figura 11.5: Mejor *Markov blanket* encontrado por MB-GA para el ejemplo *pima*.

Tabla 11.7: Porcentaje de bien clasificados estimado para cada modelo.

Archivo	CH-GA	NB	TAN-GA	MB-GA
crx	90.51	85.41	88.51	80.70
flare	81.24	82.27	83.11	80.96
heart	75.56	87.04	84.44	68.15
hepatitis	95.00	86.25	85.00	88.75
pima	69.79	77.86	76.43	60.29

Tal y como se aprecia en la Tabla 11.7, y en contra de lo que se podía esperar, el comportamiento del modelo basado en la métrica de Cooper y Herskovits como clasificador es el que mejor resultados presenta para los ficheros *crx* y *hepatitis*. El modelo NB funciona muy bien en los problemas planteados, siendo el que más acierta en los ficheros *heart* y *pima*. El modelo TAN también presenta un buen comportamiento –el mejor en el problema *flare*–, mientras que el modelo MB, no responde a las expectativas creadas, siendo el peor en cuatro de los cinco casos de estudio. Hay que decir en su defensa que en los experimentos realizados con problemas reales si que mejoraba los resultados de los modelos anteriores.

11.5 Selección de prototipos

Se han realizado experimentos para la utilización de *algoritmos genéticos* y *algoritmos de estimación de distribuciones*, junto con algoritmos provenientes de la clasificación no supervisada, en la selección de prototipos para el algoritmo K-NN.

Se ha elegido como fichero de ejemplo uno de los que se encuentran en el repositorio UCI (Murphy, 1994), concretamente el denominado *letters*, que por su tamaño –15.000 casos de entrenamiento y 5.000 de test– y sus características –16 variables predictoras, cada una de las cuales puede tener 16 valores diferentes, y 26 clases– no es fácil de abordar en su totalidad mediante un algoritmo evolutivo.

Se ha utilizado la siguiente metodología con cada uno de los algoritmos evolutivos:

1. Separar el fichero de entrada en 16 subconjuntos disjuntos mediante el algoritmo K-Means. (Los subconjuntos son los mismos en cada uno de los dos algoritmos evolutivos utilizados).

2. Aplicar a cada uno de los subconjuntos obtenidos los algoritmos evolutivos, obteniendo como resultado un conjunto de prototipos de cada uno de ellos para cada algoritmo utilizado.

3. Juntar los 16 subconjuntos de prototipos en uno.

4. Comprobar el resultado que se obtiene al aplicar el algoritmo K-NN sobre el conjunto de testeo utilizando como conjunto de entrenamiento (o como modelo, al ser el algoritmo K-NN) el subconjunto total obtenido.

En la Tabla 11.8 se muestra el número de casos de cada uno de los cluster, así como el número de prototipos que de ellos se seleccionan bien mediante el uso de un algoritmo genético, bien mediante el algoritmo PBIL de la familia EDA.

Tabla 11.8: Casos totales, seleccionados por el algoritmo genético y seleccionados por el algoritmo PBIL para cada subconjunto (Cluster).

Cluster	Casos totales	Prototipos AG	Prototipos PBIL
1	1633	326	241
2	971	232	219
3	817	195	81
4	505	106	51
5	237	45	19
6	666	122	65
7	795	162	74
8	1311	372	261
9	1491	346	435
10	639	119	66
11	1489	374	265
12	1091	283	167
13	835	229	99
14	1212	300	292
15	680	135	73
16	628	161	81

La Tabla 11.9 muestra el porcentaje de casos bien clasificados obtenido mediante los conjuntos de prototipos resultantes de unir todos los cluster, junto con el porcentaje de aciertos que se obtiene al utilizar los 15.000 casos de la base de datos de entrenamiento como modelo.

Tabla 11.9: Porcentaje de casos bien clasificados obtenido para cada conjunto de datos.

Archivo	Todos	Prototipos AG	Prototipos PBIL
Letters	95.52	82.30	76.60

Los resultados obtenidos son bastante pobres, entre otras cosas debido a que el fichero de datos elegido es de complicada compresión, ya que cada una de las 16 variables predictoras puede tener 16 valores distintos, lo que hace que el número de casuísticas a considerar sea muy grande, y las fronteras de decisión abundantes.

Se han hecho ejecuciones con ficheros más pequeños, en los que no es necesario realizar la partición con el algoritmo K-Means. Los resultados obtenidos, junto con las características de los ficheros utilizados, pueden verse en la Tabla 11.10.

Tabla 11.10: Prototipos seleccionados y número de casos bien clasificados obtenido con ellos en el conjunto de test.

	Todos			Selección AG		Selección PBIL	
Archivo	Casos totales	Casos test	Aciertos	Prototipos	Aciertos	Prototipos	Aciertos
Iris	100	50	47	17	47	14	45
Vote	300	135	128	59	126	28	123
Wine	118	60	56	22	54	13	56
Australian	460	231	188	91	189	31	185

Tal y como se aprecia en la Tabla 11.10, los resultados obtenidos indican que se puede reducir el conjunto de entrenamiento mediante estas técnicas, manteniendo un nivel aceptable de aciertos. Mediante PBIL se reduce más el tamaño del conjunto de entrenamiento, al menos en estos cuatro ficheros, siendo el nivel de aciertos un poco inferior al logrado mediante los algoritmos genéticos.

Parte VII

CONCLUSIONES Y LÍNEAS DE TRABAJO FUTURAS

Capítulo 12

Conclusiones y líneas de trabajo futuras

12.1 Introducción

Como finalización se realiza un repaso a las aportaciones realizadas en el desarrollo de esta tesis, indicándose para cada una de ellas las conclusiones que se han obtenido y líneas que se han dejado abiertas.

12.2 Aportaciones

Las aportaciones principales a nivel metodológico presentadas en esta memoria son las siguientes:

1. *Redes Bayesianas*
    - Nuevos métodos de aprendizaje de RRBB con objetivo clasificatorio, que tienen en cuenta la existencia de una variable especial, que es la correspondiente a la clase.
        - Naive Bayes.
        - Naive Bayes con agrupaciones de variables.
        - Tree Augmented Network.
        - Markov Blanket.
        - Estructura general.
2. *Algoritmos basados en vecindad*

- Nuevo algoritmo de la familia, Diplomatic K Nearest Neighbour, que busca las K instancias más cercanas en cada una de las clases consideradas, y asigna la clase con menor distancia media.
- Nuevo algoritmo de la familia, Probabilistic Weighted K Nearest Neighbour, que pondera los casos según su tipicidad. La tipicidad la obtenemos mediante una red Bayesiana.
- Tercer nuevo algoritmo de la familia, K-NN Clases, en el que cada caso aporta su clase y la de sus vecinos.
- Selección de prototipos mediante algoritmos genéticos y algoritmos de estimación de distribuciones.
- Métodos K-NN-LO y K-NN-UN de pesados de atributos. Son técnicas envolventes que utilizan el propio algoritmo basado en vecindad para pesar los distintos atributos.

3. *Clasificadores múltiples*

   - Nuevo multiclasificador de dos capas con una red Bayesiana en la segunda, siendo esta la que toma la decisión final sobre la clasificación a asignar.

Por otro lado se han presentado aportaciones a nivel de aplicación en problemas del mundo real, tanto en problemas médicos como en empresariales.

12.3 Conclusiones

En el trabajo desarrollado en esta tesis se ha hecho un estudio de las técnicas principales utilizadas en el mundo de la clasificación supervisada, y se han presentado diversos métodos novedosos para abordar el problema. Las conclusiones obtenidas a partir de los experimentos efectuados podemos resumirlas en lo siguiente:

1. *Redes Bayesianas en clasificación supervisada*
   Se ha visto que las RRBB constituyen un paradigma adecuado para abordar problemas de clasificación supervisada, dada su capacidad para aprender las interrelaciones intrínsecas entre las variables que componen el modelo.

2. *Algoritmos de clasificación por vecindad*

   - Algoritmo Diplomatic K Nearest Neighbour.
   - Algoritmo Probabilistic Weighted K Nearest Neighbour.
   - Algoritmo K Nearest Neighbour Clases.

- Métodos de pesado de atributos K-NN-LO y K-NN-UN.
- Selección de prototipos mediante algoritmos genéticos y algoritmos de estimación de distribuciones.

3. Nuevo multiclasificador bicapa.

Como conclusiones se han obtenido la posibilidad de aplicación de nuevos paradigmas de clasificación supervisada en problemas reales, la posibilidad de reducir el tamaño de la base de datos de entrenamiento mediante técnicas evolutivas, y el hecho de que cada problema clasificatorio constituye un nuevo mundo en el que cada paradigma se comporta de manera diferente a como lo hace en problemas anteriores.

12.4 Trabajo futuro

Alguien dijo una vez que la tesis no se acaba, se abandona. Dicho de otro modo, se da por finalizada, aun a sabiendas de que todavía quedan muchas cosas por hacer. Confiando, eso si, en que las cosas hechas sean suficientes. Es por ello que habitualmente quedan muchas cosas por hacer, algunas de las cuales se harán en el futuro, y otras seguramente no.

Como líneas de trabajo abiertas, hay que destacar dos: por un lado, la realización de un estudio de las modificaciones necesarias en los métodos de clasificación para poder manejar datos temporales, y por el otro, utilizar técnicas de clasificación que tengan en cuenta la especificidad y sensibilidad de los modelos creados, y no sólo el porcentaje de bien clasificados. Una tercera cuestión, relacionada con las anteriores, y que se ha tenido presente en los trabajos realizados hasta el momento, es el problema de la discretización de variables, algo que se considera fundamental en los problemas de clasificación supervisada.

1. *Datos temporales*
   Una de las líneas naturales de continuación de esta tesis parece ser la de aplicar las modificaciones necesarias a los métodos desarrollados para que puedan tratar de un modo adecuado datos que tengan una clara influencia temporal. De este modo, se plantea la posibilidad de aplicar este tipo de técnicas a temas como la diabetes, en la que este doctorando ya ha realizado algunos trabajos (Sierra y col., 1995, Sierra y col, 1997) o la predicción de las posibles fluctuaciones de valores en bolsa.

2. *Especificidad, Sensibilidad*
   Los trabajos desarrollados en esta tesis valoran el comportamiento de un clasificador en base al porcentaje de casos bien clasificados que se obtienen con los mismos (o la media

de los porcentajes obtenidos en el caso de realizar una validación cruzada). Otra forma de evaluarlos consiste en ver su comportamiento en base a la Especificidad o la Sensibilidad, o una mezcla de ambas que puede observarse mediante las denominadas curvas ROC. En el futuro se tiene la intención de utilizar este tipo de medidas para evaluar el comportamiento de los paradigmas de clasificación supervisada.

3. *Discretización de variables*
   Poco, o nada, se ha dicho en el desarrollo de esta memoria sobre algo tan fundamental en algunos algoritmos de clasificación supervisada como es la discretización de variables. Como línea abierta queda el realizar un algoritmo de discretización específico para problemas clasificatorios, para lo que se está pensando utilizar algún tipo de algoritmo evolutivo que busque en el espacio de posibles discretizaciones de las variables, cual mejora el comportamiento del clasificador.

Bibliografía

[1] E.H.L., Aarts y J.H.M. Korst (1989): *Simulated Annealing and Boltzmann Machines*. Wiley, Chichester.

[2] E.H.L. Aarts y P.J.M. Van Laarhoven (1985): Statistical cooling: A general approach to combinatorial optimization problems. *Philips Journal of Research*, 40, 193-226.

[3] S. Acid, L.M. de Campos, A. González, R. Molina y N. Pérez de la Blanca (1991): Learning with CASTLE. En Kruse, R. y Siegel, P. (eds.) *Symbolic and Quantitative Approaches to Uncertainty, Lectures Notes in Computer Science 548*, Springer-Verlag, 99-106.

[4] S. Acid y L.M. de Campos (1994): Approximations of causal networks by polytrees: An empirical study. *Proceedings of the Fifth Information Processing and Management of Uncertainty in Knowledge-Based System*, 972-977.

[5] S. Acid, L.M. de Campos (1996): BENEDICT: An Algorithm for Learning Probabilistic Belief Networks. *IPMU'96*, 979–984.

[6] S. Acid y L.M. de Campos (1999): Fast algorithms for learning simplified graphical models. *II Symposium on Artificial Intelligence. CIMAF99. Special Session on Distributions and Evolutionary Optimization* 325–331.

[7] S. Acid (1999): Métodos de aprendizaje de redes de creencia. Aplicación a la clasificación. *Tesis doctoral*, Universidad de Granada.

[8] J.B. Adams (1976): A probability model of medical reasoning and the MYCIN model. *Mathematical Biosciences*, **32**, 177-186.

[9] D. Aha, D. Kibler y M.K. Albert (1991): Instance-Based learning algorithms. *Machine Learning* **6**, 37-66.

[10] D. Aha (1992): Tolerating, irrelevant and novel attributes in instance-based learning algorithms. *International Journal of Man-Machine Studies* **36** (1), 267-287.

[11] E.I. Altman (1968): Financial Ratios, Discriminant Analysis and the Prediction of Business Failure. *Journal of Finance*, 589-609.

[12] S.K. Andersen, K.G. Olesen, F.V. Jensen y F. Jensen (1989): HUGIN - a shell for building Bayesian belief universes for expert systems. *Eleventh International Joint Conference on Artificial Intelligence* 1128-1133.

[13] S. Andreassen, F.V. Jensen y K.G. Olesen (1991): Medical expert systems based on probabilistic networks. *International Journal of Bio-Medical Computing*, **28**, 1-30.

[14] J. Argenti (1976): *Corporate Collapse: the Causes and Symptoms*. McGraw-Hill. London.

[15] S. Baluja (1994): Population-based incremental learning: A method for integrating genetic search based function optimization and competitive learning. *Technical Report No. CMU-CS-94-163*. Pittsburgh, PA: Carnegie Mellon University.

[16] W. Beaver (1966): Financial Ratios as Predictors of Failure. Empirical Research in Accounting Selected Studies. *Supplement of Journal of Accounting Research*, 71-111.

[17] I.A. Beinlinch, H.J. Suermondt, R.M. Chavez y G.F. Cooper (1989): The ALARM monitoring system: A case study with two probabilistic inference techniques for belief networks. *Proceedings of the Second European Conference on Artificial Intelligence in Medicine*, 247-256.

[18] R.R. Bouckaert (1992): Optimizing causal orderings for generating DAGs from data. *Uncertainty in Artificial Intelligence. Proceedings of the Eighth Conference*, 9-16.

[19] R.R. Bouckaert (1993): Probabilistic network construction using the minimum description length principle. en Clarke, M., Kruse, R. y Moral, S. (eds.) *Lectures Notes in Computer Science 747, Symbolic and Quantitative Approaches to Reasoning and Uncertainty, ECSQARU'93*, 41-48.

[20] R.R. Bouckaert (1994): Properties of Bayesian belief networks learning algorithms. *Uncertainty in Artificial Intelligence. Tenth Annual Conference*, 102-109.

[21] R.R. Bouckaert, E. Castillo, y J.M. Gutiérrez (1996): A Modified Simulation Scheme for Inference in Bayesian Networks. *International Journal of Approximate Reasoning*, 14:55-80.

[22] L. Breiman, J.H. Friedmann, R.A. Olshen y C.J. Stone (1984): *Classification and Regression Trees*. Wadsworth, Belmont, CA.

[23] C.E. Brodley (1994): Recursive Authomatic Algorithm selection for Inductive Learning. *Tesis doctoral*. Universidad de Massachusetts.

[24] W. Buntine (1991): Theory refinement in Bayesian networks. En *Proceedings of the Seventh Conference on Uncertainty in Artificial Intelligence*, 52–60.

[25] W. Buntine (1996): A Guide to the Literature on Learning Probabilistic Networks from Data. *IEEE Transactions on Knowledge and Data Engineering*, Vol. 8, No. 2, 195-210.

[26] M. Cameron-Jones (1995): Instance selection by encoding length heuristic with random mutation hill climbing. *IEEE Proceedings of the eighth Australian Joint Conference on Artificial Intelligence*, World Scientific, 99-106.

[27] E. Castillo, J.M. Gutiérrez y A.S. Hadi (1997): *Expert Systems and Probabilistic Network Models*. Springer-Verlag

[28] B. Cestnik (1990): Estimating Probabilities: a crucial task in Machine Learning. *Proceedings of the European Conference on Artificial Intelligence*, 147-149.

[29] B.B. Chaudhuri (1996): A new definition of neighbourhood of a point in multi-dimensional space. *Pattern Recognition Letters*, **17**, 11-17.

[30] D.M. Chickering, D. Geiger, y D. Heckerman (1994): Learning bayesian networks is NP-hard. *Technical Report MSR-TR-94-17*, Microsoft Research, Advanced Technology Division, Microsoft Corporation, One Microsoft Way, Redmond, WA 98052.

[31] D. M. Chickering, D. Geiger, y D. Heckerman (1995): Learning Bayesian Networks: Search Methods and Experimental Results. *Preliminary Papers of the Fifth International Workshop on Artificial Intelligence and Statistics*, 112-128.

[32] C.K. Chow y C.N. Liu (1968): Approximating discrete probability distributions with dependence trees. *IEEE Transactions on Information Theory*, **14**, 462-467.

[33] P. Clark y T. Nibblet (1989): The CN2 induction algorithm. *Machine Learning* **3** (4), 261-283.

[34] W.W. Cohen (1995): Fast Effective Rule Induction. *Machine Learning: Proceedings of the twuelfth Internacional Conference*.

[35] G.F. Cooper (1990): The computational complexity of probabilistic inference using Bayesian belief networks. *Artificial Intelligence*, **42(2-3)**, 393-405.

[36] G.F. Cooper y E.A. Herskovits (1992): A Bayesian method for the induction of probabilistic networks from data. *Machine Learning*, **9**, 309-347.

[37] S. Cost y S. Salzberg (1993): A weighted nearest neighbor algorithm for learning with symbolic features. *Machine Learning* **10**(1), 57-78.

[38] T.M. Cover y P.E. Hart (1967): Nearest Neighbor Pattern Classification. *IEEE Trans. IT-13* **1**, 21-27.

[39] P. Dagum y M. Luby (1993): Approximating probabilistic inference in Bayesian belief networks is NP-hard. *Artificial Intelligence*, **60(1)**, 141-153.

[40] C. Darwin (1859): *On the Origin of Species by Means of Natural Selection*, Murray, London.

[41] B.V. Dasarathy (1991): *Nearest Neighbor (NN) Norms: NN Pattern Recognition Classification Techniques*. IEEE Computer Society Press

[42] L. Davis (ed.) (1991): *Handbook of Genetic Algorithms*. Van Nostrand Reinhold, New York.

[43] A.P. Dawid (1979): Conditional independence in statistical theory. *Journal of the Royal Statistics Society, Series B*, **41**, 1-31.

[44] L.M. de Campos (1994): Independence relationships in possibility theory and their application to learning belief networks. *Proceedings of the ISSEK Workshop Mathematical and Statistical Methods in Artificial Intelligence.*

[45] K.A. De Jong (1975): An analysis of the behaviour of a class of genetic adaptive systems. *Tesis doctoral*, University of Michigan.

[46] T.G. Dietterich (1996): Proper statistical tests for comparing supervised classification learning algorithms. *Technical Report*. Departament of Computer Science, Oregon State University, Corvallis, OR.

[47] T.G. Dietterich (1997): Machine Learning Research: four current directions. *AI Magazine* **18** (4), 97-136.

[48] F.J. Díez (1992): Local conditioning in Bayesian networks. *Technical Report R-181*, Cognitive Systems Laboratory, University of California, Los Angeles.

[49] F.J. D'ıez (1994): Sistema experto bayesiano para ecocardiografía. *Tesis doctoral*, Universidad Nacional de Educación a Distancia (UNED).

[50] F.J. Díez y E. Nell (1998): Introducción al Razonamiento Aproximado. Departamento de Inteligencia Artificial, UNED.

[51] S. Dizdaverich, F. Lizarraga, P. Larrañaga, B. Sierra y M. Y. Gallego (1998): Statistical and machine learning methods in the prediction of bankruptcy. *Third International Meeting on Artificial Intelligence in Accounting, Finance and Tax*. E. Bonson, G. J. Sierra (eds.), 85-100.

[52] S. Dizdaverich, P. Larrañaga, J. M. Peña, B. Sierra, M. Y. Gallego y J. A. Lozano (1999): Predicción del fracaso empresarial mediante la combinación de clasificadores provenientes de la estadística y el aprendizaje automático. *Tecnologías Inteligentes para la Gestión Empresarial*. Bonson (ed.). ra-ma, 71-114.

[53] P. Djouadi, E. Boucktache (1997): A fast algorithm for the Nearest-Neighbor Classifier. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 19, No. 3, 277-281.

[54] P. Domingos, M. Pazzani (1997): Beyond Independence: Conditions for the Optimality of the Simple Bayesian Classifier. *Machine Learning*, **29**, 103-130.

[55] R. Duda, y P. Hart, (1973): *Pattern classification and scene analysis*. John Wiley & Sons, Inc., New York, NY.

[56] S.A. Dudani (1975): The distance-weighted K nearest Neighbour rule. *IEEE Transactions on Systems, Man and Cybernetics*, Vol. 6, No. 4, 325-327.

[57] R. Etxeberria P. Larrañaga, y J. M. Picaza (1997): Analysis of the behaviour of the genetic algorithms when searching Bayesian networks from data. *Pattern Recognition Letters* **18: 11–13**, 1269-1273.

[58] R. Etxeberria y P. Larrañaga (1999): Global optimization with Bayesian networks. *II Symposium on Artificial Intelligence. CIMAF99. Special Session on Distributions and Evolutionary Optimization.*

[59] K.J. Ezawa y S.W. Norton (1996): Constructing Bayesian networks to predict uncollectible telecomunications accounts. *IEEE Expert/Intelligent Systems & Their Applications* (11), 45-51.

[60] R. Fisher (1936): The Use of Multiple Measurements in Taxonomic Problems. *Annals of Eugenics*, 7, 179-188.

[61] E. Fix y J. L. Hodges Jr (1951): Discriminatory analysis, nonparametric discrimination. *USAF school of Aviation Medicine, Randolf field, Project 21-49-004, Rept 4.*

[62] Y. Freund y R.E. Shapire (1996). Experiments with a new boosting algorithm. *Proceedings of the therteenth international conference on Machine Learning*. Morgan Kaufmann, San Francisco, CA, 148-156.

[63] J.H. Friedman, F. Baskett y L.J. Shustek (1975): An algorithm for finding Nearest Neighbours. *IEEE Transactions on Computers*, **10**, 1000-1006.

[64] N. Friedman y M. Goldszmidt (1996): Building Classifiers using Bayesian Networks. *AAAI Conference*.

[65] N. Friedman, D. Geiger y M. Goldszmidt (1997): Bayesian Network Classifiers. *Machine Learning* **19** (4), 131-163.

[66] K. Fukunaga y M. Narendra (1975): A Branch and Bound algorithm for computing K Nearest Neighbours. *IEEE Transactions on Computers*, **10** 750-753.

[67] K. Fukunaga (1990): *Introduction to statistical pattern recognition*. Academic Press, San Diego, CA.

[68] J. A. Gámez y J. M. Puerta, (1998): *Sistemas expertos probabilísticos*. Ediciones de la Universidad de Castilla–La Mancha.

[69] G.W. Gates (1972): The reduced Nearest Neighbour rule. *IEEE Transactions on Information Theory*, **3** 431-433.

[70] D.E. Goldberg (1989): *Genetic Algorithms in Search, Optimization and Machine Learning*. Addison-Wesley, Reading, MA.

[71] I.J. Good (1965): *The estimation of probabilities*. MIT-press, Cambridge.

[72] J. Hand (1997): *Construction and Assessment of Classification Rules*. John Wiley.

[73] D. Heckerman, D. Geiger y D.M. Chickering (1995): Learning Bayesian networks: The combination of knowledge and statistical data. *Machine Learning* (20): 197-243.

[74] D. Heckerman (1996): A tutorial on learning with Bayesian networks. *Technical report MRS-TR-95-06*, Microsoft research advanced technology division.

[75] M. Henrion (1988): Propagating uncertainty in Bayesian networks by probabilistic logic sampling. *Proceedings of the Fourth Conference on Uncertainty in Artificial Intelligence*, 149-163.

[76] T.K. Ho, J.J. Hull y S.N. Srihati (1994): Decision combination in Multiple Classifier Systems. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, **vol. 16**, 66-75.

[77] J. Holland (1975): *Adaptation in Natural and Artificial Systems*. University of Michigan Press, Ann Arbor.

[78] R.C. Holte (1994): Very simple classification rules perform well on most commonly used databases. *Machine Learning*, **11**, 63-90.

[79] D.W. Hosmer, S. Lemeshow (1989): *Applied Logistic Regression*. Wiley Series in Probability and Mathematical Statistics.

[80] E.B. Hunt y C.I. Hovland (1963): Programming a model of human concept formulation. En *Computers and Thought*. Feigenbaum y Feldman (eds), McGraw-Hill Book Company, Inc, 310-325.

[81] I. Inza, P. Larrañaga y B. Sierra y M. Niño (1998): Conbination of Classifiers. A case Study in Oncology. *Informe interno* EHU-KZAA-IK 1-98.

[82] I. Inza, P. Larrañaga, B. Sierra, R. Etxeberria, J.A. Lozano, J.M. Peña (1999): Representing the joint behaviour of machine learning inducers by Bayesian networks. *Pattern Recognition Letters* (20) (11-13), 1201-1209.

[83] I. Inza (1999b): Feature Weighting for Nearest Neighbor Algorithm by Bayesian Networks based Combinatorial Optimization. *Proceedings of the Student Session of Advanced Course on Artificial Intelligence. ACAI 99*, 33-35. Chania, Greece.

[84] M.I. Izarzugaza (1994): Informe del registro de Cáncer de Euskadi 1990. *Osasunkaria* 8-11.

[85] F.V. Jensen, S.L. Lauritzen y K.G. Olesen (1990a): Bayesian updating in causal probabilistic networks by local computations. *Computational Statistics Quarterly*, **4**, 269-282.

[86] F.V. Jensen, K.G. Olesen y S.A. Andersen (1990b): An algebra of Bayesian belief universes for knowledge-based systems. *Networks* **20**, 637-659.

[87] F.V. Jensen (1996): *Introduction to Bayesian networks*. University College of London.

[88] R. Kass, L. Tierney y J. Kadane (1988): Asymptotics in Bayesian computation, En J. Bernardo, M. De Groot, D. Lindley and A. Smith, eds., *Bayesian Statistics 3*, 261–278, Oxford University Press.

[89] J.H. Kim y J. Pearl (1983): A computational model for causal and diagnostic reasoning in inference systems. *Proceedings of the Eighth International Joint Conference on Artificial Intelligence*, 190-193.

[90] S. Kirkpatrick, C.D. Gelatt y M.P. Vecchi (1983): Optimization by simulated annealing. *Science*, 220, 671-680.

[91] W.A. Knaus, E.A. Draper, D.P. Wagner y J.E. Zimmerman (1985): APACHE II: A severity of disease classification system. *Critical Care Medicine 13:818-829.*

[92] R. Kohavi, D. Sommerfield (1995): Feature Subset Selection using the wrapper model: overfitting and dynamic search space topology. *Proceedings of the First International Conference on Knowledge Discovery and Data Mining, KDD'95*, Montreal, Canada, 1995, 192-197.

[93] R. Kohavi (1996): Scaling up the accuracy of naive-Bayes classifiers: a decision-tree hybrid. *Proceedings of the Second International Conference on Knowledge Discovery and Data Mining.*

[94] R. Kohavi, D. Sommerfield y J. Dougherty (1997): Data mining using MLC++, a Machine Learning Library in C++. *International Journal of Artificial Intelligence Tools* **Vol. 6, num. 4**, 537-566. *[http://www.sgi.com/Technology/mlc/].*

[95] I. Kononenko, J. Bratko y E. Roskar (1984): Experiments in automatic learning of medical diagnostic rules. *Technical Report*. Jozef Stefan Institute.

[96] P.J. Krause (1998): *Learning Probabilistic Networks*. Technical Report. Philips Research Laboratories.

[97] S. Kullback y R.A. Leibler (1951): On information and sufficiency. *Ann. Math. Statist.*, **39**, 1236-1243.

[98] W. Lam y F. Bacchus (1993): Using causal information and local measures to learn Bayesian networks. *Uncertainty in Artificial Intelligence, Proceedings of the Ninth Conference,* 243-250.

[99] W. Lam y F. Bacchus (1994): Learning Bayesian belief networks. An approach based on the MDL Principle. *Computational Intelligence, 10(4).*

[100] P. Larrañaga, R.H. Murga, M. Poza y C.M.H. Kuijpers (1996a): Structure learning of Bayesian networks by hybrid genetic algorithms. En *Learning from Data. Artificial Intelligence and Statistics V. Lecture Notes in Statistics, 112.* Fisher, D. Lenz, H.-J. (eds.). Springer-Verlag, 165-174.

[101] P. Larrañaga, M. Poza, Y. Yurramendi, R.H. Murga y C.M.H. Kuijpers (1996b): Structure learning of Bayesian networks by genetic algorithms: A performance analysis of control parameters. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 18, No. 9, 912-926.

[102] P. Larrañaga, C.M.H. Kuijpers, R.H. Murga y Y. Yurramendi (1996c): Searching for the best ordering in the structure learning of Bayesian networks. *IEEE Transactions on Systems, Man and Cybernetics*, Vol. 26, No. 4, 487-493.

[103] P. Larrañaga, B. Sierra, M.Y. Gallego y M.J. Michelena (1996d): Bayesian networks induced by genetic algorithms in the prediction of the survival of breast cancer. *ITHURS96. International Conference on Intelligent Technologies in Human-Related Sciences*. León.

[104] P. Larrañaga, B. Sierra, M.Y. Gallego y M.J. Michelena (1996e): Predicción de la supervivencia del cáncer de mama por medio de redes Bayesianas inducidas por algoritmos genéticos. *XIV Congreso Anual de la Sociedad Española de Ingenieria Biomédica*. Pamplona.

[105] P. Larrañaga, M.Y. Gallego, B. Sierra, L. Urkola y M.J. Michelena (1997a): Bayesian networks, rule induction and logistic regression in the prediction of the survival of women suffering from breast cancer. *Lecture Notes in Artificial Intelligence 1323*. E. Costa, A. Cardoso (eds.), 303-308. Springer-Verlag.

[106] P. Larrañaga, B. Sierra, M.Y. Gallego, M.J. Michelena y J.M. Pikaza (1997b): Learning Bayesian networks by genetic algorithms. A case study in the prediction of survival in malignant skin melanoma. *Lecture Notes in Artificial Intelligence 1211*. E. Keravnou, C. Garbay, R. Baud, J. Wyatt (eds.), 261-272. Springer-Verlag.

[107] P. Larrañaga, R. Etxeberria, J.A. Lozano, B. Sierra, I. Inza y J.M. Pena (1999): A review of the cooperation between evolutionary computation and probabilistic graphical models. *Proceedings of the Second Symposium on Artificial Intelligence*. Adaptive Systems. CIMAF 99. Special Session on Distributions and Evolutionary Computation, 314-324. La Habana.

[108] S.L. Lauritzen (1996): *Graphical Models*. Oxford University Press.

[109] S.L. Lauritzen y D.J. Spiegelhalter (1988): Local computations with probabilities on graphical structures and their application on expert systems. *Journal Royal of Statistical Society B* **50**, 157-224.

[110] J.R. Le Gall, S. Lemeshow y F. Saulnier (1993): A new Simplified Acute Physiology Score (SAPS II) based on a European/North American multicenter study. JAMA 270:2957-2963.

[111] S. Lemeshow, D. Teres, J. Klar, J.S. Avrunin, S.H. Gehlbach y J. Rapoport (1993): Mortality Probability Models (MPM II) based on an international cohort of intensive care unit patients. JAMA 270:2478-2486.

[112] U. Lipowezky (1998): Selection of the optimal prototype subset for 1-NN classification. *Pattern Recognition Letters*, **19**, 907-918.

[113] F. Lizarraga (1996): Modelos Multivariantes de predicción del fracaso empresarial: una aplicación a la realidad de la información contable española. *Tesis doctoral*, Universidad Pública de Navarra.

[114] R. López de Mantaras (1991): A distance-based attribute selection measure for decision tree induction. *Machine Learning*, **6**, 81-92.

[115] Y. Lu (1996): Knowledge integration in a multiple classifier system. *Applied Intelligence* **6**, 75-86.

[116] J.B. MacQueen (1967): Some methods for classification and analysis of multivariate observations. *Proceedings of fifth Berkeley simposium*. Vol 2, 281-297.

[117] N. Metropolis, A. Rosenbluth, M. Rosenbluth, A. Teller y E. Teller (1953): Equation of State Calculations by Fast Computing Machines. *J. of Chem Physics* **21**, 1087-1092.

[118] W.S. McCulloch y W. Pitts (1943): A logical calculus of ideas inmanent in nervous activity. *Bulletin of Mathematic BioPhysics* **5**, 115-133.

[119] Z. Michalewicz (ed.) (1994): *Proceedings of the First IEEE Conference on Evolutionary Computation*. IEEE Service Center.

[120] M.L. Micó, J. Oncina y R.C. Carrasco, (1994): A new version of the Nearest Neighbour aproximating and eliminating search algorithm (EASA) with linear pre-procesing time and memory requirements. *Pattern Recognition Letters*, **15**, 9-18.

[121] M.L. Micó (1996): Algoritmos de búsqueda de vecinos más próximos en espacios métricos. *Tesis doctoral*, Universidad Politécnica de Valencia.

[122] M.L. Micó y J. Oncina (1998): Comparision of fast nearest neighbour classifiers for handwritten character recognition. *Pattern Recognition Letters*, **19** 351-356.

[123] T. Mitchell (1997): *Machine Learning*. McGraw-Hill.

[124] J.N. Morgan y J.A. Sonquist (1963): Problems in the analysis of survey data, and a proposal. *J. Americ. Statist. Assoc.* **58**, 415-434.

[125] H. Mühlenbein y G. Paß(1996): From Recombination of Genes to the Estimation of Distributions I. Binary Parameters. En Voigt, H. M., et al. (eds). *Lecture Notes in Computer Science 1411: Parallel Problem Solving from Nature - PPSN IV*, 178-187.

[126] P.M. Murphy y D.W. Aha (1994): UCI Repository of Machine Learning databases. *[http://www.ics.uci.edu/ mlearn/MLRepository.html]*. Irvine, CA: University of California, Department of Information and Computer Science

[127] S.K. Murthy y S. Salzberg (1994): A system for the induction of oblique decision trees. *Journal of Artificial Intelligence Research* 2, 1-33.

[128] J.W. Myers, K.B. Laskey y T. Levitt (1999): Learning Bayesian Networks from Incomplete data with Stochastic Search Algorithms. *Proceedings of the Fifteenth Conference on Uncertainty in Artificial Intelligence*, 476–485.

[129] R.E. Neapolitan (1990): *Probabilistic Reasoning in Expert Systems. Theory and Algorithms*. John Wiley & Sons, Inc.

[130] J. Pearl (1986): Fusion, propagation, and structuring in belief networks. *Artificial Intelligence*, **29(3)**, 241-288.

[131] J.A. Ohlson (1980): Financial Ratios and the Probabilistic Prediction of Bankruptcy. *Journal of Accounting Research*, 18, 1, 109-111.

[132] J. Pearl (1987): Evidential Reasoning using Stochastic simulation of causal models. *Artificial Intelligence* **32(2)** 245-257.

[133] J. Pearl (1988): *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. Morgan Kaufmann, San Mateo.

[134] M. Pelikan, D.E. Goldberg y F. Lobo (1999): A Survey of Optimization by Building and Using Probabilistic Models. *IlliGAL Report*, No. 99018.

[135] G.M. Provan y M. Singh (1995): Learning Bayesian networks using feature selection. *Fifth International Workshop on Artificial Intelligence and Statistics*, Florida, 450-456.

[136] J.R. Quinlan (1986): Induction of Decision Trees. *Machine Learning* **1**, 81-106.

[137] J.R. Quinlan (1993): *C4.5: Programs for Machine Learning*. Morgan Kaufmann Publishers, Inc. Los Altos, California.

[138] M. Ramoni y P. Sebastiani (1999): An Introduction to the Robust Bayesian Classifier. *KMi Technical Report*, KMi-TR-79, Knowledge Media Institute, The Open University.

[139] C. Reeves (1993): *Modern Heuristic Techniques for Combinatorial Problems*. Blackwell Scientific Publications.

[140] G.L. Ritter, H.N. Woodruff, S.R. Lowri y T.L. Isenhour (1975): An algorithm fo a selective Nearest-Neighbour Decision Rule. *IEEE Transactions on Information Theory*, **21**, 665-669.

[141] J. Rissanen (1987): Stochastic complexity. *Journal of the Royal Statistical Society, Series B,* **49(3)**, 223-239.

[142] R.W. Robinson (1977): Counting unlabeled acyclic digraphs. Little, C.H.C. (ed.) *Lectures notes in mathematics 622: Combinatorial mathematics V*, Springer-Verlag, New York, 28-43.

[143] C. Rojas-Guzmán y M. A. Kramer (1996): An Evolutionary Computing Approach to Probabilistic Reasoning on Bayesian Networks. *Evolutionary Computation, 4(1)*, 57-85.

[144] F. Rosenblatt (1962): *Principles of Neurodynamics: Perceptrons ant the Theory of Brain Mechanism*. Washington DC: Spartan Books.

[145] D.E. Rumelhart, J.L. McCelland, PDP Research group (1986): *Paralell distributed processing: Explorations in the microstructure of cognition*. **1**, Cambridge, MA: MIT Press.

[146] J.S. Sánchez, F. Pla y F.J. Ferri, (1997): Prototype selection for the Nearest Neighbour rule through proximity graphs. *Pattern Recognition Letters*, **18**, 507-513.

[147] J.S. Sánchez (1998): Aprendizaje y clasificación basados en criterios de vecindad. Métodos alternativos y análisis comparativos. *Tesis doctoral*, Universitat Jaume I.

[148] N. Serrano, P. Revuelta, J.J. Jiménez, E. Plasencia, J. Martínez, M.T. Brouard y M.L. Mora (1998): Levels of severity at ICU discharge related those at ICU admision: Criterion for ICU discharge? *Intensive Care Med*, 24:s10.

[149] C. Schaffer (1994): Cross-Validation, Stacking and Bi-level Stacking: Metamethods for classification Learning. *P. Cheesman y P. Oldford: Selecting models from data Artificial intelligence and statistics IV*, Springer-Verlag, NY, 51-59.

[150] G. Schwarz (1978): Estimating the dimension of a model. *Annals of Statistics*, 7(2): 461-464.

[151] E.H. Shortliffe, R. Dans, S.G. Axline, B.G. Buchanan, C.C. Green y S.N. Cohen (1975): Computer-based consultations in clinical therapeutics: explanation a rule acquisition capabilities of the MYCIN system. *Computers and Biomedical Research*, **8**, 303-320.

[152] B. Sierra, L. Aldamiz-Echevarria, J.M. Picaza, C. Amuchastegui, J.R. Zubizarreta, P. Martul, E. Alustiza, J. Albisu y E. Blarduni (1995): Visualization and Interpretation of Diabetic Patient Data from a Portable Storage Device. *Proceedings of the American Academy of Pediatrics 1995 Annual Meeting*. Section on Computers and Others Technologies.

[153] B. Sierra, A. Remiro y J.M. Pikaza, (1997a): DIABETES I: Ume diabetikoak zaintzeko sistema aditua. *Elhuyar. Zientzia eta Teknika*, 122,122, 36-41.

[154] B. Sierra y P. Larrañaga (1997b): Searching for the optimal Bayesian Network in classification tasks by genetic algorithms. *WUPES97*, 144-155.

[155] B. Sierra y P. Larrañaga (1998): Predicting survival in malignant skin melanoma using Bayesian Networks automatically induced by genetic algorithms. An empirical comparision between different approaches. *Artificial Intelligence in Medicine* **14**, 215-230.

[156] B. Sierra, N. Serrano, P. Larrañaga, E.J. Plasencia, I. Inza, J.J. Jiménez, J.M. De la Rosa y M.L. Mora (1999a): Machine Learning Inspired Approaches to Combine Standard Medical Measures at an Intensive Care Unit. *Lecture Notes in Artificial Intelligence*, 366-371.

[157] B. Sierra, N. Serrano, P. Larrañaga, E.J. Plasencia, I. Inza, J.J. Jiménez, J.M. De la Rosa y M.L. Mora (1999b): Bayesian networks as consensed voting system in the construction of a multi-classifier. A case study using Intensive Care Unit patient data. *AIMDM99*, CAIC Workshop, 57-66.

[158] B. Sierra, P. Larrañaga e I. Inza (2000a): K Diplomatic Nearest Neighbour: giving equal chance to all existing classes. *Journal of Artificial Intelligence Research*. Enviado.

[159] B. Sierra, N. Serrano, P. Larrañaga, E.J. Plasencia, I. Inza, J.J. Jiménez, J.M. De la Rosa y M.L. Mora (2000b): Using Bayesian networks in the construction of a multi-classifier. A case study using Intensive Care Unit patient data. *Artificial Intelligence in Medicine*. Enviado.

[160] G. Syswerda, (1993): Simulated crossover in genetic algorithms. En L. D. Whitley ed., *Foundations of Genetic Algorithms 2*, 239–255, San Mateo, CA: Morgan Kaufmann.

[161] D.B. Skalak (1994): Prototipe and feature selection by Sampling and Random Mutation Hill Climbing Algortithms. *Proceedings of the Eleventh International Conference on Machine Learning*, NJ. Morgan Kaufmann. 293-301.

[162] D.B. Skalak (1997): Prototipe selection for composite Nearest Neighbor classifiers. *Ph.D. Thesis*, University of Massachusetts Amherst.

[163] D.J. Spiegelhalter, A.P. Dawid, S.L. Lauritzen, y R.G. Cowell (1993): Bayesian analysis in expert systems. *Statistical Science* 8:219-247.

[164] M. Stone (1974): Cross-validation choice and assessment of statistical procedures. *Journal Royal of Statistical Society* **36**, 111-147.

[165] J. Suzuki (1993): A construction of Bayesian networks from databases based on an MDL Principle. *Uncertainty in Artificial Intelligence, Proceedings of the Ninth Conference*, 266-273.

[166] K.M. Ting (1997): Theory combination: an alternative to data combination. *Technical Report*, University of Waikato, New Zealand.

[167] I. Tomek (1976): An experiment with the edited nearest neighbour rule. *IEEE Transactions on Systems, Man and Cybernetics*, Vol. 6, 448-452.

[168] E. Vidal (1986): An algorithm for findinf nearest neighbours in (approximately) constant average time. *Pattern Recognition Letters*, **4** 145-157.

[169] J. Voisin y P.A. Devijver (1987): An application of the Multiedit-Condensing technique to the reference selection problem in a print recognition system. *Pattern Recognition*, **5** 465-474.

[170] D. Wettschereck (1994): A study of distance-based Machine Learning Algorithms. *Ph.D. Thesis*, Oregon State University.

[171] D. Wettschereck (1995): An Experimental Comparison of the Nearest-Neighbor and Nearest-Hyperrectangle Algorithms. *Machine Learning* **19**, 1-25.

[172] D. Wettschereck, D.W. Aha y T. Mohri, (1997): A Review and Empirical Evaluation of Feature Weighting Methods for a Class of Lazy Learning Algorithms. *Artificial Intelligence Review* **11**, 273-314.

[173] J. Whittaker (1990): *Graphical models in applied multivariate statistics.* John Wiley and Sons.

[174] B. Widrow (1962): Generalization and information storage in networks of adaline "neurons". En Yovits, M.C., Jacobi, G.T. y Goldstein, G.D., editors. *Self-Organizing Systems*, 435-461. Chicago, Illinois, Spartan.

[175] D.L. Wilson (1972): Asymptotic properties of nearest neighbour rules using edited data. *IEEE Transactions on Systems, Man and Cybernetics*, Vol. 2, 408-421.

[176] D. Wolpert (1992): Stacked Generalization. *Neural Networks* **5**, 241-259.

[177] M. L. Wong, W. Lam, W., y K. S. Leung (1999): Using Evolutionary Computation and Minimum Description Length Principle for Data Mining of Probabilistic Knowledge. *IEEE Pattern Analysis and Machine Intelligence*. **21(2)**, 174-178.

[178] L. Xu, A. Kryzak y C.Y. Suen (1992): Methods for combining multiple classifiers and their applications to handwriting recognition. *IEEE Transactions on SMC*, **22** 418-435.

[179] T.P. Yunk (1976): A technique to identify Nearest Neighbours. *IEEE Transactions on SMC*, **10**.

[180] J. Zhang (1992): Selecting Typical instances in Instance-Based Learning. *Proceedings of the Ninth International Machine Learning Workshop*, Aberdeen, Escocia. Morgan-Kaufmann, San Mateo, Ca, 470-479.