UNIVERSIDAD DE OVIEDO

Estimación de la información mutua en problemas con datos imprecisos

Tesis Doctoral

Mª del Rosario Suárez Fernández

Oviedo, Abril de 2007

Departamento de Informática

UNIVERSIDAD DE OVIEDO

Estimación de la información mutua en problemas con datos imprecisos

MEMORIA QUE PRESENTA

Mª del Rosario Suárez Fernández

PARA OPTAR AL GRADO DE DOCTOR EN INFORMÁTICA

Abril de 2007

DIRECTOR

Luciano Sánchez Ramos

Departamento de Informática

La memoria titulada "Estimación de la información mutua en problemas con datos imprecisos", que presenta Dña. Mª del Rosario Suárez Fernández para optar al grado de doctor, ha sido realizada dentro del programa de doctorado "Informática" (bienio 1999-2001) del Departamento Informática de la Universidad de Oviedo bajo la dirección del doctor D. Luciano Sánchez Ramos.

Oviedo, Enero de 2006

El Doctorando

Fdo: Mª del Rosario Suárez Fernández

El Director

Fdo: Luciano Sánchez Ramos

Tesis Doctoral cofinanciada con ayuda del
Ministerio de Ciencia y Tecnología bajo el proyecto
TIN2005-08386-C05-05.

Agradecimientos

Dedico esta memoria a mi familia y amigos por la paciencia y compresión ante los malos momentos por los que he podido pasar. A mi marido porque siempre y en todo momento me ha apoyado incondicionalmente, soportando mis momentos de mal humor pacientemente. A mis hijas porque son unas de las personitas a las que mas quiero en este mundo y a veces han tenido que soportar lo que no debían haber soportado.

Agradezco a Marta y Eugenia el apoyo y la ayuda que me han ofrecido, siempre y en todo momento, sobre todo con esos cafés interminables donde las risas y carcajadas era lo único que se oía desde el otro lado del pasillo junto a esa máquina de café.

Quiero dar las gracias también a todos los compañeros del grupo de Investigación de Metrología y Modelos de la Universidad de Oviedo, Pepe, Adolfo, Kike, Jose Ramón, Luis y Luciano, con los que siempre he podido contar para todo y que me han ayudado en todo lo que he necesitado.

Por último y no por ello lo menos importante, quiero dar las gracias a Luciano, mi director de tesis, por la labor que ha desempeñado, sin el cual esta tesis no estaría aún terminada. Le doy las gracias por el apoyo y comprensión, por su tiempo y su paciencia. Por todo ello, gracias.

Índice

Índice de Figuras

Índice de Tablas

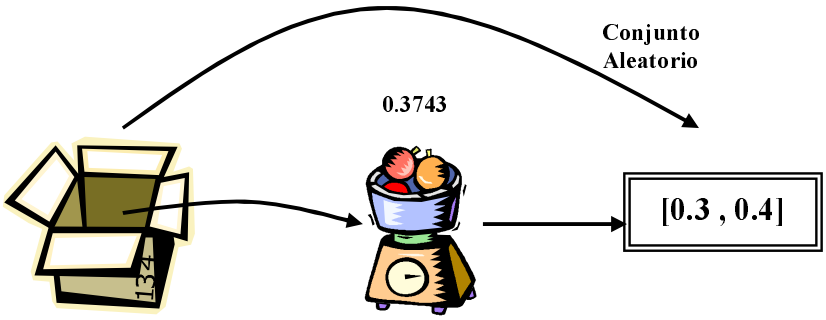
Introducción

Motivación del trabajo

Los problemas de clasificación implican procesos de clasificación que están presentes en la mayoría de las actividades humanas. En su definición más amplia, el proceso de clasificación esta presente en cualquier contexto en el que se toma una decisión en base a información disponible. Es importante medir el grado de dependencia de dicha información, que puede realizarse con técnicas de correlación si la dependencia es lineal o midiendo la Información Mutua si esta dependencia no es lineal.

En algunos problemas, la información que se nos proporciona sobre las magnitudes involucradas, no es precisa, incluye incertidumbre, es decir, no tenemos un valor definido, sino que sabemos que este pertenece a un intervalo. Normalmente el nivel de incertidumbre varía dependiendo del contexto y problema a tratar. Esta imprecisión o incertidumbre puede venir dada por la inclusión de ruido en los datos, se trata de variables aleatorias que se tratan mediante técnicas estadísticas o bien porque los datos que nos proporcionan son un intervalo. En este último caso, hablamos de un conjunto aleatorio y como consecuencia de ello de una familia de variables aleatorias. No sabemos cual es la distribución de probabilidad, ya que tenemos un intervalo, pero si la podemos aproximar.

Supongamos que tenemos un escenario como el de la figura 0.1. Una caja con frutas, de las cuales no sabemos nada y una báscula para pesar dichas frutas. La caja contiene un conjunto de piezas de frutas con distinto peso y de distinta naturaleza. El hecho de seleccionar una pieza de fruta de la caja a ciegas es lo que se llama *experimento aleatorio*, que no es mas que aquél que es susceptible de dar varios resultados, no pudiéndose predecir de antemano cuál de ellos va a producirse en una experiencia concreta. Por otra parte, dicha pieza de fruta, cuando se pesa, da lugar a una *variable aleatoria*, que no es más que una función que asocia un número real, perfectamente definido, a cada punto muestral.



Experimento Aleatorio	Variable Aleatoria	Error de Observación
Elegir una pieza de fruta	Pesar la pieza de fruta	Cuantización del peso

Fig 0.1. Ejemplo de conjunto aleatorio

El principal problema que se nos plantea es la acción que estamos realizando: "pesar". Si la báscula con la que estamos pesando la fruta es imprecisa o simplemente es una báscula que no nos dice los decimales exactos implícitos en el peso, ocurre que el peso no es un valor preciso. Dicho peso se mantendrá dentro de un intervalo como se muestra en a figura 0.1 y dicho intervalo muestra el *error en el peso* de la fruta. En definitiva, el resultado del proceso que engloba desde que se produce el experimento aleatorio hasta que se tiene el peso cuantizado, genera un *conjunto aleatorio*.

Este tipo de conjuntos, poco tratados en la literatura, van a ser uno de los objetivos propuestos en esta tesis. Sin duda, cualquier método propuesto lo verificaremos con conjuntos de datos clásicos y para los cuales se puede intuir el comportamiento, pero además demostraremos que dichos métodos son aplicables y adecuados también a estos.

Resumiendo todo los dicho hasta el momento, vamos a tener dos tipos de problemas que se nos puedan presentar: problemas donde existe ausencia de imprecisión y problemas en los que tenemos una imprecisión inherente en los datos. En este último caso, los datos no son datos precisos, sino intervalos que nos dicen el conjunto de valores entre los cuales puede estar el valor real. A su vez esta imprecisión puede ser: conocida, sabemos el rango de valores en el que puede caer el dato y totalmente desconocida, no tenemos información sobre el dato.

Independientemente de que el valor los datos sea preciso o impreciso, dado un conjunto de ejemplos con dos o más atributos o variables, en muchos problemas que se plantean es necesario saber que grado de dependencia existe entre dos atributos. Este grado de dependencia podrá medirse entre dos atributos discretos, dos atributos continuos o uno discreto y otro continuo. Pero, sin duda, este será el caso mas sencillo, ya que en la práctica, en la mayor parte de los problemas, surge la necesidad de determinar cuanta dependencia existe entre un conjunto de atributos y otro mas, que en ocasiones resulta ser la clase del ejemplo.

La entropía es una medida de información que se utiliza para definir el concepto de Información Mutua. La media de Información Mutua mide la dependencia entre variables o conjuntos de variables. Si lo que estamos midiendo es la dependencia entre una variable y la clase, cuanto mayor sea el valor de la Información Mutua más dependencia existe entre las variables.

Nosotros vamos a utilizar esta medida definiéndola de tal manera que obtengamos una nueva medida de Información Mutua que podrá ser aplicada a problemas reales como es el caso de selección de características lingüísticas o bien la optimización de particiones, problema este último, utilizado para realizar nuestra validación de resultados.

Para el caso de conjunto de datos imprecisos, algunos autores optan por sustituir el intervalo dentro del cual se encontraría el valor real del atributo en un ejemplo concreto, por un valor característico representativo. Nosotros demostraremos que esta forma de solucionar la incertidumbre no es la más correcta porque los resultados no son los esperados. Y, demostraremos que nuestro método, se convierte entonces en el más adecuado, ya que se arrastra dicha incertidumbre a lo largo del proceso, hasta llegar a un resultado final.

Frecuentemente, el ser humano se plantea cuestiones, ante las cuales no se puede dar una solución concreta o precisa, sino que la respuesta, suele ir acompañada de matices que de alguna manera, sentimos la necesidad de representar. Cuando hablamos de la altura de una persona, en muchos casos, no decimos que “Juan mide 1.70”, sino que “Juan es alto”. El problema que se nos plantea es que la altura es un concepto subjetivo y dependiendo que quien lo exprese, los intervalos que se manejan pueden variar.

Cada uno de los conceptos que se nos pueda ocurrir, bien sea la altura, la edad, el peso,…, representa un conjunto de valores reales. Ese conjunto de valores reales, basándonos en algún método existente, se puede particionar de tal forma que a la hora de hablar de una persona que mide

1.75 cm podamos decir que es “alta” por ejemplo. Cada una de estas particiones se corresponde con una etiqueta lingüística que permite dar una mayor interpretabilidad a la información aportada. El principal problema que se nos plantea es que en ocasiones ese particionamiento, no es el más adecuado para los datos que se tratan en un determinado problema.

Dada una variable o atributo de un conjunto de ejemplos, dicho atributo, como se dijo anteriormente, requiere de una representación de su conjunto de valores continuo, mediante particiones. Dichas particiones, lo más normal es que se solapen e incurran en la definición de conjuntos borrosos. Así, supongamos que tenemos un atributo que es el peso de las frutas y tengo el conjunto de ejemplos tanto con precisión como con imprecisión como se muestra en la tabla 0.1.

Ejemplo	Con Imprecisión		Sin Imprecisión	
	Peso	Clase	Peso	Clase
1	110±1	Pera	110	Pera
2	96±1	Manzana	96	Manzana
3	116±1	Pera	116	Pera
4	91±1	Plátano	91	Plátano
5	101±1	Manzana	101	Manzana

Tabla 0.1. Conjunto de ejemplos precisos e imprecisos

Tomando como base el conjunto de valores para el atributo en estudio se puede proponer de forma inicial un particionamiento de los datos de manera uniforme como se muestra en la figura 0.2. El particionamiento no seguirá ningún tipo de criterio, cogiendo el mínimo y el máximo valor posible se realizarán las particiones. Dichas particiones pueden ser las idóneas o no. Nosotros aplicaremos nuestra nueva medida de Información Mutua para optimizar esas particiones iniciales y demostrar que las obtenidas con nuestro método en el peor de los casos igualan y en un número elevado de ocasiones mejoran el resultado de las anteriores.

Finalmente, en nuestro empeño de poder demostrar que realmente esta definición, efectivamente proporciona resultados viables, se indaga en los procesos de clasificación que desde el punto de vista de esta tesis, serán la herramienta mediante la cual podamos llevar a cabo los experimentos necesarios para este trabajo.

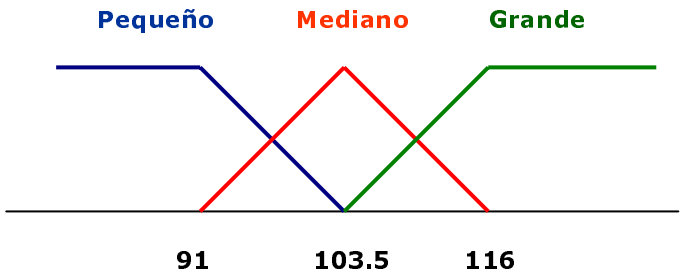


Fig 0.2. Conjunto de ejemplos precisos e imprecisos

Por todo lo mencionado anteriormente y otras cuestiones que sería demasiado extenso comentar aquí, surge la necesidad de abordar un problema abierto como es la elección de la mejor partición borrosa en conjuntos de datos imprecisos, como base para demostrar la eficacia de la nueva definición de Información Mutua propuesta.

Objetivos

El principal objetivo de esta tesis es presentar una nueva definición de información mutua basada en la definición clásica. Esta nueva definición se utilizará para aplicarla a un problema concreto de los muchos existentes. A partir de una partición definida de manera uniforme, mediante esta nueva definición obtendremos nuevas particiones optimizadas. El objetivo será demostrar que estas particiones optimizadas, haciendo uso de esa nueva definición, ofrecen un error de clasificación menor o igual que las particiones uniformes, utilizando sistemas de clasificación basados en reglas borrosas.

Asimismo se pretende ampliar aún mas el campo de estudio, abarcando el tratamiento de datos imprecisos, en los que la información es nula o imprecisa. Demostraremos que esta definición es aplicable a problemas de este tipo.

La principal contribución de la tesis, por lo tanto, será la definición de una nueva medida de información mutua aplicable a variables aleatorias borrosas, junto con un algoritmo numérico, basado en algoritmos evolutivos,

que sirva para obtener un conjunto de particiones optimizadas que proporcione mejores resultados de clasificación.

Metodología

Para completar los objetivos propuestos en el apartado anterior, se realizará un trabajo de investigación consistente en un estudio bibliográfico exhaustivo acerca del diseño de particiones borrosas, así como diferentes formas de utilizar la información mutua como medida de optimización.

En un momento inicial se plantea una definición de Información Mutua satisfactoria en cuanto a resultados, pero en problemas pequeños. En el momento en que nos íbamos a problemas de tamaño medio y grande los resultados ya no eran los esperados y en consecuencia hubo que modificar dicha definición. Como consecuencia de esto ocurrió que si bien es cierto que en la mayor parte de los problemas propuestos hubo mejoría utilizando la nueva definición, en otros casos los resultados empeoraron, cosa que era de esperar por la naturaleza de los mismos. Existen problemas que debido a su naturaleza y número de atributos son casi imposibles de tratar en algunos casos, como por ejemplo cuando el número de particiones sea elevado.

En este punto mostramos el método propuesto por nosotros, así como su aplicación para la optimización de particiones borrosas tanto con datos precisos como con datos imprecisos, punto este último, objetivo primordial.

Como la nueva definición se va a aplicar al problema de optimización de particiones, se estudiará el diseño de particiones lingüísticas aplicadas al aprendizaje de clasificadores borrosos en reglas. En este punto se estudiará tanto los algoritmos que adaptan las particiones como aquellos que no las adaptan, englobado en el este último punto la labor que el experto puede realizar en ese campo.

Se realizará también, un estudio de algoritmos de clasificación existentes en la literatura, necesarios para llevar a cabo nuestros experimentos. Dichos algoritmos nos servirán para demostrar la efectividad de nuestro método, por lo tanto serán nuestra herramienta experimental.

Asimismo es necesario el estudio e implementación de algoritmos genéticos tanto para la optimización con datos precisos como con datos imprecisos. De esta forma y con los resultados de los experimentos

podremos demostrar las mejoras obtenidas en los experimentos realizados. De estos algoritmos no esperamos rendimiento, tan sólo que cumplan el propósito de la optimización.

Organización

Este documento se va a organizar en cinco partes bien diferenciadas, que pasaremos a comentar a continuación

En la ***primera parte***, en la cual se encuentra el lector, trataremos de aportar las razones suficientes y necesarias para llegar a la conclusión de que la definición de la mejor partición borrosa para un sistema de clasificación basado en reglas borrosas, es un problema abierto y por lo tanto debe ser tratado e investigado, no para echar por tierra los resultados ya obtenidos por otros investigadores en este campo sino para aportar mas soluciones si cabe, apoyándonos siempre en los resultados ya obtenidos.

En la ***segunda parte***, correspondiente a los capítulos 1 y 2, se tratará de definir los conceptos necesarios para que se pueda seguir todo el contenido del desarrollo. Se analizan las técnicas ya existentes y se justifica la necesidad de investigar nuevas soluciones.

En la ***tercera parte***, correspondiente al capítulo 3, abordaremos nuestro método propuesto, planteando las posibles soluciones y comparándolas con las soluciones ya obtenidas por otros investigadores. Así mismo ilustraremos mediante un ejemplo fácil de seguir para el lector, como el método propuesto consigue los resultados esperados.

En la ***cuarta parte***, correspondiente al capítulo 4, se mostrarán las distintas pruebas realizadas, así como los conjuntos de datos para la realización de las mismas. Se mostrarán los resultados numéricos fruto de las mismas y su explicación textual.

En la ***quinta parte***, correspondiente al capítulo 5, se expondrán las conclusiones finales a las cuales se ha llegado tras el análisis y estudio del problema abordado. Asimismo, se propondrán posibles líneas de investigación futuras sobre las que sería interesante seguir trabajando.

CAPITULO 1

Clasificador descriptivo basado en reglas borrosas

1.1. Introducción

En esta parte se muestran los conceptos de un clasificador basado en reglas, así como los elementos del mismo, destacando aquellas partes en las que vamos a centrar el estudio de nuestro trabajo.

Asimismo se realiza un recorrido por algunos de los algoritmos de clasificación existentes que se utilizarán para realizar la experimentación de la que se desprenden nuestros resultados finales.

1.2. Clasificador borroso descriptivo

Un sistema de clasificación basado en reglas borrosas esta formado por una ***base de conocimiento*** y un método ***de razonamiento difuso ó borroso***, tal que para una entrada dada, determina la clase a la que pertenece [Jes99].

Se le llama descriptivo porque su base de conocimiento, de la cual forma parte las reglas que representan el problema, puede expresarse mediante etiquetas lingüísticas, asociadas a funciones de pertenencia, que identifican de forma mas o menos clara el conocimiento que en algún momento nos ayudará a clasificar (de forma correcta o no) un nuevo ejemplo.

La base de conocimiento, está formada por **la base de datos**, que contiene la definición de los conjuntos borrosos asociados a los términos o etiquetas lingüísticas utilizadas en la **base de reglas**, y cuyo contenido es un conjunto de reglas de clasificación.

$$R=\{R_1,\ldots,R_L\} \quad con \;\; L:1\ldots n$$

La forma general de una regla de la base de reglas será la siguiente:

si *antecedente* **entonces** *consecuente con peso o seguridad*

El antecedente de la regla estará formado por los distintos pares atributo-etiqueta lingüística, utilizadas estas últimas para discretizar los dominios continuos de las variables. El número de atributos variará dependiendo del problema que se este tratando y el número de etiquetas lingüísticas por atributo también variará

En el consecuente se puede incluir una o más variables clase, que pueden ir asociadas a una certeza. Cuando en el consecuente sólo se indica una clase, su certeza puede interpretarse como el peso de la regla [INT92, NIT96].

Según [CHHM01], podemos describir tres tipos de reglas:

- Reglas borrosas con una clase en el consecuente pero sin grado certeza. Son de la forma:

  si X_1 *es* A_1 *y ...* *y* X_N *es* A_N **entonces** Y *es* C_j

  donde N es el número de atributos y j va de 1 al numero de clases. González y Pérez en [GP98], Kuncheva en [Kun96] y Nauck y Kruse en [NK97], entre otros, desarrollan sistemas de clasificación basados en reglas borrosas de este tipo de reglas.

- Reglas borrosas con una clase y un grado de certeza asociado a la clasificación para esa clase. Son de la forma:

  si X_1 *es* A_1 *y ...* *y* X_N *es* A_N **entonces** Y *es* C_j con certeza r

  donde N es el número de atributos y j va de 1 al numero de clases. Este tipo de reglas son empleadas por Ishibuchi y otros en [INT92, NIT96] y son utilizadas también en algoritmos de boosting borroso [JHJS04, JS00].

- Reglas borrosas que indican en el consecuente, el grado del certeza para cada una de las clases existentes. Son de la forma:

  si X_1 *es* A_1 *y ...* *y* X_N *es* A_N **entonces** $(r_1,\ldots,r_M)$

  donde N es el número de atributos y M es el numero de clases, de tal forma que $(r_1,\ldots,r_M)$ son los grados de certeza para cada una de las clases. Este tipo de reglas se utilizan en los sistemas de clasificación basados en reglas borrosas propuestos por Mandal y otros en [MMP92, PM92] y también en el algoritmo Logitboost [OS06].

El método de razonamiento borroso es un procedimiento de inferencia que se utiliza para predecir una clase ante un ejemplo no clasificado. Tradicionalmente, en la literatura se ha utilizado el método de

razonamiento borroso del máximo, también denominado de la regla ganadora, que considera la clase indicada por una sola regla, aquella con la que el ejemplo tiene mayor grado de compatibilidad, calculado este a partir de un determinado procedimiento. No obstante, también suele ser frecuente la inferencia mediante suma de votos [Kun00, INM99, JHJS04, JS00, OS06].

En Cordón y otros [CJH99] y en Ishibuchi y otros [INM99], se discuten muchos otros métodos de inferencia borrosa pero nosotros tan sólo nos vamos a quedar con dos métodos que utilizan reglas lingüísticas: uno de ellos es el de la regla ganadora, solamente una regla se aplica para determinar la clase del ejemplo. El otro es un método basado en votaciones.

a) *Método de la regla ganadora:* La regla que se aplicará en este método será aquella que proporciona el máximo producto entre el grado de compatibilidad del ejemplo con el antecedente de la regla y el peso de la misma. Si ocurre que hay dos reglas con consecuentes distintos que dan el mismo valor, el ejemplo no se puede clasificar.

b) *Método basado en votaciones:* Cada regla vota por su consecuente con tantos votos como el producto del grado de pertenencia de antecedente del ejemplo con la regla y el peso. La clase mas votada será la clase que el clasificador asigne al ejemplo

No se puede decir cual de los dos métodos es mejor pues en cada caso será adecuado uno u otro, pero lo que si se puede decir es que el método basado en la regla ganadora es el más intuitivo a la hora de explicar porqué una regla clasifica a un nuevo ejemplo en una clase determinada.

1.3. Información lingüística

Cuando la extracción de reglas en problemas reales implica la utilización de atributos continuos, normalmente estos últimos pasan por un proceso de discretización. De esta forma atributos como el peso o la altura, se discretizan en intervalos. En algunos casos la división es muy clara pero en otros casos no lo es tanto. Si consideramos por ejemplo la edad y pensamos en “mayores de edad”, sabemos que si una persona tiene 18 años o mas, es mayor de edad y en caso contrario no lo es. Es el caso de un intervalo perfectamente definido.

En la mayor parte de las situaciones de la vida cotidiana, no se puede realizar intervalos perfectamente definidos si no que a la hora de discretizar lo hacemos de una forma borrosa. Además en la vida real no hablamos con valores numéricos, es decir, no decimos "mi amigo mide 1.80" o "la hermana de María pesa 100 Kg", sino que lo que generalmente se dice es "mi amigo es alto" ó "la hermana de María es gorda" respectivamente. Tanto el término ó etiqueta lingüística "alto" como "gorda" abarcan un rango de valores que se traduce a un intervalo cuyos límites no son tan claros.

Para manejar problemas de clasificación y modelado, se construyen reglas del tipo *Si-entonces* como se mostró en el apartado anterior. La ventaja de utilizar reglas con antecedentes que tienen etiquetas lingüísticas es que se pueden interpretar fácil e intuitivamente ya que el lenguaje empleado es el mismo que se utilizan en conversaciones del día a día.

Para describir un atributo continuo, se utilizan etiquetas lingüísticas, que se corresponderán con las funciones de pertenencia que describen su significado. El número de etiquetas lingüísticas variará dependiendo del atributo continuo y del problema a tratar. Los datos numéricos se sustituirán por las etiquetas lingüísticas correspondientes.

A la hora de diseñar las funciones de pertenencia se presentan dos opciones: realizar una división homogénea de los datos, por ejemplo, mediante funciones triangulares o trapezoidales, o bien realizar divisiones no homogéneas. En algunos casos, la primera forma puede resultar más intuitiva y fácil de comprender, pero no siempre es así. Se supone que la discretización de atributos continuos se realiza tanto en los atributos de entrada como en los atributos de salida.

En la siguiente subsección se muestra con detalle el diseño de una función de pertenencia.

1.3.1. Función de pertenencia

Si pensamos en problemas de la vida real, nos damos cuenta que en muchas ocasiones es muy difícil determinar si algo esta frío o caliente o si algo es grande o pequeño por ejemplo.

La mejor forma de solucionar este problema es considerar la pertenencia de un elemento a un conjunto, y esta pertenencia no es absoluta sino que es gradual. Hablamos por lo tanto de que ese conjunto es borroso

[Zad75, Men95, RN95]. Su función característica, llamada también de pertenencia, no adoptará valores en el conjunto {0,1}, sino en el intervalo cerrado [0,1].

Formalmente hablando, se define un conjunto borroso A como el conjunto de pares

$$A = \{(x, \mu_A(x)) \mid x \in S\}$$

donde S es el universo del discurso y μ_A es la función de pertenencia:

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

de tal modo que $\mu_A(x) \in [0,1]$ es el grado con el que un elemento $x \in S$ (siendo S el universo del discurso) pertenece al conjunto borroso A. Cuando $\mu_A(x) = 0$ el elemento no pertenece al conjunto y cuando $\mu_A(x) = 1$ pertenece totalmente.

La forma de la función de pertenencia es un tanto subjetiva y esta depende de los conceptos que se quieran representar con ella. La función de pertenencia puede adquirir distintas formas y en ocasiones se pueden seleccionar con cierto grado de libertad por parte del diseñador, lo que en la práctica se traduce a introducir cierto conocimiento experto, como se verá en el capítulo 2.

En general se tiende a trabajar con las funciones de pertenencia definidas de forma paramétrica [Str00]. Las parametrizaciones mas frecuentes son:

1. Funciones Gaussianas o con forma de S que utilizan la formula siguiente:

$$\mu(x) = \frac{1}{s\sqrt{2\pi}} \exp\left(\frac{(x-m)^2}{2d^2} \right)$$

y que tienen el siguiente aspecto:

Fig 1.1. Función de pertenencia Gaussiana

Permiten modelar fácilmente modificadores pero son complicadas para el cálculo

2. Funciones triangulares o trapezoidales que son sencillas de manejar en algoritmos numéricos

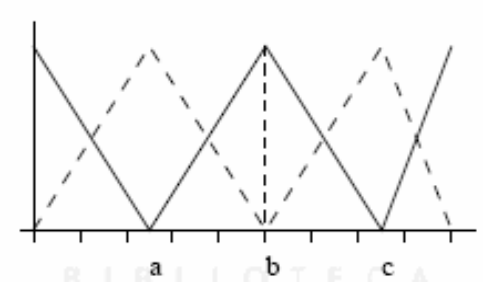


Fig 1.2. Función de pertenencia triangular

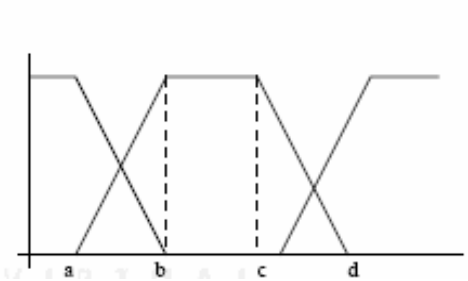


Fig 1.3 Función de pertenencia trapezoidal

Zeng y Zingh [ZZ94] definen un modelo de función de pertenencia que agrupa a las principales clases, la función Pseudo-trapezoidal.

La función Pseudos-trapezoidal es una función continua dada por:

$$A(x,a,b,c,d,h)=\begin{cases} I(x) & x\in[a,b] \\ h & x\in[b,c] \\ D(x) & x\in[c,d] \\ 0 & x\in U-[a,d]=\{x\mid x\in U, x\notin[a,d]\} \end{cases}$$

donde $a \le b \le c \le d$, $a<d$, $I(x) \ge 0$ es una función monótona estrictamente creciente en [a,b) y $D(x) \ge 0$ es una función monótona estrictamente decreciente en (c,d], como se puede observar en la figura que se muestra a continuación.

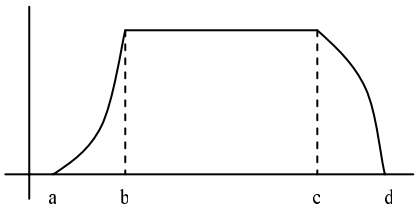


Fig 1.4. Función de pertenencia pseudo trapezoidal

Cuando la función de pertenencia de un conjunto borroso A es una función pseudo-trapezoidal, se llama función de pertenencia pseudo-trapezoidal y se denota por $A(x) = (x; a, b, c, d, h)$. Cuando el conjunto borroso es normal, es decir $h = 1$, su función de pertenencia se denota por $A(x) = (x; a, b, c, d)$.

Las funciones trapezoidales son un caso especial de las funciones pseudo-trapezoidales cuando $b < c$ y ocurre que:

$$I(x) = \frac{x-a}{b-a}$$

y

$$D(x) = \frac{x-d}{c-d}$$

Por otra parte, las funciones triangulares es un caso especial cuando $b = c$ y

$$I(x) = \frac{x-a}{b-a} = \frac{x-a}{c-a}$$

y

$$D(x) = \frac{x-d}{c-d} = \frac{x-d}{b-d}$$

1.3.2. Partición borrosa

Una partición, desde el punto de vista clásico, es una familia de subconjuntos $A_1, A_2, \ldots, A_n$ de un conjunto A perteneciente, este último, a un universo Ω, que cumple que:

a) cada uno de los subconjuntos es distinto del conjunto vacío,

b) la unión de los subconjuntos es, aproximadamente, el conjunto particionado y

c) la intersección entre dos conjuntos cualesquiera es el conjunto vacío.

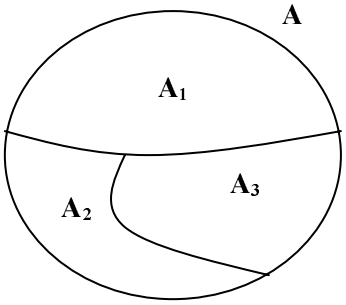


Fig 1.5. Partición borrosa

	(a)	(b)
		$A_1 \cup A_2 \cup A_3 = A$
$A_1 \neq \phi$		$A_1 \cap A_2 = \phi$
$A_2 \neq \phi$		$A_1 \cap A_3 = \phi$
$A_3 \neq \phi$		$A_2 \cap A_3 = \phi$
		(c)

Tabla 1.1. Propiedades

Este tipo de particiones no tiene porque ser el más adecuado para todo tipo de problemas y así surge el particionamiento borroso propuesto por Ruspini [Rus69] en el campo de la teoría de la probabilidad utilizando conjuntos borrosos.

Una partición borrosa es una representación de un conjunto de valores de una variable al que se le puede asignar una etiqueta lingüística de modo que la interpretabilidad de la información sea mayor. A su vez para una determinada variable surge la necesidad de representar dichos valores asociados a la variable con más de una partición que suelen solaparse. Al conjunto de particiones se le denomina partición borrosa y al conjunto de etiquetas lingüísticas asociadas a las particiones se le llama variable lingüística.

Al igual que ocurre con una partición clásica, una partición borrosa cumple que:

- cada uno de los subconjuntos es distinto del conjunto vacío,
- la unión de los subconjuntos es el conjunto particionado,

pero en este caso,

- la intersección entre dos conjuntos cualquiera puede no ser el conjunto vacío.

Este tipo de particiones lleva asociado el concepto de función de pertenencia explicado en el apartado anterior y que como se explicó nos proporciona el grado de pertenencia de un valor de la variable a una o varias particiones de la partición borrosa. En esta tesis utilizarán exclusivamente particiones borrosas de Ruspini [Rus69] para las que cualquier valor pertenece como máximo a dos etiquetas distintas y adyacentes, que se solapan, y cumplirán que la suma de esos grados de pertenencia será 1.

1.3.2.1. Ejemplo de partición borrosa

Como ilustración, en la siguiente figura se muestra un ejemplo de una partición borrosa donde el concepto que se intenta representar es una descripción de lo que se entiende por *tamaño de las frutas* para las cuales tenemos el valor de su peso.

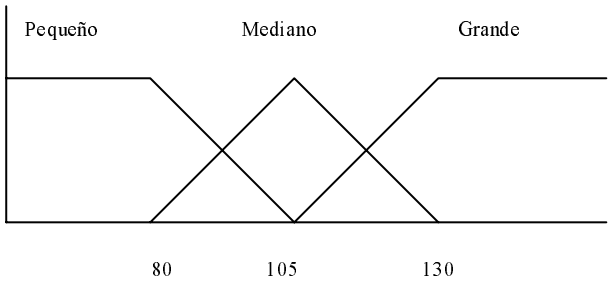


Fig 1.6. Partición borrosa *"tamaño de frutas"*

A cada conjunto borroso del conjunto de particiones se le asocia una función de pertenencia, que determina en cada caso el grado de pertenencia de un valor de un atributo al conjunto borroso correspondiente. Por ejemplo, el conjunto borroso "Pequeño" representa la posibilidad de que una pieza de fruta, con un determinado peso, sea caracterizada como "Pequeño".Un conjunto borroso podría ser considerado una distribución de posibilidad, que a sus vez puede relacionarse con cierta familia de distribuciones de probabilidad [DP91].

Como se puede observar los conjuntos borrosos se superponen de tal forma que cabe la posibilidad de que un valor de un atributo pueda caer dentro de dos conjuntos contiguos distintos. Dicho valor tendrá por lo tanto dos grados de pertenencia cuya suma valdrá 1 ya que hay que tener en cuenta que trabajamos con particiones de Ruspini [Rus69], como se ha mencionado en el apartado anterior. Por lo tanto, una pieza de fruta x, podría tener un grado de pertenencia en dos particiones a la vez, por ejemplo "Pequeño" y "Mediano", indicando que posee cualidades asociadas con ambas.

1.4. Algoritmos de clasificación utilizados en la experimentación

Una de las formas más básicas de determinar el consecuente de una regla borrosa, dado un conjunto de ejemplos de entrenamiento, es asociar a cada antecedente la clase de los ejemplos compatibles con él. En el caso de que estos ejemplos pertenezcan a clases distintas, se elige la clase más dominante.

En realidad una forma intuitiva y sencilla de ver esto es de forma gráfica. Si tenemos dos atributos continuos con sus etiquetas lingüísticas respectivas y reglas que utilizan esos atributos continuos y determinan una clase, todo esto se puede mostrar de forma gráfica mediante la representación en el plano x-y. Vamos a ver un ejemplo.

Supongamos que tenemos dos atributos continuos, normalizados en [0,1], que describen el tamaño de algo y que dependiendo del tamaño, el nuevo objeto a clasificar es de una clase u otra. Supongamos que cada uno de los atributos continuos tiene tres etiquetas lingüísticas (pequeño, mediano y grande) y que hay dos clases. Tenemos además 30 patrones o ejemplos de

entrenamiento y sabemos para cada valor de los atributos continuos de los ejemplos que etiqueta lingüística le corresponde. Asimismo sabemos que de los 30 patrones, 15 son de una clase y 15 de otra.

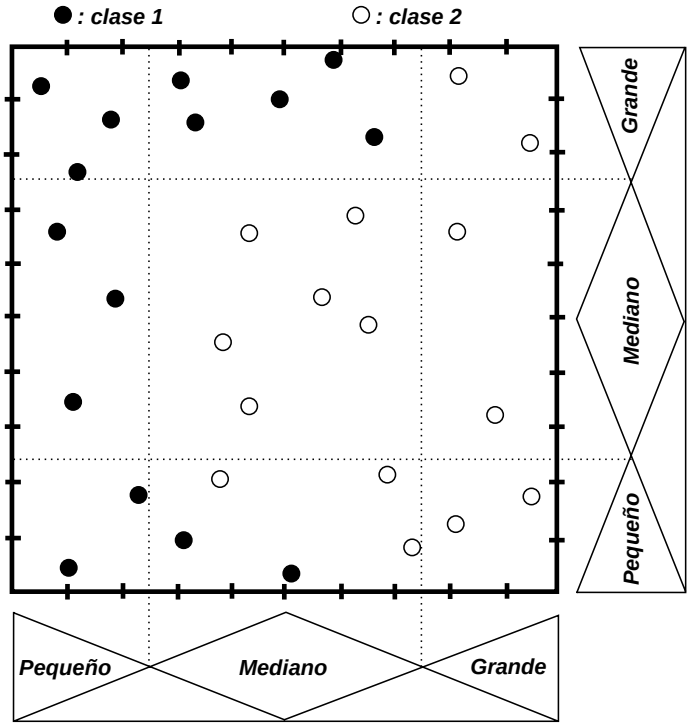


Fig 1.7. Representación gráfica de las reglas

De forma intuitiva y considerando la partición ofrecida en la figura 1.6., se pueden extraer las reglas que se muestran a continuación. Hay que tener en cuenta que estas reglas dependerán de las particiones establecidas.

Regla 1: si x1 es small y x2 es small entonces Class1
Regla 2: si x1 es small y x2 es medium entonces Class1
Regla 3: si x1 es small y x2 es large entonces Class1
Regla 4: si x1 es medium y x2 es small entonces Class1
Regla 5: si x1 es medium y x2 es small entonces Class 2
Regla 6: si x1 es medium y x2 es medium entonces Class2
Regla 7: si x1 es medium y x2 es large entonces Class1
Regla 8: si x1 es large y x2 es small entonces Class2
Regla 9: si x1 es large y x2 es medium entonces Class2

Regla 10: si x1 es large y x2 es large entonces Class2

Cuando se plantea el problema de clasificar un nuevo ejemplo con dos valores de 'x1' y de 'x2' tales que haya la posibilidad de que pertenezca a las dos clases. En este caso la clase dominante será la ganadora. Por ejemplo si x1=0.5 y x2=0.1 el ejemplo puede pertenece a las clase 1 o a la clase 2 si se observa la Fig 1.6. La clase dominante es la clase2.

1.4.1. Esquema de algoritmos de clasificación a estudiar en las siguientes secciones

A continuación se muestra un esquema de los algoritmos que se van a estudiar en secciones sucesivas y que van a servir de apoyo en nuestra experimentación. Se ha tomado como referencia a Ishibuchi [INN04] para llevar a cabo la clasificación propuesta en este apartado.

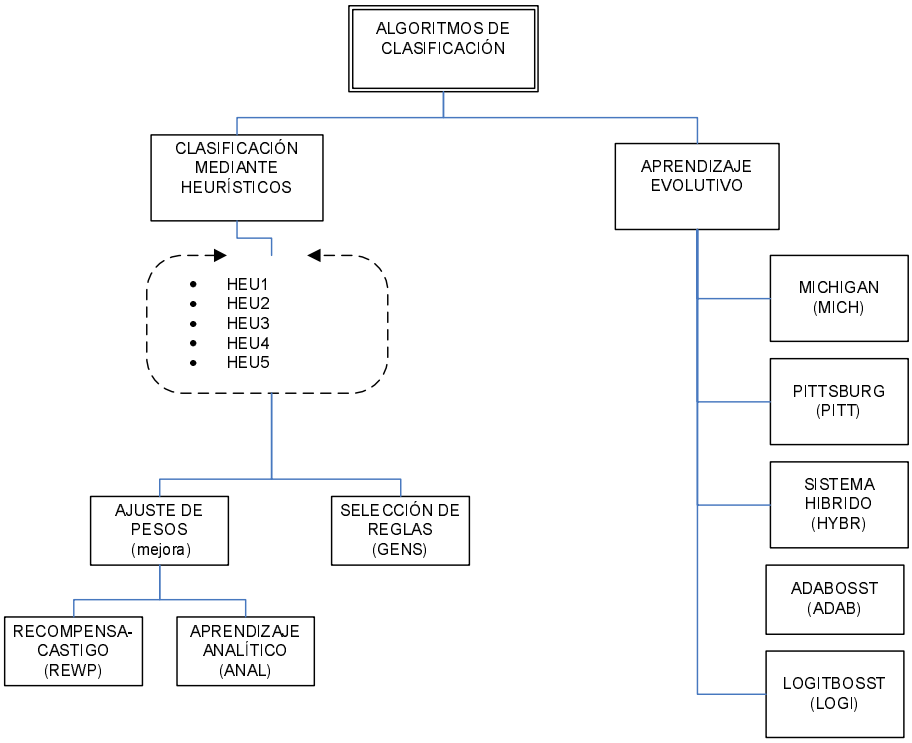


Fig 1.8. Algoritmos de clasificación utilizados en la experimentación

1.4.2 Clasificación de nuevos ejemplos mediante heurísticos

Se ha tomado como referencia a Ishibuchi para llevar a cabo la clasificación propuesta en este apartado.

El proceso para obtener el consecuente pasa por calcular el grado de compatibilidad del patrón $x_p = (x_{p1}, \ldots, x_{pn})$ con el antecedente de la regla lingüística.

Recordamos que nosotros trabajaremos con reglas lingüísticas del tipo:

si X_1 *es* A_1 *y ... y* X_N *es* A_N **entonces** Y *es* C_j

a) Primeramente hay que calcular el grado de compatibilidad de cada valor del atributo x_{pi} con la etiqueta lingüística A_{qi}. Esto se calcula mediante la función de pertenencia asociada a la etiqueta lingüística.

b) En un segundo paso hay que calcular el grado de compatibilidad del antecedente del ejemplo o patrón de entrenamiento. En realidad, no es más que operar con los grados de compatibilidad obtenidos en el apartado anterior. Se pueden utilizar dos operadores, o bien el operador mínimo de los distintos grados de compatibilidad de los distintos valores de atributos del ejemplo o patrón, o bien el producto de los mismos. En realidad se podría usar cualquier t-norma, pero el mínimo y el producto son los mas frecuentes.

Supongamos que D es el conjunto de ejemplos de entrenamiento, $D : D = (x_1, \ldots, x_m)$, la cardinalidad de D es el número de ejemplos del conjunto, esto es, m.

Sea $D(A_q)$ el conjunto borroso de ejemplos del conjunto de entrenamiento compatibles con el antecedente A_q de la regla lingüística R_q.

El grado de compatibilidad con el antecedente se calcula como:

$$| D(A_q) | = \sum_{p=1}^{m} \mu_{A_q}(x_p)$$

Así mismo se puede calcular el grado de compatibilidad de cada ejemplo de entrenamiento $\mathbf{x}_p$ con la regla lingüística $\mathbf{R}_q$ de la siguiente manera. Hay que tener en cuenta que se considera el antecedente y el consecuente.

$$\mu_{R_q}(x_p) = \begin{cases} \mu_{A_q}(x_p), si & p \in C_q \\ 0, si & p \notin C_q \end{cases}$$

Si $D(A_q) \bigcap D(C_q)$ es el conjunto borroso de la compatibilidad de los ejemplos de entrenamiento teniendo en cuenta el antecedente y el consecuente, el grado total de compatibilidad con la regla lingüística $\mathbf{R}_q$ se calcula como

$$| D(A_q) \bigcap D(C_q) | = \sum_{p=1}^{m} \mu_{R_q}(x_p) = \sum_{p \in C_q} \mu_{A_q}(x_p)$$

que básicamente nos viene a indicar el grado de compatibilidad del antecedente cuando la clase es una en concreto.

En el campo de la minería de datos [AMSTV96, AS94] se utilizan dos conceptos fundamentales, "seguridad" y "soporte". Estos dos conceptos se pueden extender al caso de las reglas lingüísticas [HKC01, IYN01].

El concepto de "seguridad" se puede describir como el cociente entre el grado de compatibilidad de un ejemplo con aquellas reglas que encajan en el antecedente y son de una determinada clase y el grado de compatibilidad de aquellas reglas que encajan en el antecedente independientemente de la clase.

Por otra parte, el concepto de "soporte" se puede describir como el cociente entre el grado de compatibilidad de un ejemplo con aquellas reglas que encajan en el antecedente y son de una determinada clase y el número total de ejemplos de entrenamiento.

El concepto de seguridad se puede utilizar como el peso CF_q de la regla lingüística R_q. Así una primera aproximación al cálculo del peso de una regla es la que se daba en el apartado anterior.

$$CF_q^I = c(A_q \Rightarrow C_q) = \frac{|D(A_q) \cap D(C_q)|}{|D|} = \frac{\sum_{p \in C_q} \mu A_q(x_p)}{m}$$

Ishibuchi y otros [INT92] usan varias definiciones heurísticas diferentes para asignarle peso a las reglas. En la primera, a la fórmula anterior le resta el grado medio de seguridad de aquellas reglas lingüísticas que encajan en el antecedente del ejemplo pero tienen un consecuente o clase distintos.

$$CF_q^{II} = c(A_q \Rightarrow C_q) - C_{Media}$$

Una segunda definición para los pesos es la que resta a la fórmula inicial, el segundo grado de seguridad mayor de aquellas reglas que encajan en el antecedente, para las distintas clases. Esto es:

$$CF_q^{III} = c(A_q \Rightarrow C_q) - C_{Segunda}$$

Por último, una tercera forma es aquella es la que se le resta a la fórmula inicial la suma grado de seguridad de aquellas reglas lingüísticas que encajan en el antecedente del ejemplo pero tienen un consecuente o clase distintos.

$$CF_q^{IV} = c(A_q \Rightarrow C_q) - C_{Suma}$$

Hay que tener en cuenta que si solo tenemos dos clases, las definiciones 2, 3, y 4 dan el mismo resultado. Solo cuando tenemos un número de clases grande, se puede hablar de cambios significativos.

Y en general se puede decir que:

$$CF_q^{IV} \leq CF_q^{III} \leq CF_q^{II} \leq CF_q^I$$

1.4.2.1. Mejoras añadidas a la clasificación: ajuste de pesos

La idea básica de este esquema de aprendizaje es decrementar o incrementar el peso de una regla ganadora de acuerdo con la clasificación resultante, correcta o incorrecta, de cada ejemplo de entrenamiento [NIT96].

Así, se distinguen dos métodos para conseguir el objetivo mencionado.

- Método de recompensa-castigo
- Método analítico

1.4.2.1.1. Método de recompensa-castigo

Si consideramos el conjunto de reglas lingüísticas con los pesos correspondientes, este conjunto puede verse como un sistema basado en reglas lingüísticas. Cada nuevo ejemplo que queremos clasificar está representado mediante esta base de reglas. Suponiendo que aplicamos el método basado en la regla ganadora, el aprendizaje consistiría en penalizar aquellas reglas que clasifican mal a un nuevo ejemplo y premiar a las que los clasifican bien.

El premio de la nueva regla consiste en la actualización del peso de la misma y vendría dada por la siguiente expresión,

$$CF_w^{Nueva} = CF_w^{Vieja} + \eta^+ * (1 - CF_w^{Vieja}),$$

donde η^+ es una constante positiva entre 0 y 1. El peso de la regla se incrementa pudiendo llegar a tener el valor 1.

Por otro lado la penalización consiste en una disminución del peso mediante la siguiente expresión,

$$CF_w^{Nueva} = CF_w^{Vieja} - \eta^- * CF_w^{Vieja},$$

donde η^- es una constante positiva entre 0 y 1. El peso de la regla se decrementa pudiendo llegar a tener el valor 0.

Tan solo se premia o penaliza a la regla ganadora, ya que en el caso de que existan dos reglas ganadoras con distinto consecuente el ejemplo no se clasifica y no se actualiza nada.

El diseño del algoritmo consistirá en seleccionar un conjunto de entrenamiento y clasificar dicho conjunto de entrenamiento con la base de reglas. En este proceso se actualizarán los pesos de las reglas según proceda.

El algoritmo se repetirá si no se satisface la condición de parada. Normalmente el algoritmo volverá a seleccionar otro conjunto de entrenamiento, en otro orden distinto y volverá a realizar todo el proceso. La condición de parada suele ser que todos los conjuntos de entrenamiento sean correctamente clasificados.

1.4.2.1.2. Método de aprendizaje analítico

Cuando un ejemplo de entrenamiento es clasificado por nuestro sistemas de reglas, el método de la regla ganadora, identifica aquella que proporciona el máximo producto entre el grado de compatibilidad del antecedente y el peso de la regla, de entre todas las existentes.

Dada la regla ganadora R_w con clase C_w, un ejemplo de entrenamiento x_q, con clase conocida t_q, es clasificado correctamente, si la clase de la regla coincide con la clase del ejemplo de entrenamiento. En este caso se cumple que:

$$\max\{\mu A_q(x_p).CF_q \mid C_q = t_q, R_q \in S\} = \mu A_w(x_p).CF_w$$

Por otra parte, un ejemplo no se clasifica correctamente si la clase de la regla no coincide con la clase del ejemplo de entrenamiento. En este caso se cumple que:

$$\max\{\mu A_q(x_p).CF_q \mid C_q = t_q, R_q \in S\} < \mu A_w(x_p).CF_w$$

Se buscará en este caso, aquellas reglas que coinciden con el consecuente del ejemplo a clasificar y se seleccionará como mejor candidata, aquella R_{q^*} que tiene el máximo producto entre la pertenencia del antecedente y el peso de la regla y entonces se puede decir que:

$$\mu A_{q^*}(x_p).CF_{q^*} < \mu A_w(x_p).CF_w \quad (1)$$

Para clasificar correctamente el ejemplo hay que hacer que la desigualdad mostrada anteriormente cambie de sentido y esto sólo se puede hacer ajustando las el peso de las reglas.

Si el producto de las pertenencias por el peso de esta regla candidata es cero es que no hay clasificación posible, así que no se hace nada, pero en el caso de que sea mayor que cero pueden ocurrir dos casos:

a) Incrementa el peso CF_{q^*} de la regla R_{q^*} en la parte izquierda de la desigualdad (1) de la forma:

$$CF_{q^*}^{Nueva} = \frac{\mu A_w(x_p)}{\mu A_{q^*}(x_p)}.CF_w^{Vieja} + \varepsilon$$

b) Decrementar el peso CF_w de la regla R_w en la parte derecha de la desigualdad (1) de la forma

$$CF_{q^*}^{Nueva} = \frac{\mu A_{q^*}(x_p)}{\mu A_w(x_p)}.CF_{q^*}^{Vieja} - \varepsilon$$

donde ε es un número real positivo, muy pequeño.

El proceso termina cuando todos los ejemplos de entrenamiento están bien clasificados ó cuando en dos iteraciones no se cambia ninguna regla.

1.4.2.2. Mejoras añadidas a la clasificación: selección de reglas

El uso de un algoritmo genético es uno de los métodos propuesto por Ishibuchi y otros [INYT94, INYT95], para seleccionar un número pequeño de reglas lingüísticas de un conjunto grande de regla.

Este consiste en dos partes. La primera consiste en generar las reglas candidatas utilizando alguno de los métodos heurísticos mencionados con anterioridad. Se seleccionarán aquellas reglas con dos o menos de dos atributos en el antecedente.

La segunda parte consiste en la selección de una pequeña parte de las reglas lingüísticas del total de las reglas utilizando un algoritmo genético [Gol89, Hol75].

En este caso cada individuo será una base de reglas lingüísticas que consistirá en una cadena de tantos 1 y 0 como reglas existan en la base de reglas, de tal forma que el 1 significa que se incluye la regla y el 0 que no se incluye.

Para evaluar el fitness de un individuo se calcula el número de ejemplos correctamente clasificados de un conjunto de entrenamiento y se le resta una constante multiplicada por el número de reglas lingüísticas representadas por el individuo. Esto es:

$$fitness(S) = NECC(S) - const^* \mid S \mid$$

La función de la segunda parte de la expresión es penalizar con respecto al número de reglas para encontrar el menor número de reglas lingüísticas con un alto grado de acierto en la clasificación.

1.4.3. Aprendizaje evolutivo

En esta sección se explicará como los algoritmos genéticos, servirán como herramienta parar diseñar sistemas de clasificación basados en reglas lingüísticas. Los algoritmos genéticos para aprendizaje se suelen dividir según tres enfoques muy distintos: el enfoque Michigan, el enfoque Pittsburg y el enfoque de aprendizaje iterativo de reglas.

En el enfoque Michigan cada regla se representa como una cadena y es un individuo y una población de cadenas se correspondería con la base de reglas. Por otro lado, con el enfoque Pittsburg [Smi80] toda la base de reglas es una única cadena y el individuo es toda la base reglas.

En definitiva, mientras que en el enfoque Michigan lo que se seleccionan son las mejores reglas, en el enfoque Pittsburg lo que se seleccionan son las mejores bases de reglas, es decir, se optimiza mediante la función fitness la base de reglas, cosa que no se consigue con el enfoque Michigan ya que se trata a cada regla como un individuo.

El enfoque de aprendizaje iterativo de reglas considera, al igual que en el enfoque Michigan, que cada individuo representa una regla individual

pero, a diferencia de éste, aporta como solución parcial, el problema de aprendizaje y necesita ejecutarse varias veces para generar una base de conocimiento completa.

En Cordón y otros [CHHM01], se muestran diferentes algoritmos genéticos de aprendizaje desde el punto de vista del aprendizaje iterativo de reglas, así como los enfoques Michigan y Pittsburg.

1.4.3.1. Enfoque Michigan

En el enfoque Michigan cada individuo de la población es una cadena de caracteres que representa el antecedente de la regla [INM95]. Si tenemos la siguiente regla:

Si x_2 es pequeño y x_4 es mediano y x_5 es grande entonces clase C_q con seguridad CF_q,

tal que cada regla tiene cinco atributos y cada atributo tiene tres etiquetas lingüísticas (pequeño, mediano y grande) y además si no hay valor para el atributo se le asigna un carácter.

Suponiendo que a cada etiqueta se le asigna un valor para poder representarla como individuo (pequeño=1, mediano=2 y grande =3) y no tener valores se representa con el símbolo #, entonces la regla se podría codificar como

#	1	#	2	3

Lo primero que hay que hacer es seccionar la población inicial de nuestro problema n-dimensional (n será 4 si tenemos 4 atributos por ejemplo). Se generan M cadenas (reglas codificadas) de longitud n de forma aleatoria seleccionando uno de los 4 símbolos propuestos (o los que sea en cada caso) con probabilidad ¼.

Inicialmente tendremos que saber tanto el número de reglas a sustituir, la probabilidad de cruce, la probabilidad de mutación y la condición de parada, que puede ser un número de iteraciones deseado.

Seguidamente se evalúa el fitness para la población de reglas. Para cada ejemplo de entrenamiento, se sabrá que regla es la mejor para su clasificación, mediante el método de la regla ganadora. Si dicha regla clasifica bien el ejemplo recibe un voto. Al final todos los ejemplos habrán

sido clasificados, y sabremos el número de ejemplos que ha clasificado de forma correcta cada regla. Esto será el fitness de la regla. Si ahora sumamos los resultados de todas las reglas, esto nos da el fitness de la población.

Para realizar el cruce de dos individuos se seleccionan dos nuevos padres mediante el método de selección binaria por torneo con reemplazamiento. Para ello se utilizará el fitness y la probabilidad de cruce. Se procede al cruce que genera dos nuevos hijos, a los cuales se les aplica una mutación en base a la probabilidad de mutación especificada.

Las operaciones de cruce y mutación se repiten hasta conseguir reemplazar el número de reglas deseadas. Finalmente del conjunto de individuos se seleccionan aquellos con peor fitness para ser eliminados.

El proceso se repite con cada nueva población de reglas mientras no se cumpla la condición de parada especificada.

1.4.3.2. Enfoque Pittsburg

En el enfoque Pittsburg la base de reglas se representa como una única cadena la cual es considerada como un individuo de la población. Cada subcadena de longitud n es una regla del sistema de clasificación n-dimensional. En este caso y usando cinco atributos por regla y tres etiquetas lingüísticas por atributo como en el caso del enfoque Michigan, un ejemplo sería el siguiente:

R_1: X_2 es pequeño y x_4 es mediano entonces clase C_1
con seguridad CF_1,
R_2: X_1 es mediano y x_2 es mediano entonces clase C_2
con seguridad CF_2,
R_3: X_5 es pequeño entonces clase C_3 con seguridad CF_3,
R_4: X_1 es grande y x_5 es grande entonces clase C_4
con seguridad CF_4,

cuya codificación seria:

#1#2#	*22###*	*####1*	*3###3*

de longitud 20.

El primer paso es seleccionar de forma aleatoria los n conjuntos de reglas que constituirán los n individuos de la población. Posteriormente definiremos los operadores de cruce y mutación, estableciendo unas probabilidades de cruce y mutación.

Antes de aplicar las operaciones de cruce calcularemos el fitness de cada subconjunto de reglas codificado, o lo que es lo mismo, de cada individuo. El fitness de un individuo formado por n reglas es la suma del fitness de cada una de ellas aplicando el método de la regla ganadora. De esta forma, tal y como ocurría en el enfoque Michigan, se premia a las reglas que clasifican bien.

Para realizar el cruce de dos individuos se seleccionan dos nuevos padres mediante el método de selección binaria por torneo con reemplazamiento. Para ello se utilizará el fitness y la probabilidad de cruce. Se procede al cruce que genera dos nuevos hijos, a los cuales se les aplica una mutación en base a la probabilidad de mutación especificada.

Las operaciones de cruce y mutación se repiten n-1 veces. Finalmente del conjunto de individuos de la población actual se seleccionan el mejor para ser añadido a la nueva población generada.

El proceso se repite con cada nueva población de conjuntos de reglas mientras no se cumpla la condición de parada especificada.

1.4.3.2.1. Sistema hibrido

La idea de la realización de un sistema hibrido es implementar un algoritmo que tenga las siguientes ventajas obtenidas de la combinación del enfoque Michigan y el enfoque Pittsburg,

a) Generar nuevas reglas a partir de de las mejores reglas de la población.
b) Heredar las mejores reglas de la población actual a la siguiente.
c) Optimizar el conjunto de reglas.

El algoritmo híbrido puede optimizar el conjunto de reglas ya que la forma de trabajo es como en el enfoque Pittsburg.

Primeramente como en cualquier algoritmo genético, se especifica el tamaño de la población, el número de reglas lingüísticas, la probabilidad de

cruce y mutación en la parte Pittsburg, la probabilidad de cruce y mutación en la parte Michigan y la condición de parada.

Se generan de forma aleatoria los conjuntos de reglas lingüísticas. Cada conjunto de reglas lingüísticas será un individuo.

Seguidamente se calcula el fitness de cada uno de los individuos de la población actual y se generan n-1 nuevos individuos o conjuntos de reglas usando selección, cruce y mutación en la población actual. Las operaciones de selección y cruce son las mismas que las utilizadas en el enfoque Pittsburg. La única iteración del enfoque Michigan es aplicada como operación de mutación a cada uno de los individuos o conjuntos de reglas generados en la selección y cruce de la parte del enfoque Pittsburg.

Añadimos el mejor individuo de la población actual a la siguiente generada y finalmente, si la condición de parada se cumple, la solución será el mejor conjunto de reglas de la población final obtenida.

1.4.3.3. Aprendizaje iterativo de reglas y Boosting

Las técnicas de boosting borroso [OS06, SO07, SOV06] son reglas repetitivas de métodos de aprendizaje que de forma incremental obtienen bases de conocimiento borrosas a partir de los datos. En cada paso, se añade una regla a la base de conocimiento y se calcula un peso para la misma, en función de los ejemplos de la base de datos que han sido mal clasificados. Estos pesos dependen de las funciones de pertenencias asociadas a las variables lingüísticas, por lo tanto cualquier cambio en estas funciones implicaría volver a calcular los pesos de las reglas. El primer algoritmo de boosting borroso se originó de AdaBoost [JHJS04, JS00], pero también tenemos LogitBoost [FHT98].

Existe la polémica sobre los beneficios que implica tener las reglas con pesos frente a cambiar las funciones de pertenencia, y además se puede decir que la semántica de las variables lingüísticas no se deber alterar en clasificadores lingüísticos comprensibles [SO03] claro, que cuando dicha alteración se permite, hay problemas. En este último caso, boosting borroso podría también beneficiarse de una selección adecuada de las particiones borrosas de las variables de entrada.

Por otra parte existen muchas similitudes entre boosting borroso y el aprendizaje de iterativo de reglas (AIR) [CHHM01], pero hay una serie de diferencias fundamentales. En particular AIR realiza un modelado o cambio

de las funciones de pertenencia, en cambio boosting no altera la forma o posición de las mismas, es decir, no es fácil integrar el modelado o cambio de las funciones de pertenencia con el algoritmo de aprendizaje. Inversamente, las bases de conocimiento aprendidas u originadas mediante boosting contienen reglas borrosas con pesos asociados.

CAPITULO 2

Diseño de particiones lingüísticas para el aprendizaje de clasificadores borrosos basados en reglas

2.1. Introducción

A lo largo del Capítulo 1 se ha realizado un estudio de las características generales de un clasificador descriptivo basado en reglas borrosas, así como un estudio de los distintos algoritmos de clasificación existentes. Para llevar a cabo un proceso de clasificación, uno de los elementos necesarios es la definición de la base de conocimiento, esto es, las particiones que se establecen para cada uno de los atributos del conjunto de ejemplos del clasificador.

En este Capítulo de pretende realizar un recorrido sobre diferentes técnica relacionadas con el diseño de particiones. Se estudiarán tanto las técnicas que adaptan las particiones lingüísticas, como aquellas que no las adaptan. Nuestra técnica está englobada en el segundo grupo y su estudio detallado se proporcionará en el Capítulo 3.

Existen muchas técnicas para diseñar funciones de pertenencia, que podrían ser usadas para resolver el problema de seleccionar la mejor partición borrosa [CHHM01], a la hora de decidir cuál será aquella que mejor se adapte a los datos. Nosotros sugeriremos que la decisión sea tomada en base a una medida de la perdida de información que se produce entre las variables de entrada y salida en el caso de que los datos de entrada sean discretizados. Esta medida es lo que se llama *Información Mutua (IM)* entre la variable de salida y las variables de entrada y como se ha dicho se verá detalladamente en el capítulo 3 de esta tesis.

Para trabajar con clasificación de datos continuos e incluso valores continuos y discretos en necesario que el dominio de cada atributo continuo se discretice en un número finito de intervalos fraccionados [CWC95, DKS95, FI93, LWW04, WC87].

Por otra parte, las funciones de pertenencia de los conjuntos borrosos pueden afectar de forma muy importante en la construcción de modelos borrosos, por lo que la determinación de la función de pertenencia es muy importante. Así, métodos típicos para determinar particiones borrosas incluyen el clustering borroso [FJ00, JF99], redes neuronales [Jan93] y algoritmos genéticos [Kar91].

- Los algoritmos de clustering borroso encuentran particiones (soft) de los datos basados en ciertos criterios. Un dato de un conjunto de datos puede pertenecen a varias particiones (clusters) [FJ00, JF99].

- Se pueden emplear también algoritmos de entrenamiento de redes neuronales no supervisadas para particiones borrosas. Después del entrenamiento, un SOM muestra un conjunto de observaciones a tener en cuenta [Jan93].

- Con los algoritmos genéticos, para formar una partición borrosa, los parámetros de un conjunto borroso se codifican en un cromosoma. Cada cromosoma representa una partición borrosa y esta se evalúa con una función de fitness apropiada [Kar91].

2.2. Esquema de estudio

A continuación se muestra un esquema de los distintos métodos que se van a estudiar en secciones sucesivas y que van a servir como referencia para nuestro estudio.

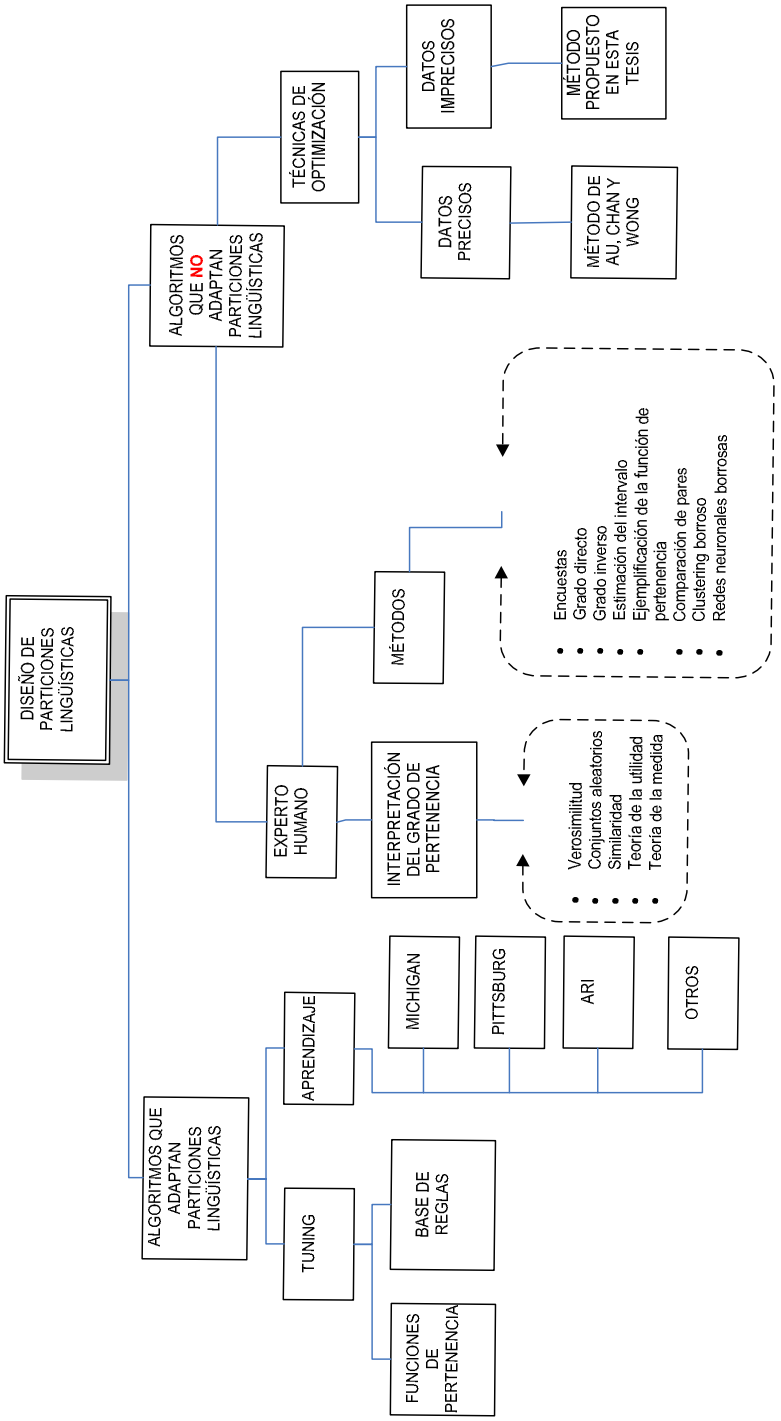


Fig 2.1. Métodos relacionados con el diseño de particiones lingüísticas

2.3. Algoritmos que adaptan las particiones lingüísticas

Este tipo de algoritmos se basan en programación genética evolutiva y el objetivo es encontrar una base de conocimiento tal que el sistema resultante resuelva el problema que se plantea.

El primer problema que se presenta es decidir que partes de la base de conocimiento deberán ser optimizadas por el algoritmo genético.

Si recordamos, la base de conocimiento esta constituida por la base de datos donde se encuentran representadas las funciones de pertenencia del conjunto borroso asociadas a las etiquetas lingüísticas y la base de reglas borrosas.

Este tipo de algoritmos se clasifican en aquellos que realizan un proceso de *tuning* y aquellos que realizan un proceso de aprendizaje. En ambos casos adaptan el contenido de la base de datos, de la base de reglas o de ambos a la vez.

Es difícil realizar una clara distinción entre el proceso de *tuning* y el proceso de aprendizaje [CHHM01], pero se podría hacer la siguiente diferenciación. El t*uning* esta mas relacionado con la optimización de un sistema basado en reglas borrosas ya existente, mientras que el aprendizaje constituye un método de diseño automatizado para los sistemas basados en reglas borrosos a partir de cero.

2.3.1. Tuning

El objetivo del proceso de *tuning* es adaptar un conjunto de reglas borrosas de tal formas que el sistema basado en reglas borrosas obtenga unos resultados mejores de los que ya se obtenían.

Dado que la base de conocimiento esta formada por la base de reglas y la base de datos, y de lo que se trata es de optimizar la base de conocimiento, podremos actuar sobre cualquiera de las dos.

Por otra parte hay que tener en cuenta que un pequeño cambio en las funciones de pertenencia pueden causar un cambio en el comportamiento del sistema lo que conlleva un cambio de la base de reglas.

Dentro del tuning y según se dice en [CHHM01] se distingue:

- El *tuning de las funciones de pertenencia* en lo que se refiere a las etiquetas lingüísticas de las reglas borrosas. Se supone que hay una base de reglas predefinida que representa el conocimiento estructural sobre el sistema a modelar expresado en términos lingüísticos. El funcionamiento del sistema se puede llevar a cabo modelando los conjuntos borrosos y sin modificar la base de reglas.

- El *tuning de la base de reglas* adapta las etiquetas borrosas de la parte del consecuente [PM79]. La idea es identificar reglas en la base de reglas con peor funcionamiento y reemplazarlas por reglas mejores. Reemplazar el consecuente de una única regla tiene un efecto local y el comportamiento del sistema solo afecta al subespacio borroso definido por el antecedente de la regla. Como se dice en Bonissone, Khedkar y Chen [BKC96], el tunning de reglas individuales tiene un efecto pequeño comparado con el gran efecto causado por el *tuning* de las funciones de pertenencia.

2.3.2. Aprendizaje

Una de las principales características del proceso de aprendizaje es la generación de sistemas con la capacidad para cambiar la estructura subyacente, con el objetivo de mejorar su funcionamiento o la calidad de su conocimiento según algunos criterios [CHHM01].

Existen tres principales enfoques relacionados con el aprendizaje genético:

- El *enfoque Michigan*, cuya idea es que cada individuo sea una única regla y la población el conjunto de todas las reglas [HR78] y [Boo82]. Mediante este enfoque nos acercamos a la idea de representar el conocimiento como una única entidad que aprende con la interacción del entorno. Se produce un proceso de adaptación.

- El *enfoque Pittsburg*, cuya idea consiste en representar el conjunto de reglas como un único individuo [Hol75]. Como consecuencia de esto tenemos una población de conjuntos de reglas. Mediante este enfoque la competición dentro del algoritmo genético es entre individuos que se adaptan al entorno.

- El *aprendizaje iterativo de reglas* ha sido creado para combinar las mejores características de ambos enfoques y para resolver los problemas relacionados tanto con el enfoque Michigan como con el enfoque Pittsburg. Fue propuesto por primera vez en SIA [Ven93] y desarrollado ampliamente en el campo de los sistemas genéticos basados en reglas borrosas [CJHL99], [CHGP98] y [GH97].
  Cada individuo de la población representa una única regla, como ocurría en el enfoque Michigan, y solamente el mejor individuo se considera para formar parte de la solución final.
  El algoritmo genético proporciona una solución parcial al problema de aprendizaje y contrariamente a ambos enfoques, el proceso se repite hasta conseguir el conjunto completo de reglas. Esto reduce el espacio de búsqueda, ya que en cada paso sólo se busca la mejor regla.

En [CHHM01] se mencionan también otras tres técnicas utilizadas:

- *COGIN (COverage-Based Genetic INduction)*. Fue propuesto por Greene y Smith [GSm92], [GSm93] y [GSm94] y consiste en un sistema inductivo basado en algoritmos genéticos que explota las convenciones de la inducción de ejemplos para proporcionar este marco de trabajo.

- *REGAL*, propuesto por Giordana y Saitta [GS93], Neri y Giordana [NG95] y Giordana y Neri [GN96], se centra en problemas de aprendizaje de conceptos multi-modales de lógica de primer orden.

- *DOGMA (Domain Oriented Gentic Machine)*, propuesto por Hekanaho [Hek96] y [Hek97], es un algoritmo de aprendizaje basado en genéticos que soporta tanto características de Michigan como de REGAL.

- Medida de Información Mutua propuesta por Sánchez [SSC05] y otros aplicada a la optimización de particiones lingüísticas en clasificadores basados en reglas borrosas.

2.4. Algoritmos que no adaptan las particiones lingüísticas

Dentro del campo del diseño de particiones lingüísticas no hay que olvidar aquellos métodos que no adaptan las particiones lingüísticas, entre otras cosas porque es aquí donde encaja nuestra aportación.

Lo normal en este tipo de algoritmos es que el experto nos de una situación inicial, es decir, a partir de unas observaciones, determinar las particiones borrosas de los datos. Luego se puede utilizar esa definición de las particiones borrosas dada por el experto o bien se pueden aplicar técnicas de optimización para determinar cual es la partición borrosa que mas se ajusta para el problema que se trata.

2.4.1. El papel del experto

Desde que Zadeh [Zad75] introdujo la noción de los conjuntos borrosos, una de las principales dificultades esta relacionada con el significado y la medida del grado de pertenencia de las funciones de pertenencia y esto ha llevado en algunos casos a confusión.

Hay varias interpretaciones en cuanto a de donde puede provenir la borrosidad. Dependiendo de la interpretación de la borrosidad, el significado de la función de pertenencia cambia.

2.4.1.1. Interpretación del grado de pertenencia

Cuando nos introducimos en la teoría de conjuntos borrosos, el concepto de grado de pertenencia parece bastante intuitivo ya que no es mas que una extensión del concepto que todos conocemos como pertenencia a un conjunto. No obstante cualquiera que utilice conjuntos borrosos deberá preguntarse lo siguiente:

- Qué significa grado de pertenencia?
- Cómo se mide?
- Qué operaciones son significativas para trabajar con él?

Existen dos tendencias a la hora de interpretar la borrosidad: aquellas que se basan en decir que la borrosidad es subjetiva en oposición a la palabra objetiva y aquellas que se basan en decir que la borrosidad depende de un individuo en oposición a un grupo de ellos.

Seguidamente vamos a ver algunas de las interpretaciones, que algunos autores, han dado al concepto de grado de pertenencia asociado con la función de pertenencia.

2.4.1.1.1. Punto de vista de la verosimilitud

Este punto de vista sugiere que hay mas de un evaluador o bien que los experimentos se repiten. El modelo TEE (Threshold, Error, assumptions of Equivalente) de Hisdal [His85, His88] para funciones de pertenencia considera que el origen de la borrosidad puede ser debido a:

- Errores en la medida
- Información incompleta
- Contradicciones interpersonales

Según Hisdal, si se tiene la información completa para determinar algo, no hay borrosidad y se puede etiquetar ese algo. Por otra parte dice también, que cuando se pregunta por el grado de pertenencia de algo a un conjunto borroso, la persona, que probablemente tiene una medida imprecisa, construye una curva de error alrededor de la medida. Es decir no tenemos la seguridad al 100% sino de un porcentaje inferior a 100. Son *errores en la medida*.

Supongamos que tenemos una medida imprecisa de la temperatura que oscila en 20 más menos 2 grados Centígrados. Las sucesivas mediciones construyen una curva de error alrededor de 20 grados Centígrados. Si el 90% de las temperaturas esta dentro del intervalo definido, podremos decir que la probabilidad de la temperatura de esa habitación es 0.9 (a la que se le podría asignar la etiqueta lingüística “calido”):

$$\mu_{calido}(20) = P(calido \mid x = 20) = 0.9$$

Es decir, el 90% de la población opina que la temperatura de 20 grados Centígrados es “calida”.

2.4.1.1.2. Punto de vista de conjuntos aleatorios

Este punto de vista utiliza el concepto de α-cortes (ver apéndice A.1), cada uno de los cuales se compone de aquellos elementos cuyo grado de pertenencia es mayor o igual que un umbral dado α. En este caso el conjunto borroso F se representa en términos de α-cortes tal que:

$$\{F_\alpha : \alpha \in (0,1]\}$$

donde:

$$F_\alpha = \{x : \mu_F(x) >= \alpha\}$$

En este tipo de representación la función de pertenencia es una familia de α-cortes y cada α-cortes es un conjunto en el sentido clásico. La función de pertenencia, puede verse como un conjunto aleatorio uniformemente distribuido.

Supongamos que tenemos la siguiente partición, donde la media viene representada por el vértice y el resto de los datos están distribuidos alrededor de la media.

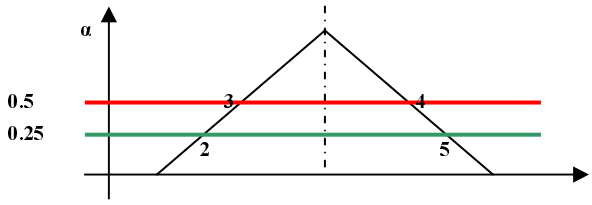


Fig 2.2. Ejemplo de punto de vista de conjuntos aleatorios

El nivel α=0.5 nos dice que la seguridad de que la partición este comprendida en el intervalos [3,4] es de un 50%, en cambio el nivel α=0.25 nos dice que la seguridad de que la partición este comprendida en el intervalo [2,5] es de un 75% ya que al ser un intervalo mayor tenemos mas posibilidad de que el intervalo sea el correcto. En definitiva, a niveles de alfa pequeños, mayor seguridad con respecto al intervalo que representa dicho nivel alfa.

Siguiendo con el ejemplo de las temperaturas del apartado 2.4.1.1.1, en este caso la partición resultante se interpretaría como que tengo una seguridad de un 90% de que la temperatura de 20 grados Centígrados este en el intervalo “cálido”.

2.4.1.1.3. Punto de vista de la similaridad

El punto de vista de similaridad o semejanza de las funciones de pertenencia se basa en la existencia de un ejemplo representativo del conjunto para el cual se quiere diseñar la función de pertenencia [RM75, Lak87]. Se asume que dicho ejemplo "perfecto" del conjunto (o de la categoría) tiene un nivel de pertenencia que será el máximo. El grado de pertenencia de otros ejemplos al conjunto vendrá dado por su distancia relativa al ejemplo perfecto.

En este caso la función de pertenencia podría ser de la forma [Zys81]:

$$\mu(x) = \frac{1}{1 + f(d_x)}$$

donde d_x es la distancia de un ejemplo cualquiera al ejemplo "perfecto". Este punto de vista está relacionado con la Teoría de la Medida y las reglas desde este punto de vista se pueden representar como

SI (x_1 es p_1^i **y** ... **y** x_n es p_n^i) **ENTONCES** (y ES z_i)

donde los conjuntos borrosos se sustituyen por valores de referencia p_j y la función de pertenencia es la distancia entre x_j y p_j.

Si tomamos como ejemplo el del apartado 2.4.1.1.1, relacionado con las temperaturas, diremos que la temperatura de 20 grados esta a una distancia de 0.1 del valor real que representa el concepto "calido".

2.4.1.1.4. Punto de vista de la teoría de la utilidad

Según Giles [Gil88], un conjunto es equivalente a una propiedad o característica y argumenta que se le puede dar un significado a la función de pertenencia si se considera dicha característica junto con el problema del razonamiento borroso. Por lo tanto, está utilizando el camino de la lógica para aproximarse, con una teoría semántica, al concepto del grado de valor de verdad.

Giles define una sentencia borrosa como "una sentencia a la cual atribuimos un grado de certeza" y dependerá del problema a tratar. Por ejemplo, cuando se afirma que "Pedro es alto", Giles asume que existe una

función relacionada con esta aserción. Esta función es mas eficaz cuanto mas cercana a la verdad esté la sentencia.

2.4.1.1.5. Punto de vista de la teoría de la medida

Mediante el punto de vista de la medida, los elementos de un conjunto pueden ser comparados unos con oíros. La relación $A(x)>A(y)$ nos dice que x es mas representativo que y en el conjunto A. Se pueden utilizar, por tanto, valores relativos para obtener información acerca del grado de pertenencia relativo de A.

Utilizamos la teoría de la medida para contestar formalmente a las preguntas generales que se han formulado en el apartado 2.4.1.1.

- ***Medida de Saaty en una escala de cociente***. Saaty [Saa86] piensa que no tiene sentido intentar determinar la función de pertenencia con el escalamiento directo y propone una escala derivada. Supone un conjunto finito ***A*** con ***n*** elementos que llama alternativas, y supone además un conjunto de criterios que llama ***C***. Define una relación binaria en ***A*** y asume que dicha relación es completa y existe una representación. De esta forma va agregando axiomas a la escala fundamental.

- ***Medida de pertenencia***. Existen dos importantes y diferentes problemas de medida en la teoría de conjuntos borrosos. El primer tipo se ocupa de medir el grado de pertenencia de un elemento a un único conjunto borroso. Este problema ha sido estudiado por [Yag79], [NT82b], [NT84], [Tür91], [BWY93] y [Bil95] entre otros. En este problema nos encontramos con un conjunto borroso representado por una etiqueta lingüísticas y un conjunto de elementos. La pregunta que se formula es "¿Con que grado un elemento pertenece a un conjunto borroso?". La idea es realizar comparaciones entre elementos sobre el mismo conjunto borroso en cuanto al grado pertenencia se refiere. Se supone que hay elementos con pertenencia absoluta y elementos con pertenencia nula. Si todo va bien se puede medir la pertenencia mediante una escala ordinaria. Por otra parte [NT82b] argumentan, mediante otra visión distinta del problema que la función de pertenencia se debe medir mediante una escala de intervalos.

- ***Ranking de propiedades***. Un problema complementario es considerar muchas etiquetas lingüísticas para un solo elemento. Este

enfoque esta de acuerdo con algunas teorías lingüísticas donde se demanda que, aunque los adjetivos lingüísticos precedan a sus formas comparativas, estas últimas preceden a la forma simple [Kam75].

- ***Medida de la función de pertenencia y ranking de propiedades***. Es casi imposible tratar de forma separada la medida de las funciones de pertenencia y la selección de conectivas utilizadas en combinación con las funciones de pertenencias. Por otra parte, las escalas resultantes de cada una de las dos técnicas no tienen porque medir la misma entidad, el grado con el que un elemento pertenece a un conjunto borroso no tiene porque ser el mismo con el que un conjunto borroso es asociado a dicho elemento. Por ello surge la idea de combinar los problemas de medida de funciones de partencia y de ranking de propiedades elaborando una nueva estructura donde la escala de medida resultante mide necesariamente el grado de pertenencia en un conjunto borroso [BWY93, Bil95].

- ***Medida conjunta.*** Un término borroso, puede estar compuesto por uno o más términos simples. Por ejemplo el término "confortable" puede estar compuesto por el término "temperatura" y "espacio". Türksen [Tür91] propone técnicas de medida conjunta de tal forma que la escala de medida para el término compuesto, se construye a partir de las escalas de medida de los términos simples utilizando normas y conormas triangulares. Asegura además que se preserva la monotonicidad de la nueva escala construida.

2.4.1.2. Métodos para obtener funciones de pertenencia

Relacionado con los experimentos llevados a cabo para obtener la función de pertenencia, se observan dos tendencias principales: aquella que intenta validar la teoría de conjuntos borrosos y aquella que adopta cierta interpretación del concepto del grado pertenencia y en consecuencia busca una forma de implementación. Puesto que en ambos casos necesitamos métodos para obtener funciones de pertenencia, mostraremos un resumen de algunos de los métodos encontrados en la literatura [NT82a], [CS87] y [Tür91].

2.4.1.2.1. Encuestas

Este método se basa en el punto de vista de que la borrosidad viene dada por los desacuerdos interpersonales. Se toma un conjunto de individuos y se les pregunta, "¿Estás de acuerdo en que un elemento ***a*** es un concepto ***F***?". La respuesta puede ser "si" o "no". Se tratan las respuestas, realizando un promedio y se construye la función de pertenencia. Hersh y Carmazza [HC76] utilizan este enfoque en sus experimentaciones.

El método de encuestas es una de las maneras naturales de obtener las funciones de pertenencia aplicables a la interpretación de la borrosidad basada en el punto de vista de la interpretación de la probabilidad.

2.4.1.2.2. Grado directo o pertenencia directa

Es uno de los más directos para obtener la función de pertenencia. Se basa en el hecho de que la borrosidad viene dada por la imprecisión subjetiva individual. Al individuo se le necesita para clasificar un elemento ***a*** con respecto a un conjunto ***F*** repetidamente un y otra vez. Hersh y Carmazza [HC76] también utilizan este enfoque en sus experimentaciones.

Los métodos de grado directo requieren evaluaciones para ser medidos, al menos, en escalas del intervalo [Tür88] y [Tür91].

2.4.1.2.3. Grado inverso

En este método, a un concepto se le asigna un grado de pertenencia y se le pregunta a un individuo si es capaz de identificar si el objeto pertenece al conjunto que se corresponde con ese grado de pertenencia [Tür88] y [Tür91].

Una vez obtenidas las respuestas, se pueden tomar las distribuciones condicionales para la distribución normal y la media y la varianza se puede estimar como de costumbre. Este método también requiere evaluaciones para ser medidos, al menos, en escalas del intervalo.

2.4.1.2.4. Estimación del intervalo

La estimación del intervalo se basa en la interpretación del conjunto aleatorio para determinar la función de pertenencia. Se le preguntan a varios

individuos para que proporcione un intervalo que describa al concepto ***F*** dado un objeto ***a***.

Supongamos que tenemos que representar valores, resultado de una encuesta que se realiza a tres personas. Cada una de ellas proporciona unos valores que no tienen porque ser los mismo pero que tienen la característica de que la media es la misma. Si podemos determinar para cada uno de ellos y dentro del intervalo, el subintervalo que contiene el 100% de los datos, resulta que puede ocurrir que el uno de ellos sea menor que el otro. Si los "apilo", obtengo como resultado una función de pertenencia, para la cual conservo la media, a partir de los intervalos proporcionados por los individuos encuestados.

2.4.1.2.5. Ejemplificación de la función de pertenencia

La idea de este método es obtener resultados mediante ejemplos. Dado un conjunto de objetos ordenados la idea es que un grupo de individuos sea capaz de etiquetar los objetos con conceptos concretos. Los resultados están en desacuerdo con el método del grado directo y con el método de las encuestas, muy probablemente porque no hay repetición para normalizar los efectos del error o del "ruido" [KB74b].

2.4.1.2.6. Comparación de pares

Kochen y Prade [KB74a] aportan evidencias experimentales relacionadas con la precisión de "mayor", de "mucho más mayor" y de "mucho mayor". En relación a cómo se añada "muy" hace que se destaque el adjetivo de forma mas precisa.

2.4.1.2.7. Clustering borroso

Este método construye la función de pertenencia a partir de un conjunto de datos. Se recoge una cantidad adecuada de datos, se requiere un modelo basado en conjuntos borrosos para analizar el problema y las funciones de pertenencia se construyen a partir de los datos disponibles.

Normalmente el procedimiento es el siguiente [NTS93]:

- Agrupar los datos de salida y entonces proyectarlo en los datos de entrada, generar agrupamientos y seleccionar las variables asociadas

con relaciones de la entrada-salida. Este método determina agrupaciones en el espacio de los datos basándose en la norma Euclidiana.

- Formar las funciones de pertenencia para las variables seleccionadas.

- Seleccionar las variables de entrada dividiendo los datos en tres subgrupos, dos para el aprendizaje y el tercero para las comprobaciones.

2.4.1.2.8. Técnicas de redes neuronales borrosas

Con el avance en modelos de redes neuronales y sus usos en aprendizaje, tiene un interés considerable la unificación de las redes neuronales y de la teoría de conjuntos borrosas [Roc82], [Tak90] y [Kos91].

Un ejemplo de utilización de estos métodos es el estudio que hace Takagi y Hayashi [TH91] sobre una red neuronal que genera funciones de pertenencia no lineales y mutidimensionales. Este tipo de funciones de pertenencia reducen el número de reglas borrosas en la base de reglas.

2.4.1.3. Relación entre interpretación del grado de pertenencia y las técnicas para obtener funciones de pertenencia

A continuación, en la figura 2.3. se muestra la relación existente entre la interpretación del grado de pertenencia y las técnicas para obtener funciones de pertenencia.

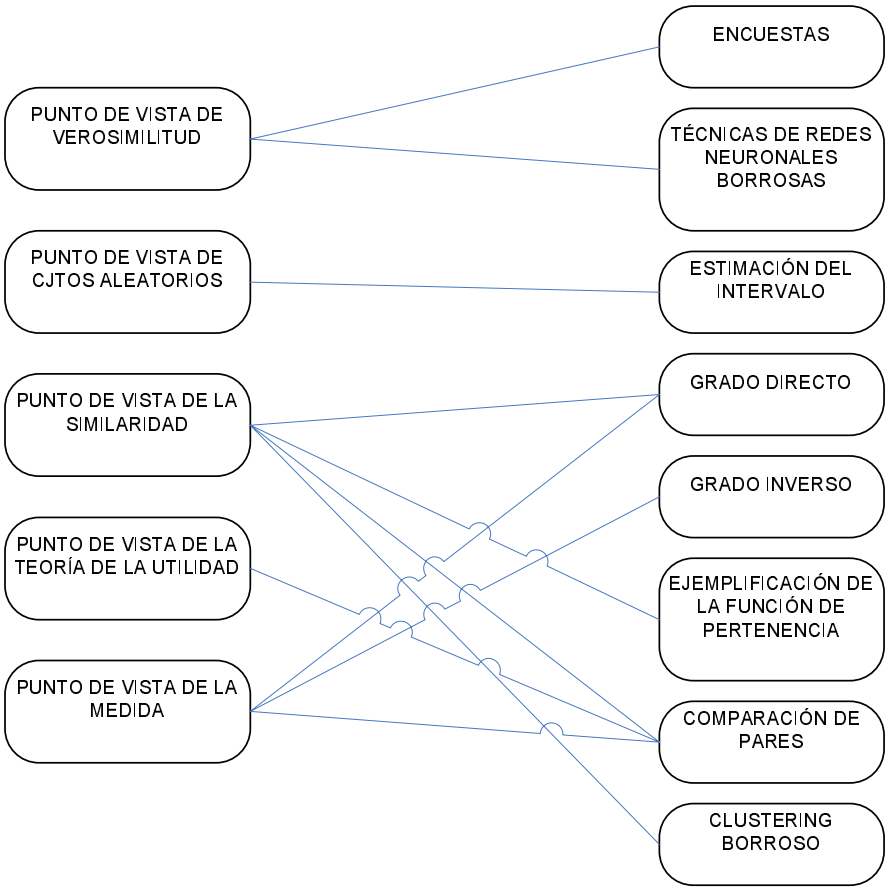


Fig 2.3. Técnicas de optimización

2.4.2. Técnicas de optimización

Con el objetivo de compararnos con otros métodos existentes, se estudian posibles trabajos que traten el problema de la optimización de particiones, ya que vamos a aplicar lanuela definición propuesta al problema mencionado. Las técnicas de optimización se dividen en aquellas orientadas al algoritmo y aquellas que se centran en otro tipo de optimizaciones independientemente del algoritmo que se utilice, esto es, no importa la eficacia, sino la eficiencia. En este último grupo, es donde encajaría el método propuesto en esta tesis, que se explicará con más detalle en el capítulo siguiente.

Por otra parte hay que tener en cuenta que los datos utilizados pueden ser de dos tipos: datos precisos y datos impresitos. Cuando se trata de optimizar particiones con datos precisos se encuentran algunos métodos que a partir de los valores de los atributos de los ejemplos generan un conjunto de particiones posibles para ese atributo. Nuestro método, por otra parte, optimiza tanto particiones con datos precisos como con datos imprecisos y proporciona la mejor o mejores particiones que optimizan la partición de partida, es nuestro caso, uniforme.

2.4.2.1. Método propuesto por Au, Chan y Wong

Dentro de las técnicas de optimización orientadas al algoritmo esta la propuesta por [ACW06] donde se plantea la utilización de una nueva medida que llaman *medida redundante de la interdependencia entre la clase y un atributo* y que se aplicará en combinación con la definición de información mutua a la generación de forma automática de particiones borrosas. Está medida se aplica a funciones de pertenencia con forma trapezoidal ya que según se dice en [ACW06] las triangulares son un caso particular de estas últimas.

Partimos, como en cualquier problema real de un conjunto de atributos $A_1, \cdots, A_k$ y las clases existentes $A_c, c \in \{1, \cdots, k\}$.

Para cada atributo continuo, $A_k, k \in \{1, \cdots, k\} - \{C\}$, el dominio se puede representar mediante conjuntos borrosos, $S_{k1}, \cdots S_{kj_k}$, tal que μ_{Sk_j} es la función de pertenencia del conjunto borroso $S_{kj}, j = 1, \cdots, J_k$.

Se define la medida de la redundancia de la interdependencia entre el atributo clase y el conjunto borroso como:

$$R(A_c; S_{kj}) = \frac{I(A_c; S_{kj})}{H(A_c; S_{kj})}$$

donde $I(A_c; S_{kj})$ es la información mutua entre la clase y el conjunto borroso que viene dada por:

$$I(A_c; S_{kj}) = \sum_{r=1}^{R_c} ps_{kj} a_{cr} \log \frac{ps_{kj} a_{cr}}{pa_{cr} ps_{kj}}$$

y $H(A_c; S_{kj})$ es la entropía conjunta de la clase y el conjunto borroso calculado como:

$$H(A_c; S_{kj}) = -\sum_{r=1}^{R_c} ps_{kj} a_{cr} \log ps_{kj} a_{cr}$$

Como resultado, el particionamiento se puede considerar como el proceso de eliminar la redundancia introducida por demasiados valores del atributo. Al mismo tiempo el proceso de particionamiento borroso deberá minimizar la perdida de correlación entre el atributo clase y otro atributo.

El problema del particionamiento se puede resolver encontrando la partición borrosa en el dominio del atributo que maximice la medida redundante de la interdependencia.

En definitiva, se trata de encontrar aquella partición del conjunto de particiones posibles tal que la medida de la redundancia de la interdependencia entre la clase y el atributo sea máxima.

Para realizar esta tarea, el método que se presenta Au W.-H. y otros en [ACW06] genera particiones borrosas a partir de los datos para maximizar la interdependencia entre la clase y el atributo.

Para encontrar la partición borrosa, que maximice la medida dada en el apartado anterior, proponen utilizar programación fraccional [Sni92] y basándonos en el algoritmo de Dinkelbach [Din67], hace uso de la medida proporcionada y propone un nuevo algoritmo de particionamiento borroso, para particionar atributos continuos. Tiene dos componentes importantes:

uno es un proceso iterativo que utiliza el primer componente para conducir hacia la solución óptima global final, y la otra intenta obtener el valor máximo de la función objetivo aplicando programación dinámica para optimizar la interdependencia entre la clase y el atributo.

Nosotros hemos llevado a cabo la implementación del algoritmo pero tras pruebas realizadas y siguiendo la explicación propuesta por los autores, no produce los resultados esperados y no se ha podido realizar un análisis comparativo de los resultados. En los trabajos encontrados no se especifica claramente los objetivos y parámetros necesarios, así que, aunque se ha estudiado, se ha desestimado en la experimentación.

CAPITULO 3

Estimación de la Información mutua en problemas con datos imprecisos

3.1. Introducción

En el capítulo 1 se ha realizado un estudio de los distintos sistemas de clasificación basados en reglas borrosas. En el capitulo 2 se ha realizo un estudio de las distintas formas de diseño de particiones, desde el punto de vista del experto, pasando por la utilización de algoritmos específicos. Algunas de las técnicas parten de una partición inicial para obtener la partición optimizada y otras proponen una partición a partir de los datos.

En este capítulo se aportará una nueva definición de la Información Mutua que aplicada al diseño de particiones borrosas y apoyándonos en un algoritmo genético, nos proporcionará como resultado una partición optimizada, que maximice la información acerca de la clase, que demostraremos, nos ofrece unos resultados de clasificación mejores que los que proporcionaría la partición inicial.

En un primer momento se propone una definición de Información Mutua como se muestra en el apéndice B, pero dicha definición tenía sus problemas que se detallan en el apéndice y esto nos lleva a proponer nuevas definiciones que mostraremos en las secciones siguientes.

3.2. ¿Por qué una nueva definición?

El uso de términos lingüísticos para describir valores de una variable es característico de muchas técnicas basadas en la lógica y en la estadística borrosa. El proceso de fuzzificación o discretización permite que una variable numérica pueda ser discretizada y cada uno de sus valores asociado a un conjunto borroso, definido a su vez sobre el conjunto de términos lingüísticos.

En este trabajo utilizaremos particiones de tipo Ruspini para codificar la semántica de un clasificador borroso. En estas particiones, la suma de las pertenencias de cualquier valor crisp a todos los conjuntos que la forman es la unidad. Asimismo, este conjunto de pertenencias tiene, a lo sumo, dos valore son nulos. Esto no es cierto, sin embrago, cuando el valor considerado es intercalar o borroso. La suma de las probabilidades entre un intervalo y los conjuntos que forman una partición de Ruspini, será en general, mayor que la unidad, y si es posible que haya mas de dos particiones no nulas.

Por ejemplo, para una partición borrosa del universo de las temperaturas, con términos 'FRIO', 'TEMPLADO' y 'CALIENTE', un valor de 45 grados se podría representar mediante el conjunto borroso

{0.0/FRIO+0.2/TEMPLADO+0.8/CALIENTE}

y el valor 45±1 grados se puede representar mediante el conjunto borroso que sigue (ver figura 3.1.),

{0.0/FRIO+0.3/TEMPLADO+0.9/CALIENTE}

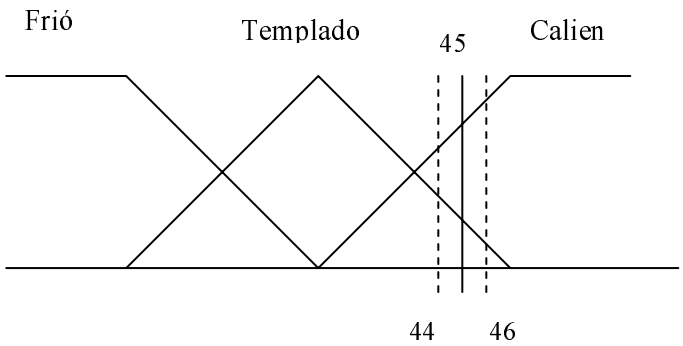


Fig 3.1. Ejemplo de partición borrosa

Más aún, con esta representación basada en pertenencias a conjuntos borrosos, también podemos manejar información incompleta o imprecisa, que no puede asociarse a ningún valor numérico de la variable. Por ejemplo, un conjunto como

{0.5/FRIO+0.5/TEMPLADO+ 0.5/CALIENTE}

donde al valor del atributo del ejemplo se le asocia una pertenencia de 0.5 a cada una de las etiquetas lingüísticas del conjunto borroso, no se corresponde con ningún valor crisp del universo de entrada, pero es un dato de entrada válido para un clasificador borroso.

Los clasificadores borrosos basados en reglas utilizan datos codificados de la forma anterior para incluir la clase a la que pertenece el objeto del que se conocen esos datos. De igual manera que, en clasificación estadística, es posible incluir una regla de decisión a partir de un conjunto de datos clasificados, en un clasificador se puede estimar las reglas que lo componen a partir de una muestra de datos con la codificación mencionada, y que, a su vez, son una muestra de una variable aleatoria borrosa. En

clasificación estadística, tiene sentido estimar la información mutua entre la clase del objeto y el valor de las mediciones que se han tomado sobre el mismo, porque pueden contestarse preguntas como 'qué subconjunto de variables es el mas representativo' o, en otros casos, 'qué discretización de los datos de entrada es la más adecuada'. En un clasificador borroso tiene sentido hacerse las mismas preguntas, sin embargo, la respuesta a ambas depende de una estimación de la información mutua entre una variable aleatoria y una variable aleatoria borrosa. Hasta donde nosotros conocemos, dicha estimación aún no ha sido resuelta.

En este capítulo, propondremos nuestra propia definición de la información mutua entre una variable aleatoria clásica y una variable aleatoria borrosa. En esta definición, la información mutua no será un número sino un conjunto borroso. Como se mostrará en la siguiente sección, este resultado es una consecuencia de las definiciones de variable aleatoria borrosa que hemos considerado. Así mismo, y como ejemplo de su aplicación, mostraremos como esta medida de información puede optimizarse mediante un algoritmo genético y usarse para encontrar la partición borrosa que, dado un número de términos lingüísticos, más información aporta acerca de la clase del objeto.

Se mostrará, por una parte, las distintas propuestas para el cálculo de la Información Mutua, aplicada en esta tesis a la optimización de particiones con una futura aplicación a la selección de características lingüísticas, y por otra parte, se proponen las soluciones algorítmicas-genéticas para la resolución deforma optima del problema desde el punto de vista computacional.

Un esquema de estudio es el que se muestra a continuación:

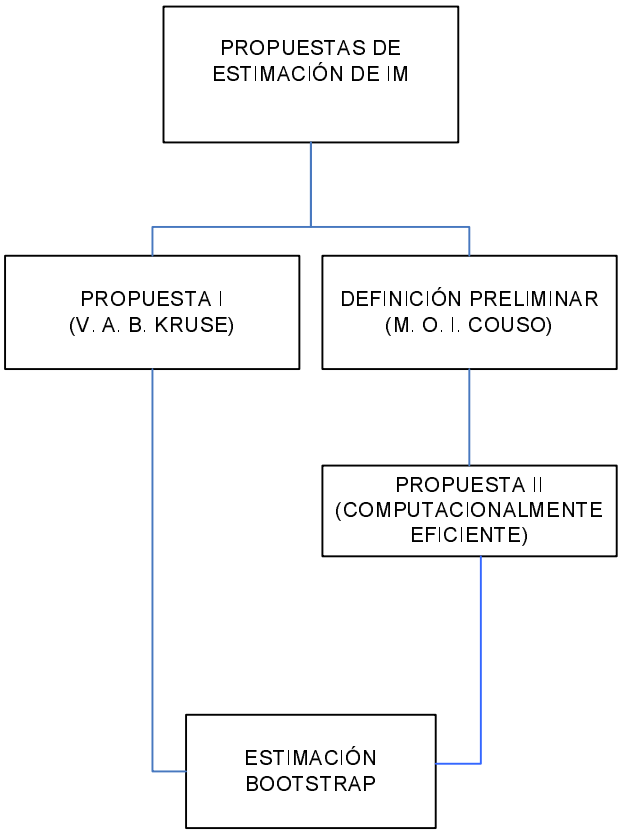


Fig 3.2. Estructura del Capitulo 3

3.3. Propuestas de estimación de la Información Mutua

En esta sección se realizará un estudio detallado de las distintas propuestas aportadas para el cálculo de la Información Mutua. Ambas propuestas se reducen a la definición clásica cuando la variable aleatoria es crisp, pero generalizan de forma distinta esta definición en variables aleatorias borrosas.

En ambos casos, se estudia como construir una estimación computacionalmente eficiciente cuando contamos con grandes datasets con un número elevado de variables. Todo ello se expone en las siguientes subsecciones.

3.3.1. Interpretación I. Información mutua entre una variable aleatoria y una variable aleatoria borrosa.

En la interpretación de variable aleatoria mas extendida en la literatura, se considera que una variable aleatoria borrosa se define mediante sus α-cortes (ver apéndice A.1), cada uno de los cuales es un conjunto aleatorio. Un conjunto aleatorio es una aplicación en la que las imágenes de los resultados del experimento aleatorio son conjuntos clásicos.

Se dice que una variable aleatoria X está contenida en un conjunto aleatorio S, cuando la imagen de cualquier resultado mediante X está contenida en la imagen de ese mismo resultado mediante S. Esto es, para una variable aleatoria $X : \Omega \to \Re$ y un conjunto aleatorio $S : \Omega \to \mathrm{P}(\Re)$, X está contenida en S (y lo denotaremos $X \in C(S)$) cuando

$$X(\omega) \in S(\omega) \text{ para todo } \omega \in \Omega$$

En este sentido, podemos decir que un conjunto aleatorio es un conjunto de variables aleatorias (aquellas contenidas en él). Volviendo a la variable aleatoria borrosa, cada uno de sus α-cortes, que es un conjunto aleatorio, contiene a un conjunto de variable aleatorias. Como estos α-cortes están anidados, las variables contenidas en cualquiera de ellos contenidas también en el conjunto aleatorio de nivel α=0, al que llamaremos soporte de la variable aleatoria borrosa. Asociemos entonces a cada variable aleatoria contenida en este soporte el mayor índice α de todos los α-cortes (conjuntos aleatorios) que la contienen. Este valor fue llamado aceptabilidad de la variable por Kruse [KM87]. Según este autor, una variable aleatoria borrosa es un conjunto borroso definido sobre el espacio de todas las aplicaciones medibles de Ω en $\Re$, donde la pertenencia de cada una de estas variables viene dada por su aceptabilidad.

3.3.1.1. Propuesta I de definición de Información mutua

En esta sección se definirá lo que es la Información Mutua entre una variable aleatoria y uno conjunto aleatorio, así como lo que es la Información Mutua entre una variable aleatoria y una variable aleatoria borrosa. Seguidamente, se explica el procedimiento numérico para estimar la información mutua entre dos variables aleatorias, y se introducirá el concepto de aceptabilidad de una muestra, ilustrando todo ello con un ejemplo.

3.3.1.1.1. Información Mutua entre una variable aleatoria y un conjunto aleatorio

Definimos la Información Mutua entre una variable aleatoria X y un conjunto aleatorio S como el conjunto de valores de información mutua entre la variable X y cada una de las variables contenidas en S, es decir:

$$IM(X,S) = \{IM(X,T) \mid T \in C(S)\}$$

3.3.1.1.2. Información Mutua entre una variable aleatoria y una variable aleatoria borrosa

Definimos la Información Mutua entre una variable aleatoria X y una variable aleatoria borrosa Λ como el conjunto borroso que contiene los valores de información mutua entre la variable X y cada una de las variables cuya aceptabilidad con respecto a Λ es mayor que cero. La pertenencia del valor t al conjunto borroso $IM(X,\Lambda)$ es

$$IM(X,\Lambda)(t) = \max_{T \in C(soporte(\Lambda))} \{acc(T) \mid IM(X,T) = t\}$$

3.3.1.1.3. Estimación a partir de dos muestras

En primer lugar, necesitamos un procedimiento numérico para estimar la información mutua entre dos variables aleatorias. En todos los problemas que estudiaremos, las variables aleatorias son discretas. Si las frecuencia relativas de cada uno de los valores de dos variables X e Y son $p_1, p_2, \ldots, p_n$ y $q_1, q_2, \ldots, q_m$, y las frecuencias relativas de los valores de la variable $X \times Y$ son $r_1, r_2, \ldots, r_{nm}$, entonces la información mutua entre las variables X e Y es

$$IM(X,Y) = -\sum_{i=1}^{n} p_i \log p_i - \sum_{i=1}^{m} q_i \log q_i + \sum_{i=1}^{nm} r_i \log r_i$$

En segundo lugar, introduciremos el concepto de aceptabilidad de una muestra: dada una nuestra $\{\Lambda_1, \Lambda_2, \ldots, \Lambda_N\}$ de una variable aleatoria borrosa, definimos la aceptabilidad de la muestra $\{X_1, X_2, \ldots, X_N\}$ de una

variable aleatoria clásica como el mínimo de las pertenencias de cada valor de X_i a su componente correspondiente Λ_i, es decir:

$$acc(X,\Lambda) = \min_{i=1}^{N} \Lambda_i(X_i)$$

Supongamos ahora que se nos da una muestra de una variable X, $\{X_1, X_2, \ldots, X_n\}$, y una muestra de una variable aleatoria Λ, $\{\Lambda_1, \Lambda_2, \ldots, \Lambda_m\}$. La información mutua entre X y Λ se estima mediante el conjunto borroso $IM(X,\Lambda)$:

$$IM(X,\Lambda)(t) = \max_{Y \in C(soporte(\Lambda))} \{acc(Y,\Lambda) \mid t = IM(X,Y)\}$$

La única dificultad en la implementación de esta fórmula se reduce a generar el conjunto de las muestras Y con aceptabilidad no nula, como se muestra en el ejemplo que sigue,

3.3.1.1.4. Ejemplo

Considérese la siguiente muestra de tamaño tres de las variables Λ y X:

Λ	X
{0.0/FRIO+0.2/TEMPLADO+0.8/CALIENTE}	A
{0.4/FRIO+0.6/TEMPLADO+0.0/CALIENTE}	B
{1.0/FRIO+0.0/TEMPLADO+0.0/CALIENTE}	A

Tabla 3.1. Muestra de un conjunto borroso de tamaño 3 con datos precisos

Se desea estimar la información mutua entre X y Λ. En primer lugar, generaremos el conjunto de muestras contenidas en el soporte de Λ, y calcularemos la aceptabilidad de cada una de ellas, como ya se explico con anterioridad. Llamaremos Y_1, Y_2, Y_3, Y_4 a las cuatro muestras o alternativas posibles.

Y_1	X
TEMPLADO	A
FRIO	B
FRIO	A
Aceptabilidad=min{0.2,0.4,1}=0.2	

Tabla 3.2. Alternativa 1

Y_2	X
TEMPLADO	A
TEMPLADO	B
FRIO	A
Aceptabilidad=min{0.2,0.6,1}=0.2	

Tabla 3.3. Alternativa 2

Y_3	X
CALIENTE	A
FRIO	B
FRIO	A
Aceptabilidad=min{0.8,0.4,1}=0.4	

Tabla 3.4. Alternativa 3

Y_4	X
CALIENTE	A
TEMPLADO	B
FRIO	A
Aceptabilidad=min{0.8,0.6,1}=0.6	

Tabla 3.5. Alternativa 4

A continuación, estimamos la Información Mutua $IM(Y_1, X), IM(Y_2, X), IM(Y_3, X), IM(Y_4, X)$ de cada una de las muestras o alternativas y las relacionamos con las aceptabilidades de las muestras correspondientes:

IM	Aceptabilidad
$IM(Y_1, X) = 0.1744$	0.2
$IM(Y_2, X) = 0.1744$	0.2
$IM(Y_3, X) = 0.1744$	0.4
$IM(Y_4, X) = 0.6365$	0.6

Tabla 3.6. Relación entre IM y Aceptabilidad

Por último, definimos la información mutua entre Λ y X como el conjunto borroso correspondiente a la máxima aceptabilidad asociada a cada posible valor de IM:

$$IM(X, \Lambda) = 0.4/0.1744 + 0.6/0.6365$$

Esta definición aunque sencilla y con buenos resultados a priori, tiene el problema de que no es aplicable a grandes datasets con un número de variables grande, como estudiaremos en la sección 3.3.3.3.

3.3.2. Interpretación II. Información Mutua entre una variable aleatoria y una variable aleatoria borrosa.

La representación borrosa a la que se hizo mención en la sección 3.2. tiene también una interpretación basada en la teoría de probabilidades imprecisas. Siguiendo con el ejemplo anterior, podemos entender que el conjunto borroso

{0.1/FRIO+0.2/TEMPLADO+ 0.9/CALIENTE}

significa que la probabilidad de que el resultado del experimento haya sido 'FRIO' es como máximo de 0.1, la probabilidad de que sea 'TEMPLADO' es como máximo 0.2 y de que sea 'CALIENTE' es como máximo 0.9 (y por lo tanto, la probabilidad de que sea 'CALIENTE' es como mínimo, de 1-0.1-0.2=0.7).

Hay que tener en cuenta que, con esta interpretación se nos presentan , el conjunto {1/FRIO+1/TEMPLADO+1/CALIENTE} representa el desconocimiento absoluto del valor de la entrada (éste podría ser por ejemplo, la codificación de un dato con una coordenada desconocida), mientras que el conjunto {0.5/FRIO+0.5/TEMPLADO+0.5/CALIENTE} mencionado en la introducción, aún sin indicar una mayor preferencia hacia

ninguno de los tres valores lingüísticos, contiene la información de que la probabilidad de cualquiera de los tres resultados nunca es superior a 0.5.

Además, un conjunto como {0.0/FRIO+0.2/TEMPLADO +0.8/CALIENTE} está asociado a una única asignación de probabilidad porque 0.0+0.2+0.8=1 (conjuntos como este típicamente se asocian a una partición de Ruspini [Rus69]) y un conjunto como {0.0/FRIO+ 0.2/TEMPLADO+0.4/CALIENTE} (donde 0.0+0.2+0.4<1) no tiene sentido con esta interpretación, aunque sí lo tendría con otras, basadas en la lógica borrosa.

Nosotros nos basaremos en la interpretación basada en probabilidades imprecisas para proponer una segunda definición de información mutua entre una variable aleatoria y una variable aleatoria borrosa. Esta nueva definición, aunque menos general (dado que los resultados de la variable aleatoria borrosa no pueden ser conjuntos borrosos arbitrarios) nos permitirá, como se discutirá en la sección 3.3.3.3 posteriores, una estimación computacionalmente más eficiente, en la que se evalúe una fracción de las parejas de muestras necesarias en la definición anterior.

3.3.2.1. Una definición preliminar de Información Mutua, basada en probabilidades imprecisas.

Según la definición, una variable aleatoria borrosa Λ puede interpretarse, desde un punto de vista basado en la teoría de la posibilidad, como una distribución de probabilidad p_Λ, conocida sólo en parte, sobre el conjunto de resultados observables del experimento aleatorio.

Por ejemplo, supongamos que el resultado del experimento es un individuo al que llamaremos ω, y que el resultado observado ante la aparición de ese ω podría ser una de los tres términos lingüísticos 'FRIO', 'Ttemplado' o 'CALIENTE'. En este caso, el que la imagen de ω por medio de la variable aleatoria borrosa Λ sea el conjunto borroso $\Lambda(\omega)$ = {0.1/FRIO+0.2/TEMPLADO+0.9/CALIENTE} significa que, cuando el resultado del experimento aleatorio es ω, la probabilidad de que el resultado observado sea 'FRIO', 'TEMPLADO' o 'CALIENTE' está dominada por la posibilidad mencionada, esto es:

$$p_\Lambda(FRIO \mid \omega) \le 0.1$$
$$p_\Lambda(TEMPLADO \mid \omega) \le 0.2$$
$$p_\Lambda(CALIENTE \mid \omega) \le 0.9$$

Obsérvese que de esta información también podemos deducir que

$$p_\Lambda(CALIENTE \mid \omega) \ge 1-(0.1+0.2)$$

y, en ese caso, las probabilidades inferiores de las otras dos etiquetas son cero.

Supongamos ahora que el experimento aleatorio puede producir como resultado sólo son individuos ω_1 y ω_2, con probabilidad $p(\omega_1) = p(\omega_2) = 1/2$. Admitamos también que las imágenes de estos individuos son:

$$\Lambda(\omega_1) = \{0.1/\text{FRIO}+0.2/\text{TEMPLADO}+0.9/\text{CALIENTE}\}$$
$$\Lambda(\omega_2) = \{0.0/\text{FRIO}+0.2/\text{TEMPLADO}+0.8/\text{CALIENTE}\}$$

Entonces estaremos indicando que:

$$p_\Lambda(FRIO \mid \omega_1) \le 0.1$$
$$p_\Lambda(TEMPLADO \mid \omega_1) \le 0.2$$
$$0.7 \le p_\Lambda(CALIENTE \mid \omega_1) \le 0.9$$

y por otra parte:

$$p_\Lambda(FRIO \mid \omega_2) = 0$$
$$p_\Lambda(TEMPLADO \mid \omega_2) = 0.2$$
$$p_\Lambda(CALIENTE \mid \omega_2) = 0.8$$

y aplicando la regla de la probabilidad total deduciremos que la probabilidades de que los resultados 'FRIO', 'TEMPLADO' y 'CALIENTE' sean observadas sean las siguientes:

$$p_{\Lambda}(FRIO) \leq 0.05$$
$$p_{\Lambda}(TEMPLADO) \leq 0.2$$
$$0.75 \leq p_{\Lambda}(CALIENTE) \leq 0.85$$

Es decir, una variable aleatoria borrosa Λ representa unas cotas superior e inferior de las probabilidades de la observación de cada resultado.

Dicho esto, la información mutua entre la variable aleatoria borrosa Λ y una variable aleatoria X se podría definir mediante el conjunto de valores de información mutua entre X y cada una de las distribuciones de probabilidad compatibles con la distribución de posibilidad anterior. Si p_y es la probabilidad inducida por una variable aleatoria Y, este conjunto es

$$IM(X,\Lambda) = \{t \mid t = IM(X,Y) \text{ y la variable Y cumple que}$$
$$p_y(FRIO) \leq 0.05, p_y(TEMPLADO) \leq 0.2,$$
$$0.75 \leq p_y(CALIENTE) \leq 0.85$$
$$p_y(FRIO) + p_y(TEMPLADO) + p_y(CALIENTE) = 1\}$$

Las ventajas teóricas de este enfoque son evidentes: la información mutua entre una variable aleatoria borrosa y una variable aleatoria clásica es un intervalo, y en el caso de que la variable aleatoria borrosa provenga de un interfaz de fuzzificación a través de una partición de Ruspini (donde la suma de las probabilidades superiores de cada etiqueta es 1), el resultado es un número.

Por el contrario, esta definición tiene importantes desventajas prácticas. El problema radica en que la información mutua no solo depende de las probabilidades de cada resultado observable, sino de las probabilidades conjuntas de cada uno de estos resultados y de los resultados de la variable X. En una implementación en un computador de esta medida, sería necesario enumerar todas las variables Y que están dominadas por la posibilidad antes mencionada, y evaluar $IM(X,Y)$ para cada una. Aunque pudiésemos definir una clase de equivalencia de estas variables Y y dividir el problema en partes, la solución no es sencilla.

3.3.2.2. Propuesta II de definición de Información Mutua

Es posible formular, sin embargo, otra definición de variable aleatoria que comporta las ventajas teóricas de la anterior y que a la vez pueda ser programada de forma eficiente. En esta nueva propuesta (a la que llamaremos propuesta II) supondremos que una variable aleatoria borrosa Λ puede interpretarse como una distribución de probabilidad p_Λ, conocida sólo en parte, sobre el conjunto de variables aleatorias contenidas en el soporte Λ. El conocimiento que tenemos acerca de esta distribución de probabilidad se reduce a que, para cada variable Y, disponemos de los valores $p_{*\Lambda}(Y)$ y $p^{*}{}_{\Lambda}(Y)$, tales que $p_{*\Lambda}(Y) \le p_\Lambda(Y) \le p^{*}{}_{\Lambda}(Y)$.

La diferencia entre las interpretaciones I y II está en que, en el primer caso, definíamos una función de aceptabilidad sobre las variables aleatorias contenidas en el soporte de la variable aleatoria borrosa, mientras que ahora estamos definiendo una distribución de probabilidad imprecisa sobre ese mismo conjunto. En la interpretación I, obteníamos que la información mutua era un conjunto borroso, mientras que ahora obtendremos una probabilidad superior y una probabilidad inferior para cada uno de los valores. La esperanza de la información mutua con respecto a esa probabilidad imprecisa será un intervalo, en el caso general, y un número, cuando la variable aleatoria borrosa proceda de una distribución de Ruspini, similares a los vistos en la sección anterior. Como veremos mas adelante, existen algoritmos numéricos computacionalmente eficaces para la estimación de una información mutua definida así.

Siguiendo entonces con esta interpretación, definimos que la probabilidad de que la información mutua entre X y Λ tome el valor *t* es el valor

$$p(IM(X,\Lambda)=t) = \sum_{Y \in C(soporte(\Lambda))} \{p_\Lambda(Y) \mid t = IM(X,Y)\}$$

Como $p_\Lambda(Y)$ no es conocida, tampoco lo es $p(IM(X,\Lambda)=t)$, pero podemos determinar dos cotas $p_*(IM(X,\Lambda)=t)$ y $p^{*}(IM(X,\Lambda)=t)$, definidas de la siguiente manera:

$$p_*(IM(X,\Lambda)=t) = \sum_{Y \in C(soporte(\Lambda))} \{p_{*\Lambda}(Y) \mid t = IM(X,Y)\}$$

$$p^*(IM(X,\Lambda)=t)=\sum_{Y\in C(soporte(\Lambda))}\{p^*{}_\Lambda(Y)\mid t=IM(X,Y)\}$$

Obsérvese que, aunque es cierto que $p_*(IM(X,\Lambda)=t)\le p(IM(X,\Lambda)=t)\le p^*(IM(X,\Lambda)=t)$, no estamos afirmando que este par de cotas sean las más estrictas. En general, existirá un conjunto más pequeño que $[p_*(IM(X,\Lambda)=t),\, p^*(IM(X,\Lambda)=t)]$ y que también contendrá con seguridad el valor $p(IM(X,\Lambda)=t)$.

3.3.2.2.1. Estimación numérica a partir de dos muestras

De forma análoga a la aceptabilidad definida en la propuesta anterior, introduciremos en esta sección el concepto de verosimilitud de una muestra. Utilizaremos la palabra 'verosimilitud' porque el concepto tiene cierta semejanza con el utilizado en estimación paramétrica.

Dada una muestra $\{\Lambda_1,\Lambda_2,\ldots,\Lambda_N\}$ de una variable aleatoria borrosa, definimos la verosimilitud de una muestra $\{X_1,X_2,\ldots,X_N\}$ de una variable aleatoria clásica como el producto de probabilidades de cada X_i haya sido la auténtica imagen del experimento, de acuerdo con el modelo dado por Λ_i, es decir:

$$p_\Lambda(X)=\prod_{i=1}^{N}p_{\Lambda_i}(X_i)$$

Supongamos ahora que se nos da una muestra de una variable aleatoria clásica X, $\{X_1,X_2,\ldots,X_n\}$, y una muestra de una variable aleatoria borrosa Λ, $\{\Lambda_1,\Lambda_2,\ldots,\Lambda_m\}$. La información mutua entre X y Λ está contenida en el conjunto de valores de información mutua entre X y cada una de las variables contenidas en el soporte de Λ, y la probabilidad de que un valor *t* sea la información mutua correcta es

$$p(IM(X,\Lambda)=t)=\sum_{Y\in C(soporte(\Lambda))}\{p_\Lambda(Y)\mid t=IM(X,Y)\}$$

Como en general los valores $p_{\Lambda_i}(X_i)$ no son conocidos, sino que se dispone de parejas $p_{*\Lambda_i}(X_i) \le p_{\Lambda_i}(X_i) \le p^*{}_{\Lambda_i}(X_i)$, esta última probabilidad es desconocida. En cualquier caso, podemos determinar intervalos de valores que lo contengan. Por ejemplo, definamos

$$[p_{*\Lambda}(X), p^*{}_{\Lambda}(X)] = \bigotimes_{i=1}^{N} [p_{*\Lambda_i}(X_i), p^*{}_{\Lambda_i}(X_i)]$$

entonces podemos afirmar que la probabilidad de que un valor *t* sea el auténtico valor de la información mutua está comprendida en el intervalo

$$[p_*(IM(X,\Lambda)=t), p^*(IM(X,\Lambda)=t)] = \oplus\{[p_{*\Lambda}(Y), p^*{}_{\Lambda}(Y)] \mid t = IM(X,Y)\}$$

donde la suma se extiende, como en las fórmulas anteriores, al conjunto de las muestras Y contenidas en el soporte del conjunto aleatorio Λ.

3.3.2.2.2. Ejemplo

Considere las siguientes muestras de tamaño tres de las variables Λ y X:

Λ	X
{0.0/FRIO+0.2/TEMPLADO+0.9/CALIENTE}	A
{0.4/FRIO+0.6/TEMPLADO+0.0/CALIENTE}	B
{1.0/FRIO+0.0/TEMPLADO+0.0/CALIENTE}	A

Tabla 3.7. Muestra de un conjunto borroso de tamaño 3 con datos imprecisos

Se desea estimar la información mutua entre X y Λ. En primer lugar, generamos el conjunto de muestras contenidas en el soporte de Λ, y calculamos intervalos que contienen las verosimilitudes de cada una de ellas. Llamemos $Y_1, \ldots, Y_4$ a esas muestras:

Y_1	X
TEMPLADO	A
FRIO	B
FRIO	A
$Verosimilitud \in [0,0.2] \otimes 0.4 \otimes 1 = [0,0.08]$	

Tabla 3.8. Alternativa 1

Y_2	X
TEMPLADO	A
TEMPLADO	B
FRIO	A
$Verosimilitud \in [0,0.2] \otimes 0.6 \otimes 1 = [0,0.12]$	

Tabla 3.9. Alternativa 2

Y_3	X
CALIENTE	A
FRIO	B
FRIO	A
$Verosimilitud \in [0.8,0.9] \otimes 0.4 \otimes 1 = [0,32,0.36]$	

Tabla 3.10. Alternativa 3

Y_4	X
CALIENTE	A
TEMPLADO	B
FRIO	A
$Verosimilitud \in [0.8,0.9] \otimes 0.6 \otimes 1 = [0.48,0.54]$	

Tabla 3.11. Alternativa 4

A continuación, estimamos las informaciones mutuas $IM(Y_1, X), \ldots, IM(Y_4, X)$, y las relacionamos con las probabilidades de las muestras correspondientes:

IM	Verosimilitud
$IM(Y_1, X) = 0.1744$	[0,0.08]
$IM(Y_2, X) = 0.1744$	[0,0.12]
$IM(Y_3, X) = 0.1744$	[0.32,0.36]
$IM(Y_4, X) = 0.6365$	[0.48,0.54]

Tabla 3.12. Relación entre IM y Aceptabilidad

Por tanto, definimos la información mutua entre Λ y X como el conjunto {0.5441,1.2108}, y le asociamos a cada elemento el intervalos de probabilidad que sigue:

$$p(IM(X,\Lambda)=0.1744)=[0,0.08]\oplus[0,0.12]\oplus[0.32,0.36]=[0.32,0.56]$$
$$p(IM(X,\Lambda)=0.6365)=[0.48,0.54]$$

Por último, determinaremos que el valor medio de la información mutua entre X y Λ está contenido en el intervalo:

$$E(IM(X,\Lambda))=0.1744\otimes[0.32,0.56]\oplus 0.6365\otimes[0.48,0.54]=$$
$$[0.0913,0.4414]$$

En este caso, lo que se obtiene es un intervalo. Cuando los datos son imprecisos, se arrastra la imprecisión en forma de intervalo con un límite superior y un límite inferior. El valor de información mutua esta comprendido en ese intervalo.

3.3.3. Estimaciones computacionalmente eficientes

El número de muestras contenidas en una muestra borrosa puedes ser, en el peor caso, el número de etiquetas elevado al volumen de la muestra. Claramente, no es factible el enumerar con un computador el conjunto de muestras compatibles en problemas de tamaño medio o alto.

En esta sección propondremos técnicas aproximadas de estimación de la información mutua borrosa. El objetivo de estas técnicas es evitar el cálculo de la información mutua para todas las muestras contenidas en la muestra borrosa original, reempleando este por un subconjunto, tan pequeño como sea posible, de muestras representativas. Se tratará de generalizar los resultados obtenidos sobre este conjunto al problema general.

Hay dos enfoques que pueden seguirse en esta aproximación:

a) Selección de ejemplos: se elige un número reducido de elementos de la muestra borrosa, y se analizan todas las variables contenidas en una nueva submuestra. A su vez, esta selección se realiza varias veces de forma independiente, y se combinan los resultados obtenidos en cada una de ellas para construir la solución final.

b) Selección de variables contenidas: se elige una fracción de todas las variables contenidas en la muestra original, y se estima el conjunto de valores de información a partir de estas variables.

Ambos caminos se seguirán, con la ayuda de ejemplos numéricos, en las secciones que siguen.

3.3.3.1. Aproximaciones basadas en reducción del número de ejemplos

Dependiendo de la complejidad del problema, es posibles que con una fracción de la muestra haya datos suficientes como para estimar la información mutua entre dos variables.

Supongamos que nos dan muestras de tamaño N de dos variables Λ y X, y que el conjunto imagen de la variable Λ tiene T términos. El número de muestras contenidas será del orden T^N, y la complejidad del algoritmo se estimación de la información mutua es de este mismo orden. Si uno divide ambas muestras en dos mitades, el número de variables contenidas en cada una de ellas será $T^{N/2}$. El tiempo necesario para hacer la estimación de la información en ambas mitades es $2T^{N/2} < T^N$, y la desigualdad es más acusada cuanto mas alto es N. Si cada mitad se divide recursivamente en otras dos, el tiempo será $4T^{N/4}$, y así sucesivamente. Las cuestiones a responder son:

a) Cómo se combinan las estimaciones de la información mutua obtenidas en cada submuestra, para obtener la estimación global.

b) Cuántas subdivisiones pueden hacerse antes de que la estimación combinada difiera significativamente del valor deseado.

La misma ganancia de velocidad puede conseguirse si, en vez de particionar la muestras, aplicamos sucesivamente la estimación a submuestras extraídas aleatoriamente y combinamos los resultados. En la siguiente sección haremos un estudio acerca de la influencia del tamaño y número de demuestras (muestras con reemplazo) de las variables en los resultados de la estimación.

El procedimiento de combinación de las estimaciones obtenidas en cada submuestras será como sigue:

a) Definición tipo I: La información mutua se construye combinando con el operador 'máximo' las estimaciones obtenidas en las muestras.

b) Definición II. La información mutua se construye combinando con el operador 'suma intercalar', seguida de la normalización, las estimaciones de las submuestras.

3.3.3.2. Aproximaciones basadas en muestreo de variables contenidas

Intuitivamente, no es esperable que en la definición de tipo I el segundo de los enfoques (selección de variables contenidas) produzca buenos resultados. Dado que el conjunto borroso que define la información mutua de tipo I se construye mediante el máximo de las pertenencias asociadas a las aceptabilidades, a menos que ese valor máximo aparezca una fracción significativa de veces, es probable que tal valor no se encuentre en el subconjunto de las variables contenidas que hayamos considerado, y por tanto la estimación no necesariamente convergerá al valor real aún cuando el tamaño de la muestra se acerque al total. Esto es así, por el contrario, para la definición tipo II, donde veremos que la distribución de probabilidad se los ejemplos es , en muchos casos prácticos, aproximable medinate una función suave y que las aproximaciones basadas en subconjuntos pequeños de variables convergen rápidamente a la solución exhaustiva.

3.3.3.3. Una propuesta de estimación booststrap

Esta está sección aplicaremos la estimación booststrap, consistente en tomar un subconjunto de la muestra de forma aleatoria para calcular la información mutua. Una vez calculada la información mutua de la muestra seleccionada, se volverá a tomar otra submuestra de tal forma que los ejemplos de la submuestra anterior pueden formar parte de la nueva. Esta operación se repite N veces y se obtienen N valores de información mutua que habrá que combinar y si se trata de

a) la definición I, se combinan sin mas los conjuntos borrosos con el operador 'máximo'

b) la definición II, se hace la media de los intervalos medios obtenidos en cada submuestra

3.3.3.3.1. Una propuesta de estimación booststrap con la definición I

Considere una muestra en la cual tenemos pesos de frutas y cuya clase es el tipo de fruta. Suponemos una muestra de tamaño cinco de las variables Λ y X:

Λ	X
{0.0/PEQUEÑO+0.4/MEDIANO+0.6/GRANDE}	Pera
{0.6/PEQUEÑO+0.4/MEDIANO+0.0/GRANDE}	Manzana
{0.0/PEQUEÑO+0.0/MEDIANO+1.0/GRANDE}	Pera
{1.0/PEQUEÑO+0.0/MEDIANO+0.0/GRANDE}	Plátano
{0.2/PEQUEÑO+0.8/MEDIANO+0.0/GRANDE}	Manzana

Tabla 3.13. Muestra de un conjunto borroso de tamaño 5 con datos precisos

Se desea estimar la información mutua entre X y Λ aplicando una estimación booststrap. El número de muestras que como máximo se pueden generar es 2^3=8 ya que hay dos ejemplos para los cuales sabemos la etiqueta lingüística con exactitud y son el 3 y el 4. Además, un valor de un atributo de un ejemplo pertenecerá como mucho a dos variables lingüísticas, cuya suma de las pertenencias será uno.

En primer lugar, hay que decidir cual será el número de muestras que formara el subconjunto de la muestra a evaluar y también el número se veces que se repite el proceso. Vamos a suponer que la muestra a evaluar esta formada por el 75% de las muestras, seleccionadas estas de forma aleatoria, y que repetimos el proceso 5 veces.

Generaremos el subconjunto de muestras seleccionadas de forma aleatoria y que están contenidas en el soporte de Λ, y calcularemos la aceptabilidad de cada una de ellas. Suponemos que las muestras seleccionadas en este caso son la 1, 2, 3, 5, 7 y 8 y las llamaremos Y_i:

Y_1	X
MEDIANO	Pera
PEQUEÑO	Manzana
GRANDE	Pera
PEQUEÑO	Plátano
PEQUEÑO	Manzana
Aceptabilidad=min{0.4,0.6,1.0,1.0,0.2}=0.2	

Tabla 3.14. Alternativa 1

Y_2	X
MEDIANO	Pera
MEDIANO	Manzana
GRANDE	Pera
PEQUEÑO	Plátano
PEQUEÑO	Manzana
Aceptabilidad=min{0.4,0.4,1.0,1.0,0.2}=0.2	

Tabla 3.15. Alternativa 2

Y_3	X
MEDIANO	Pera
PEQUEÑO	Manzana
GRANDE	Pera
PEQUEÑO	Plátano
MEDIANO	Manzana
Aceptabilidad=min{0.4,0.6,1.0,1.0,0.8}=0.4	

Tabla 3.16. Alternativa 3

Y_5	X
GRANDE	Pera
PEQUEÑO	Manzana
GRANDE	Pera
PEQUEÑO	Plátano
PEQUEÑO	Manzana
Aceptabilidad=min{0.6,0.6,1.0,1.0,0.2}=0.2	

Tabla 3.17. Alternativa 5

Y_7	X
GRANDE	Pera
PEQUEÑO	Manzana
GRANDE	Pera
PEQUEÑO	Plátano
MEDIANO	Manzana
Aceptabilidad=min{0.6,0.6,1.0,1.0,0.8}=0.6	

Tabla 3.18. Alternativa 7

Y_8	X
GRANDE	Pera
MEDIANO	Manzana
GRANDE	Pera
PEQUEÑO	Plátano
MEDIANO	Manzana
Aceptabilidad=min{0.6,0.4,1.0,1.0,0.8}=0.4	

Tabla 3.19. Alternativa 8

A continuación, estimamos las informaciones mutuas $IM(Y_i, X)$, y las relacionamos con las aceptabilidades de las muestras correspondientes:

IM	Aceptabilidad
$IM(Y_1, X) = 0.6730$	0.2
$IM(Y_2, X) = 0.5004$	0.2
$IM(Y_3, X) = 0.5004$	0.4
$IM(Y_5, X) = 0.6730$	0.2
$IM(Y_7, X) = 0.7777$	0.6
$IM(Y_8, X) = 1.0549$	0.4

Tabla 3.20. Relación entre Aceptabilidad e IM para el primer subconjunto

Por último, definimos la información mutua entre Λ y X como el conjunto borroso correspondiente a la máxima aceptabilidad asociada a cada posible valor de IM:

$$IM(X, \Lambda) = 0.4/0.5004 + 0.2/0.6730 + 0.6/0.7777 + 0.4/1.0549$$

Este sería el resultado para el primer subconjunto de a muestra seleccionado. Ahora habría que seleccionar los otros 4 restantes y volver a hacer todos los cálculos. No vamos a mostrar todo, tan sólo el resultado final.

Subconjunto de la muestra 2: 2, 3, 5, 6, 7 y 8

IM	Aceptabilidad
$IM(Y_2, X) = 0.5004$	0.2
$IM(Y_3, X) = 0.5004$	0.4
$IM(Y_5, X) = 0.6730$	0.2
$IM(Y_6, X) = 0.7777$	0.2
$IM(Y_7, X) = 0.7777$	0.6
$IM(Y_8, X) = 1.0549$	0.4

Tabla 3.21. Relación entre Aceptabilidad e IM para el segundo subconjunto

La información mutua entre Λ y X será:

$$IM(X, \Lambda) = 0.4 / 0.5004 + 0.2 / 0.6730 + 0.6 / 0.7777 + 0.4 / 1.0549$$

Subconjunto de la muestra 3: 1, 4, 5, 6, 7 y 8

IM	Aceptabilidad
$IM(Y_1, X) = 0.6730$	0.2
$IM(Y_4, X) = 0.6730$	0.4
$IM(Y_5, X) = 0.6730$	0.2
$IM(Y_6, X) = 0.7777$	0.2
$IM(Y_7, X) = 0.7777$	0.6
$IM(Y_8, X) = 1.0549$	0.4

Tabla 3.22. Relación entre Aceptabilidad e IM para el tercer subconjunto

La información mutua entre Λ y X será:

$$IM(X, \Lambda) = 0.4 / 0.6730 + 0.6 / 0.7777 + 0.4 / 1.0549$$

Subconjunto de la muestra 4: 1, 2, 3, 4, 5 y 6

IM	Aceptabilidad
$IM(Y_1, X) = 0.6730$	0.2
$IM(Y_2, X) = 0.5004$	0.2
$IM(Y_3, X) = 0.5004$	0.4
$IM(Y_4, X) = 0.6730$	0.4
$IM(Y_5, X) = 0.6730$	0.2
$IM(Y_6, X) = 0.7777$	0.2

Tabla 3.23. Relación entre Aceptabilidad e IM para el cuarto subconjunto

La información mutua entre Λ y X será:

$$IM(X, \Lambda) = 0.4 / 0.5004 + 0.4 / 0.6730 + 0.2 / 0.7777$$

Subconjunto de la muestra 5: 3, 4, 5, 6, 7 y 8

IM	Aceptabilidad
$IM(Y_3, X) = 0.5004$	0.4
$IM(Y_4, X) = 0.6730$	0.4
$IM(Y_5, X) = 0.6730$	0.2
$IM(Y_6, X) = 0.7777$	0.2
$IM(Y_7, X) = 0.7777$	0.6
$IM(Y_8, X) = 1.0549$	0.4

Tabla 3.24. Relación entre Aceptabilidad e IM para el quinto subconjunto

La información mutua entre Λ y X será:

$$IM(X, \Lambda) = 0.4 / 0.5004 + 0.4 / 0.6730 + 0.6 / 0.7777 + 0.4 / 1.0549$$

Al final tenemos tantos resultados de información mutua como veces se haya realizado el proceso.

$$IM(X,\Lambda) = 0.4/0.5004 + 0.2/0.6730 + 0.6/0.7777 + 0.4/1.0549$$
$$IM(X,\Lambda) = 0.4/0.5004 + 0.2/0.6730 + 0.6/0.7777 + 0.4/1.0549$$
$$IM(X,\Lambda) = 0.4/0.6730 + 0.6/0.7777 + 0.4/1.0549$$
$$IM(X,\Lambda) = 0.4/0.5004 + 0.4/0.6730 + 0.2/0.7777$$
$$IM(X,\Lambda) = 0.4/0.5004 + 0.4/0.6730 + 0.6/0.7777 + 0.4/1.0549$$

Estos resultados se agregan mediante la operación de máximo y la representación de gráfica del resultado de esta agregación se muestra en la figura 3.3.

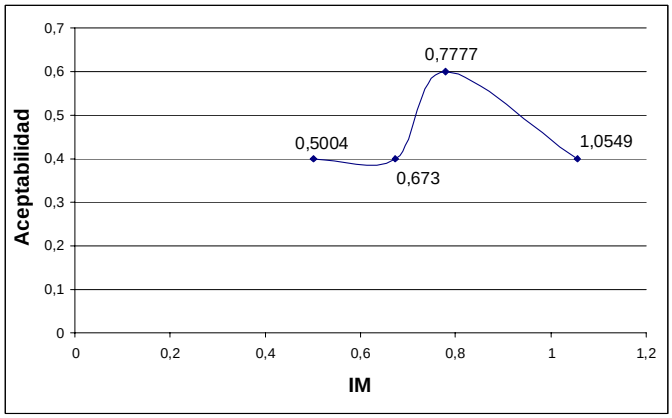


Fig 3.3. IM con estimación bootstrap. Definición I

La idea de partida fue es que el resultado de realizar todo el proceso de calculo de la información mutua con estimación bootstrap sea lo mas parecido posible al resultado obtenido sin estimación bootstrap para así poder concluir que este tipo de estimación además de obtener los resultados deseados, es computacionalmente mas eficiente.

Si evalúo todas las muestras sin estimación bootstrap, el resultado es el siguiente:

IM	Aceptabilidad
$IM(Y_1, X) = 0.6730$	0.2
$IM(Y_2, X) = 0.5004$	0.2
$IM(Y_3, X) = 0.5004$	0.4
$IM(Y_4, X) = 0.6730$	0.4
$IM(Y_5, X) = 0.6730$	0.2
$IM(Y_6, X) = 0.7777$	0.2
$IM(Y_7, X) = 0.7777$	0.6
$IM(Y_8, X) = 1.0549$	0.4

Tabla 3.25. Relación entre Aceptabilidad e IM para todos las muestras posibles.

$$IM(X, \Lambda) = 0.4/0.5004 + 0.4/0.6730 + 0.6/0.7777 + 0.4/1.0549$$

Y cuya representación gráfica sería la siguiente:

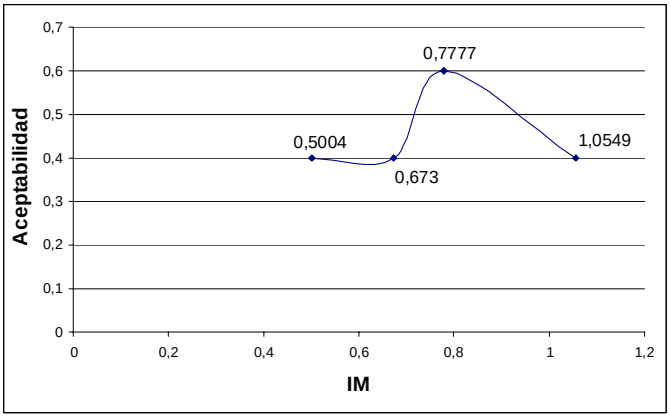


Fig 3.4. IM sin estimación bootstrap. Definición I

Si comparamos la Fig 3.3 y la Fig 3.4. se observa, para este caso que los resultados son los mismo y computacionalmente hablando, la opción de utilizar bootstrap, es mejor, ya que el numero d muestra analizadas es menor.

En este ejemplo, la estimación bootstrap se ha realizado a partir de 6 evaluaciones de la información mutua crisp. La estimación completa necesita 8 evaluaciones, lo que supone un ahorro del 25%. De acuerdo con nuestra experimentación (ver apéndice C) la estimación bootstrap no es eficaz a menos que evaluemos un porcentaje muy alto de los casos. Es necesario un algoritmo aproximado que pueda proporcionar resultados a partir de, a lo sumo algunos miles de evaluaciones de la información Mutua

3.3.3.3.2. Una propuesta de estimación booststrap con la definición II

Considérese de nuevo una muestra en la cual tenemos pesos de frutas y cuya clase es el tipo de fruta. En este caso tenemos una información imprecisa ya que las suma de las pertenencias para algunos ejemplos es mayor que 1. Suponemos una muestra de tamaño cinco de las variables Λ y X:

Λ	X
{0.0/PEQUEÑO+0.4/MEDIANO+0.9/GRANDE}	Pera
{0.8/PEQUEÑO+0.4/MEDIANO+0.0/GRANDE}	Manzana
{0.0/PEQUEÑO+0.0/MEDIANO+1.0/GRANDE}	Pera
{1.0/PEQUEÑO+0.0/MEDIANO+0.0/GRANDE}	Plátano
{0.2/PEQUEÑO+0.8/MEDIANO+0.0/GRANDE}	Manzana

Tabla 3.26. Muestra de un conjunto borroso de tamaño 5 con datos imprecisos

Se desea estimar la información mutua entre X y Λ aplicando una estimación bootstrap. El número de muestras que se pueden generar es 2^3=8 ya que hay dos ejemplos para los cuales sabemos la etiqueta lingüística con exactitud y son el 3 y el 4.

En primer lugar, hay que decidir cual será el número de muestras que formara el subconjunto de la muestra a evaluar y también el número se veces que se repite el proceso. Vamos a suponer que la muestra a evaluar esta formada por el 75% de las muestras, seleccionada esta de forma aleatoria y que repetimos el proceso 5 veces.

Generaremos el subconjunto de muestras seleccionadas de forma aleatoria y que están contenidas en el soporte de Λ, y calcularemos intervalos que contienen las verosimilitudes de cada una de ellas. Suponemos que las muestras seleccionadas en este caso son la 1, 2, 3, 5, 7 y 8 y las llamaremos Y_i:

Y_1	X
MEDIANO	Pera
PEQUEÑO	Manzana
GRANDE	Pera
PEQUEÑO	Plátano
PEQUEÑO	Manzana
$Verosimilitud \in [0,0.4] \otimes [0.6,0.8] \otimes 1 \otimes 1 \otimes 0.2 = [0,0.072]$	

Tabla 3.27. Alternativa 1

Y_2	X
MEDIANO	Pera
MEDIANO	Manzana
GRANDE	Pera
PEQUEÑO	Plátano
PEQUEÑO	Manzana
$Verosimilitud \in [0,0.4] \otimes [0,0.4] \otimes 1 \otimes 1 \otimes 0.2 = [0,0.032]$	

Tabla 3.28. Alternativa 2

Y_3	X
MEDIANO	Pera
PEQUEÑO	Manzana
GRANDE	Pera
PEQUEÑO	Plátano
MEDIANO	Manzana
$Verosimilitud \in [0,0.4] \otimes [0.6,0.8] \otimes 1 \otimes 1 \otimes 0.8 = [0,0.256]$	

Tabla 3.29. Alternativa 3

Y_5	X
GRANDE	Pera
PEQUEÑO	Manzana
GRANDE	Pera
PEQUEÑO	Plátano
PEQUEÑO	Manzana
$Verosimilitud \in [0.6,0.9] \otimes [0.6,0.8] \otimes 1 \otimes 1 \otimes 0.2 = [0.072,0.144]$	

Tabla 3.30. Alternativa 5

Y_7	X
GRANDE	Pera
PEQUEÑO	Manzana
GRANDE	Pera
PEQUEÑO	Plátano
MEDIANO	Manzana
$Verosimilitud \in [0.6,0.9] \otimes [0.6,0.8] \otimes 1 \otimes 1 \otimes 0.8 = [0.288,0.576]$	

Tabla 3.31. Alternativa 7

Y_8	X
GRANDE	Pera
MEDIANO	Manzana
GRANDE	Pera
PEQUEÑO	Plátano
MEDIANO	Manzana
$Verosimilitud \in [0.6,0.9] \otimes [0,0.4] \otimes 1 \otimes 1 \otimes 0.8 = [0,0.288]$	

Tabla 3.32. Alternativa 8

A continuación, estimamos las informaciones mutuas $IM(Y_i, X)$, y las relacionamos con las probabilidades de las muestras correspondientes:

IM	Verosimilitud
$IM(Y_1, X) = 0.6730$	[0,0.072]
$IM(Y_2, X) = 0.5004$	[0,0.032]
$IM(Y_3, X) = 0.5004$	[0,0.256]
$IM(Y_5, X) = 0.6730$	[0.072,0.144]
$IM(Y_7, X) = 0.7777$	[0.288,0.576]
$IM(Y_8, X) = 1.0549$	[0,0.288]

Tabla 3.33. Relación entre Aceptabilidad e IM para el primer subconjunto

Por tanto, definimos la información mutua entre Λ y X como el conjunto {0.5004, 0.6730, 0.7777, 1.0549}, y le asociamos a cada elemento el intervalo de probabilidad que sigue:

$$p(IM(X,\Lambda)=0.5004)=[0,0.032]\oplus[0,0.256]=[0,0.288]$$
$$p(IM(X,\Lambda)=0.6730)=[0,0.072]\oplus[0.072,0.144]=[0.072,0.216]$$
$$p(IM(X,\Lambda)=0.7777)=[0.288,0.576]$$
$$p(IM(X,\Lambda)=1.0549)=[0,0.288]$$

Por último, determinaremos que el valor medio de la información mutua entre X y Λ está contenido en el intervalo:

$$E(IM(X,\Lambda))=0.5004\otimes[0,0.288]\oplus 0.673\otimes[0.072,0.216]\oplus$$
$$0.7777\otimes[0.288,0.576]\oplus 1.5049\otimes[0,0.288]=[0.2724,1.1708]$$

Este sería el resultado para el primer subconjunto de a muestra seleccionado. Ahora habría que seleccionar los otros 9 restantes y volver a hacer todos los cálculos. No vamos a mostrar todo, tan sólo el resultado final.

Subconjunto de la muestra 2: 2, 3, 5, 6, 7 y 8

IM	Verosimilitud
$IM(Y_2,X)=0.5004$	[0,0.032]
$IM(Y_3,X)=0.5004$	[0,0.256]
$IM(Y_5,X)=0.6730$	[0.072,0.144]
$IM(Y_6,X)=0.7777$	[0,0,072]
$IM(Y_7,X)=0.7777$	[0.288,0.576]
$IM(Y_8,X)=1.0549$	[0,0.288]

Tabla 3.34. Relación entre Aceptabilidad e IM para el segundo subconjunto

Por tanto, definimos la información mutua entre Λ y X como el conjunto {0.5004, 0.6730, 0.7777, 1.0549}, y le asociamos a cada elemento el intervalo de probabilidad que sigue:

$$p(IM(X,\Lambda)=0.5004)=[0,0.032]\oplus[0,0.256]=[0,0.288]$$
$$p(IM(X,\Lambda)=0.6730)=[0.072,0.144]$$
$$p(IM(X,\Lambda)=0.7777)=[0,0.072]\oplus[0.288,0.576]=[0.288,0.648]$$
$$p(IM(X,\Lambda)=1.0549)=[0,0.288]$$

Por último, determinaremos que el valor medio de la información mutua entre X y Λ está contenido en el intervalo:

$$E(IM(X,\Lambda)) = 0.5004 \otimes [0,0.288] \oplus 0.673 \otimes [0.072,0.216] \oplus$$
$$0.7777 \otimes [0.288,0.648] \oplus 1.5049 \otimes [0,0.288] = [0.2724,0.9387]$$

Subconjunto de la muestra 3: 1, 4, 5, 6, 7 y 8

IM	Verosimilitud
$IM(Y_1, X) = 0.6730$	[0,0.072]
$IM(Y_4, X) = 0.6730$	[0,0.128]
$IM(Y_5, X) = 0.6730$	[0.072,0.144]
$IM(Y_6, X) = 0.7777$	[0,0,072]
$IM(Y_7, X) = 0.7777$	[0.288,0.576]
$IM(Y_8, X) = 1.0549$	[0,0.288]

Tabla 3.35. Relación entre Aceptabilidad e IM para el tercer subconjunto

Por tanto, definimos la información mutua entre Λ y X como el conjunto {0.6730, 0.7777, 1.0549}, y le asociamos a cada elemento el intervalo de probabilidad que sigue:

$$p(IM(X,\Lambda) = 0.6730) = [0,0.072] \oplus [0,0.128] \oplus [0.072,0.144] = [0.072,0.344]$$
$$p(IM(X,\Lambda) = 0.7777) = [0,0.072] \oplus [0.288,0.576] = [0.288,0.648]$$
$$p(IM(X,\Lambda) = 1.0549) = [0,0.288]$$

Por último, determinaremos que el valor medio de la información mutua entre X y Λ está contenido en el intervalo:

$$E(IM(X,\Lambda)) = 0.673 \otimes [0.072,0.344] \oplus$$
$$0.7777 \otimes [0.288,0.648] \oplus 1.5049 \otimes [0,0.288] = [0.2723,1.1688]$$

Subconjunto de la muestra 4: 1, 2, 3, 4, 5 y 6

IM	Verosimilitud
$IM(Y_1, X) = 0.6730$	[0,0.072]
$IM(Y_2, X) = 0.5004$	[0,0.032]
$IM(Y_3, X) = 0.5004$	[0,0.256]
$IM(Y_4, X) = 0.6730$	[0,0.128]
$IM(Y_5, X) = 0.6730$	[0.072,0.144]
$IM(Y_6, X) = 0.7777$	[0,0,072]

Tabla 3.36. Relación entre Aceptabilidad e IM para el cuarto subconjunto

Por tanto, definimos la información mutua entre Λ y X como el conjunto {0.5004, 0.6730, 0.7777}, y le asociamos a cada elemento el intervalo de probabilidad que sigue:

$$p(IM(X,\Lambda) = 0.5004) = [0,0.032] \oplus [0,0.256] = [0,0.288]$$
$$p(IM(X,\Lambda) = 0.6730) = [0,0.072] \oplus [0,0.128] \oplus [0.072,0.144] = [0.072,0.344]$$
$$p(IM(X,\Lambda) = 0.7777) = [0.288,0.576]$$

Por último, determinaremos que el valor medio de la información mutua entre X y Λ está contenido en el intervalo:

$$E(IM(X,\Lambda)) = 0.5004 \otimes [0,0.288] \oplus 0.673 \otimes [0.072,0.344] \oplus$$
$$0.7777 \otimes [0.288,0.576] = [0.2723,0.8235]$$

Subconjunto de la muestra 5: 3, 4, 5, 6, 7 y 8

IM	Verosimilitud
$IM(Y_3, X) = 0.5004$	[0,0.256]
$IM(Y_4, X) = 0.6730$	[0,0.128]
$IM(Y_5, X) = 0.6730$	[0.072,0.144]
$IM(Y_6, X) = 0.7777$	[0,0,072]
$IM(Y_7, X) = 0.7777$	[0.288,0.576]
$IM(Y_8, X) = 1.0549$	[0,0.288]

Tabla 3.37. Relación entre Aceptabilidad e IM para el tercer subconjunto

Por tanto, definimos la información mutua entre Λ y X como el conjunto {0.5004, 0.6730, 0.7777, 1.0549}, y le asociamos a cada elemento el intervalo de probabilidad que sigue:

$$p(IM(X,\Lambda)=0.5004)=[0,0.256]$$
$$p(IM(X,\Lambda)=0.6730)=[0,0.128]\oplus[0.072,0.144]=[0.072,0.272]$$
$$p(IM(X,\Lambda)=0.7777)=[0,0.072]\oplus[0.288,0.576]=[0.288,0.648]$$
$$p(IM(X,\Lambda)=1.0549)=[0,0.288]$$

Por último, determinaremos que el valor medio de la información mutua entre X y Λ está contenido en el intervalo:

$$E(IM(X,\Lambda))=0.5004\otimes[0,0.256]\oplus 0.673\otimes[0.072,0.272]\oplus$$
$$0.7777\otimes[0.288,0.648]\oplus 1.5049\otimes[0,0.288]=[0.2723,1.2484]$$

Al final tengo tantos resultados de información mutua como veces se haya realizado el proceso.

CAPITULO 4

Aplicación al diseño de particiones borrosas en problemas de clasificación

4.1. Introducción

La información mutua definida en el capítulo anterior sirve para cuantificar la dependencia entre dos variables aleatorias borrosas. En general, como se ha visto, las muestras de variables aleatorias. En general, como se ha visto, las muestras de variables aleatorias borrosas se originan a transformar una variable numérica mediante un interfaz de fuzzificación. En el caso de que la variable numérica se mida con imprecisión, la suma de las pertenencias de todos los términos lingüísticos sumará más que la unidad. Lo mismo ocurrirá cuando el valor de la variable sea desconocido, porque esta circunstancia puede verse como un caso límite de medida imprecisa.

Además de la justificación teórica de nuestra definición, en esta tesis vamos a proponer una aplicación práctica de la misma. El propósito de esta aplicación, como se verá, es el de mostrar la coherencia de los conceptos expuestos en el tema anterior. Pretendemos mostrar que la aplicación de la información mutua borrosa en problemas originados en mediciones precisas es compatible con la visión clásica, y que de forma natural podemos proponer conjuntos de datos incompletos y/o imprecisos.

Hay numerosas aplicaciones de la Información Mutua que podemos generalizar a datos imprecisos. La mas conocida sería, tal vez, la selección de características, donde podríamos tomar algoritmos como el de Battiti [Bat94] y extenderlos a los nuevos tipos de datos. Sin embargo, y aunque pretendemos realizar este estudio en el futuro, la selección de características es un problema con suficiente entidad como para realizar una nueva tesis. Dado que nuestro interés se centra en mostrar la coherencia de las medidas de cantidad de información propuestas en esta tesis, hemos decidido resolver un problema más sencillo, la selección automática de la partición borrosa más informativa, suponiendo además conocido el número de términos lingüísticos. Aunque este problema también plantea algunas dificultades numéricas, que se explicarán en este capítulo, podemos confiar en que los malos resultados serán consecuencia de inconsistencias en la definición y no se deberían a un tratamiento incorrecto de la optimización numérica de una función que toma valores intercalares o borrosos.

4.2. Solución algorítmica

En esta sección presentaremos la solución algorítmica pensada para abordar los dos tipos de problemas que se nos plantean. Dichos algoritmos se

aplicarán a un problema concreto que es el de optimización de particiones lingüísticas en un problema de clasificación. Una partición óptima será la que maximice la información mutua con respecto a la variable 'clase'. Se usara la definición II con aproximación bootstrap.

En un sistema de clasificación, la forma que tengo de saber si dicho sistema es eficiente o no, es mediante el error medio de clasificación. En nuestro intento por demostrar que nuestros resultados son iguales o mejores que los obtenidos con el sistema clásico se utilizan los sistemas de clasificación basados en reglas borrosas, como herramienta de verificación de resultados. Particularizando en el problema que nos sirve de herramienta para esta tesis que es la optimización de particiones, la idea de la implementación es obtener el error medio de clasificación utilizando una partición definida de forma uniforme, optimizar dicha partición mediante las técnicas genéticas correspondientes y por último volver a obtener el error medio de clasificación, ahora con la partición optimizada, para compararlo con el valor inicial y sacar las conclusiones pertinentes.

El esquema gráfico de los contenidos del capítulo 4 es el que se muestra en al figura 4.1.

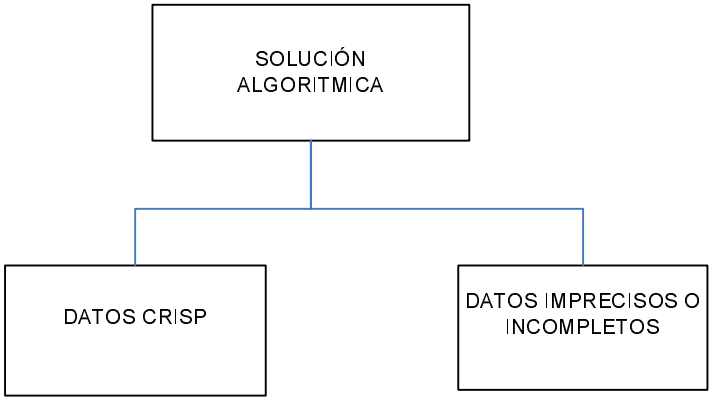


Fig 4.1. Soluciones algorítmicas

Se plantean dos soluciones algorítmicas para nuestro problema, la primera de ellas para el tratamiento de datos crips sin imprecisión mediante un algoritmo genético clásico, y la segunda es una a adaptación del algoritmo NSGA II para el caso de datos imprecisos.

4.2.1. Diseño genético de particiones borrosas

En el marco de nuestro trabajo de investigación, el principal objetivo que se persigue es la búsqueda de un conjunto de particiones optimizadas a partir de un conjunto inicial. Se trata de utilizar un algoritmo genético para resolver el problema de tal forma que nos proporcione una familia de particiones optimizadas de las cuales podré saber cual es la que mejor se adapta al problema mediante los métodos planteados en secciones anteriores.

Se plantean dos tipos de problemas, aquel en el que el dataset esta formado por datos precisos que vienen determinados por un valor concreto y aquellos problemas cuyo dataset es un conjunto de datos impresisos cuyos valores viene determinados por el intervalo de imprecisión como hemos vistos en la sección 3.3.2. En el primer caso los valores de información mutua serán números y en el segundo caso, los valores de información mutua se corresponden con intervalos.

En ambos casos se puede solucionar la búsqueda con un algoritmo genético. Para el caso de optimización de particiones con datos precisos, se implementará un algoritmo genético clásico que utilice la definición de Información Mutua para calcular lo bueno o malo que son los individuos generados, en nuestro caso, particiones. Para el caso de optimización de particiones con datos imprecisos se utilizará el algoritmo NSGA-II con ciertas modificaciones; en particular proporcionaremos nuestra propia formulación de todas las operaciones en que el algoritmo NSGA-II se base en que el valor de la función objetivo es un vector de números reales y no un intervalo.

4.3. Conocimiento completo de la probabilidad de observación: problema con orden total. Definición del sistema genético

En este apartado mostraremos las características de nuestro algoritmo genético que servirá para solucionar la problemática de la definición II en problemas con datos no imprecisos.

Antes de comenzar a describir el sistema genético mostraremos un esquema de lo que tiene que hacer:

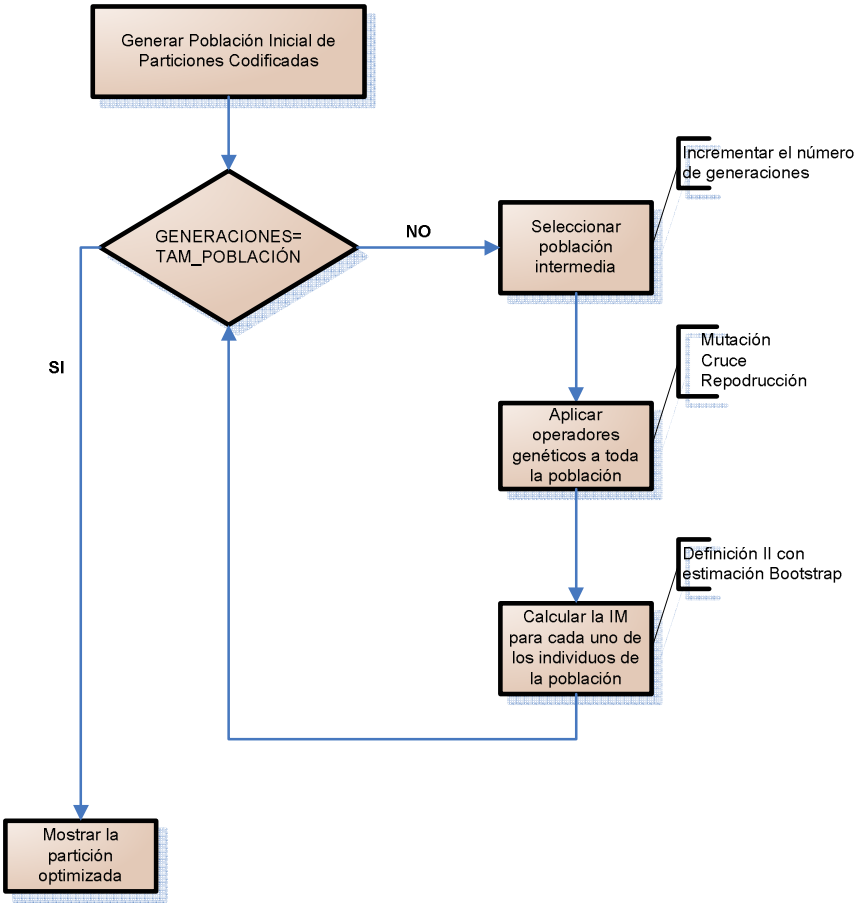


Fig 4.2. Algoritmo genético para datos crips

Como cualquier algoritmo genético, el primer paso es la definición de la codificación de los individuos, en nuestro caso se utilizará una codificación real de particiones uniformes. Seguidamente y hasta que no se cumpla nuestro condición de parada que indique que ya se ha obtenido la partición optimizada, se aplica el proceso genético.

Nuestro proceso genético consistirá en realizar una selección de los individuos utilizando un muestreo estocástico de reemplazamiento y para cada uno de los individuos de esa nueva generación intermedia aplicar los operadores genéticos de

- cruce, max-min-aritmético [HLV95]
- mutación, no uniforme propuesto por Michalewicz [Mic96]

- reproducción, copia

de forma aleatoria. Una vez aplicados se evalúa la población utilizando la definición I de Información Mutua propuesta aplicando estimación bootstrap.

En las subsecciones siguientes se muestra con detalle tanto el esquema de codificación, como los operadores genéticos, como la selección y evaluación de los individuos de la población, así como ejemplos ilustrativos y sencillos.

4.3.1. Esquema de codificación y población inicial

El esquema de representación o codificación es un factor clave en la aplicación de loas algoritmos genéticos, ya que estos manipulan directamente una representación codificada del problema y, en consecuencia, el esquema escogido puede limitar de una forma muy severa la ventana desde la cual el algoritmo genérico afronta el problema.

Existen distintos esquemas generales de codificación, pero nosotros vamos a utilizar para nuestro problema, una codificación real, es decir, los parámetros reales son las unidades de representación del algoritmo genético (genes). Se podría haber utilizado una codificación binaria, pero esta representa una serie de inconvenientes importantes cuando se trabaja con problemas que incluyen variables definidas sobre dominios continuos: excesiva longitud de los cromosomas, falta de precisión, etc. En los últimos años, se ha estudiado ampliamente la codificación real [HLV98], mas adecuada para este tipo de problema.

En el esquema de representación real, cada variable del problema se asocia a un único gen que toma un valor real dentro del intervalo especificado, y por lo que no existen diferencias entre el genotipo (la codificación empleada y el fenotipo (la propia solución codificada). Gracias a esta propiedad se solucionan los problemas comentados. Si manejamos tres etiquetas lingüísticas por variable, la longitud del cromosoma será de 3 por el número de etiquetas lingüísticas sumando tantas veces como variables se manejen para el problema especificado.

Por ejemplo, supongamos que queremos codificar la siguiente partición borrosa que cumple la condición de que el punto de corte es 0.5:

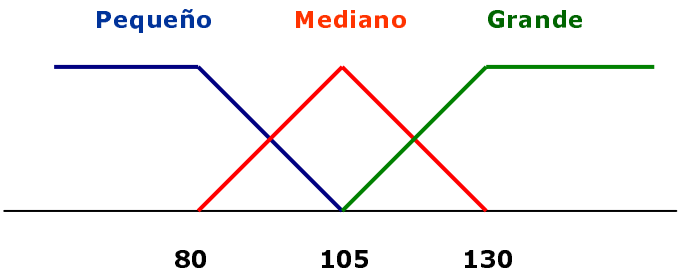


Fig 4.3. Conjunto borroso "tamaño frutas"

80.0	105.0	130.0

Para poder lanzar el algoritmo genético necesitamos disponer de la población inicial. En nuestro caso estará formada por un conjunto de cromosomas cada uno de los cuales representará posibles particiones borrosas para nuestro problema.

Partimos de un cromosoma inicial C_1 con la partición codificada. Para generar los individuos restantes, se asocia un intervalo de rendimiento $[c_h^l, c_h^r]$ a cada gen c_h de C_1, $h = 1, \ldots, \sum_{i=1}^{N} l_i * 3$, que será el intervalo de ajuste para el gen correspondiente, es decir, $c_h \in [c_h^l, c_h^r]$.

Para definir los intervalos de ajuste de cada gen, y así generar posteriormente de forma aleatoria los distintos individuos se sigue el siguiente proceso: si (t mod 3)=1 entonces (t es la generación) el gen c_t representa el valor de la parte izquierda del conjunto soporte de un número difuso. El número difuso se define con los tres parámetros (c_t, c_{t+1}, c_{t+2}) y los intervalos de rendimiento son los siguientes:

$$c_t \in [c_t^l, c_t^r] = \left[c_t - \frac{c_{t+1} - c_t}{2}, c_t + \frac{c_{t+1} - c_t}{2} \right]$$

$$c_{t+1} \in [c_{t+1}^l, c_{t+1}^r] = \left[c_{t+1} - \frac{c_{t+1} - c_t}{2}, c_{t+1} + \frac{c_{t+2} - c_t}{2} \right]$$

$$c_{t+2} \in [c_{t+2}^l, c_{t+2}^r] = \left[c_{t+2} - \frac{c_{t+2} - c_{t+1}}{2}, c_{t+2} + \frac{c_{t+2} - c_{t+1}}{2} \right]$$

Con este proceso de generación de la población inicial se crea una población de cromosomas que contiene a C_1 como primer individuo y los restantes se generan aleatoriamente con cada gen definido dentro de su intervalo de ajuste correspondiente.

4.3.2. Mecanismo de selección

El mecanismo de selección es el encargado de seleccionar la población intermedia de individuos la cual, una vez aplicados los operadores de cruce y mutación, formará la nueva población del algoritmo genético en la siguiente generación. De este modo, si notamos por P la población actual formada por N cromosomas, $C_1,\ldots C_n$, el mecanismo de selección se encarga de obtener una población intermedia P`, formada por copias de los cromosomas de P. El número de veces que se copia cada cromosoma depende de su adecuación, por lo que generalmente aquellos que presentan un valor mayor en la función de adaptación suelen tener mas oportunidades para contribuir con copias a la formación de P`.

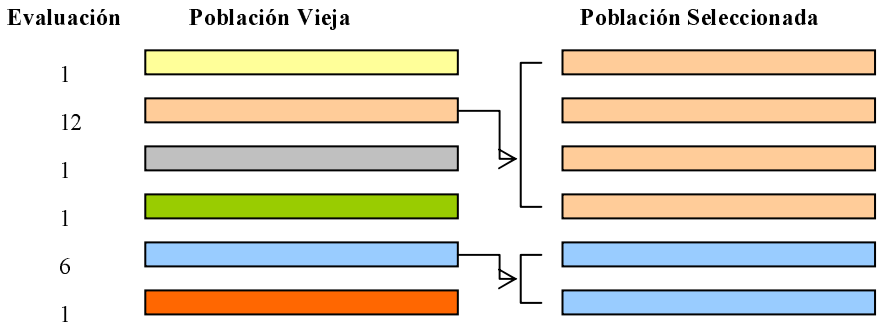


Fig 4.4. Selección

Existen diferentes formas de poner en práctica la selección [BS91]. Por ejemplo, puede establecerse un paralelismo entre la población y una ruleta, en la que cada cromosoma está representado por un sector de la misma cuyo tamaño es proporcional a la adaptación de dicho cromosoma. Los cromosomas se seleccionan girando a ruleta tantas veces como individuos tengamos que seleccionar para formar la población intermedia. Este mecanismo de selección es un de los mas conocidos y se denomina muestreo estocástico con reemplazamiento. Uno de los más eficientes es el muestreo universal estocástico, propuesto por Baker en [BS91], en el cual el número de copias de cada individuo en la población intermedia está acotado

inferior y superiormente por un número de copias esperado y calculado en función de su adaptación.

El mecanismo de selección puede ser completado con el modelo de selección elitista, basado en mantener un número de terminado de los individuos mejor adaptados de la población anterior en la nueva población (la obtenida después de llevar a cabo el proceso de selección y de aplicar los operadores de cruce y mutación) [Gol89, Mic96]

4.3.3. El operador de cruce

Este operador constituye un mecanismo para compartir información entre cromosomas. Combina las características de dos cromosomas padre para obtener dos cromosomas descendientes, con la posibilidad de que los cromosomas hijos obtenidos mediante la recombinación de sus padres, estén mejor adaptados que éstos. No suele ser aplicado a todas las parejas de cromosomas de la población intermedia sino que se lleva a cabo una selección aleatoria en función de una determinada probabilidad de aplicación llamada probabilidad de cruce, P_c.

El operador de cruce utilizado será el max-min-aritmético [HLV95] que emplea herramientas difusas para mejorar el comportamiento del algoritmo genético. Para dos cromosomas C_v y C_w, este último operador genera dos descendientes de la siguiente manera:

$$C_1 = aC_w + (1-a)C_v$$
$$C_2 = aC_v + (1-a)C_w$$
$$C_3 \ con \ c_{3k} = \min\{c_k, c_k^{'}\}$$
$$C_4 \ con \ c_{4k} = \max\{c_k, c_k^{'}\}$$

Este operador utilizará un parámetro *a* que puede tener un valor constante o dependiente de la edad de la población. La descendencia resultante estará formada por los dos mejores individuos de los cuatro hijos generados.

Vamos a ver lo explicado anteriormente con un ejemplo. Supongamos que tenemos dos cromosomas codificados de forma real. Cada uno de los cromosomas codifica una única partición borrosa por simplificar.

C_v	80.0	105.0	130.0
C_w	82.0	107.0	132.0

Fig 4.5. Individuos padres para el cruce

Aplicando el operador de cruce max-min-aritmético y siendo a=0.9, el resultado será el siguiente:

		Cromosomas			Evaluación
	C_1	81.8	106.8	131.8	4
→	C_2	80.2	105.2	130.2	12
→	C_3	80.0	105.0	130.0	6
	C_4	82.0	107.0	132.0	2

Fig 4.6. Hijos después del cruce

4.3.4. El operador de mutación

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

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

El operador de mutación utilizado será el de mutación no uniforme propuesto por Michalewicz [Mic96]. Este operador ha demostrado tener un buen comportamiento en numerosas aplicaciones evolutivas con codificación real. Su funcionamiento es el que se describe a continuación.

Si $C = \{c_1, \ldots, c_i, \ldots, c_n\}$ es un cromosoma y $c_i \in [a_i, b_i]$ es el elemento a mutar es la generación t, el cromosoma resultante

$C' = \{c_1, \ldots, c_i', \ldots, c_n\}$, donde c_i' se determina de acuerdo a la siguiente expresión:

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

siendo r un numero aleatorio perteneciente al conjunto {0,1}, y

$$\Delta(t, y) = y\left(1 - \tau^{\left(1-\frac{1}{T}\right)^b}\right)$$

donde τ un numero aleatorio perteneciente al intervalos [0,1], b es un parámetro elegido por el usuario, que determina el grado de dependencia del número de iteraciones y T es el número de generaciones. Esta función da un valor en el rango [0,y] tal que la probabilidad de devolver un número cercano a cero se incrementa conforme el algoritmo avanza, es decir, la proporción en la cual un gen es mutado disminuye conforme el algoritmo avanza. Esta propiedad hace que el operador realice una búsqueda uniforme en el espacio inicial cuando t es pequeño, y muy local en las últimas generaciones.

Vamos a ver lo explicado anteriormente con un ejemplo. Supongamos que tenemos el siguiente cromosoma codificado de forma real. El cromosoma codifica una única partición borrosa por simplificar.

C	80.0	**105.0**	130.0

Fig 4.7. Individuo antes de una mutación

Supongamos que queremos mutar el valor 105.0 y los parámetros toman los siguientes valores:

t=generación 2
$[a_i, b_i] = [80,130]$
$r = 1 \in \{0,1\}$

Como $r = 1$

$$c_i' = 105.0 - \Delta(2,105.0 - 80) = 105.0 - \Delta(2,25) \quad \text{(a)}$$

entonces si $\tau = 0.5 \in [0,1]$ y b=0.1 tenemos que,

$$\Delta(2,25) = 25 * \left(1 - 0.5^{\left(1-\frac{1}{20}\right)^{0.1}} \right) = 25 * (1 - 0.5^{0.994}) = 25 * (1 - 0.5) = 12.5$$

Si lo aplicamos a la formula (a)

$$c_i^{'} = 105.0 - \Delta(2,25) = 105.0 - 12.5 = 92.5$$

lo que nos produce el siguiente cromosoma mutado

C'	80.0	**95.5**	130.0

Fig 4.8. Individuo después de una mutación

4.3.5. La función de evaluación

Cada vez que se obtienen una población o bien se produce un cruce entre dos padres, hay que evaluar los individuos de la población de alguna manera. Para nuestro problema hemos utilizado la nueva definición de la medida de Información Mutua propuesta: definición II.

Para cada individuo de la población, que en nuestro caso será una definición de una partición borrosa, se calcula la Información Mutua con estimación boostrap vista en la sección 3.3.3.3 que aporta el conjunto de variables del conjunto de ejemplos con respecto a la clase.

Cuando el algoritmo genético termina, el resultado es una partición optimizada, aquella que nos proporciona un valor de Información Mutua mayor de las variables con respecto a la clase.

4.3.6. Esquema del algoritmo utilizado

PASO1: Generar_Poblacion_Inicial
generacion=0;

MIENTRAS (generacion<POBLACION) ***HACER***

PASO2: Siguiente generacion --> generacion++
PASO3: Seeleccion

MIENTRAS (No operadores genéticos a toda la población) ***HACER***

PASO4: Saco un numero al azar (0,1,2) y reproduzco, cruzo o muto respectivamente
PASO5: ***SI*** (Reproducir) ***ENTONCES***
Seleccionar un individuo al azar hacer una copia
Marcar el individuo
FIN_SI

SI (Cruzar) ***ENTONCES***
Seleccionar dos individuos al azar
Sacar una probabilidad entre 0 y 1.
SI (probabilidad < prob. cruce) ***ENTONCES***
Cruzar y marcar los individuos
FIN_SI
FIN_SI

SI (Mutar) ***ENTONCES***
Seleccionar un individuo al azar
Seleccionar un gen del cromosoma
Sacar una probabilidad entre 0 y 1.
SI (probabilidad < prob. mutación) ***ENTONCES***
Mutar y marcar el individuo
FIN_SI
FIN_SI
Volver al ***PASO4*** si se puede
FIN_MIENTRAS

PASO6: Evaluar la población

FIN_MIENTRAS
Devolver la partición óptima

Fig 4.9. Genético aplicado a la definición I

4.4. Conocimiento de la probabilidad superior de observación: problema con orden parcial

En este apartado mostraremos las características de nuestro algoritmo genético multiobjetivo que servirá para solucionar la problemática de la definición II.

4.4.1. Algoritmo genético NSGA II.

Un problema de optimización multiobjetivo plantea la optimización (minimización o maximización) de un conjunto de funciones, habitualmente en conflicto entre sí. La existencia de múltiples funciones objetivo plantea una diferencia fundamental con un problema monoobjetivo: no existirá una única solución al problema, sino un conjunto de soluciones que plantearán diferentes compromisos entre los valores de las funciones a optimizar.

Los primeros intentos de utilizar múltiples criterios en un algoritmo genético se remontan a finales de los 60, tal y como señala Goldberg [Gol89], cuando estos primeros intentos se centraban fundamentalmente en la agregación de los criterios mediante una escalarización de los mismos. Sin embargo, desde entonces han aparecido en la literatura otros enfoques distintos que se basan en la eficiencia de Pareto y en otros tipos de ordenación, tal y como se puede ver por ejemplo en Fonseca y Fleming [FF93].

Se utilizará un enfoque basado en el orden de Pareto. El valor de adaptación de cada individuo depende no del valor de cada uno de los criterios, sino de su eficiencia o dominación dentro de cada población. Así, Goldberg [Gol89] propuso establecer un ranking de no-dominación para resolver los problemas del enfoque VEGA. La idea es encontrar los individuos en cada generación que no están dominados por ningún otro individuo, asignarles el ranking más alto y extraerlos de la población. Con el resto de individuos se repite este proceso hasta que todos ellos tienen asignado una posición en el ranking según este proceso, realizándose entonces una selección por ranking. Este enfoque se ha mostrado superior al enfoque VEGA en algunos casos, como se puede ver en Hilliard y otros [HLPR89].

Fonseca y Fleming en [FF93] desarrollaron esta idea con el método MOGA (MultiObjective Genetic Algorithm), donde el ranking de cada individuo depende del número de individuos en la población que lo dominan, teniendo los individuos eficientes ranking uno. Por otra parte el

algoritmo NSGA-II (Non-dominated Sorting Genetic Algorithm, versión II) fue presentado por K. Deb y sus colegas del Laboratorio de Algoritmos Genéticos del Instituto Tecnológico Kanpur en India en el año 2000 [DAPM00]. Surgió como una versión mejorada del algoritmo NSGA [SD94], de quién heredó su estructura principal, pero incluyendo características distintivas para resolver tres aspectos fuertemente criticados en la comunidad de investigadores sobre el NSGA: el ordenamiento no dominado, la ausencia de elitismo y la dependencia del parámetro σ para aplicar la técnica de sharing.

Las características principales del algoritmo NSGA-II son las siguientes:

- El ordenamiento no-dominado elitista mediante una técnica de comparación que utiliza una subpoblación auxiliar, que le permite disminuir la complejidad de los chequeos de dominancia de $O(MP^3)$ a $O(MP^2)$, siendo M el número de funciones objetivo y P el tamaño de la población utilizada.

- La utilización de una técnica de ***crowding*** que no requiere especificar parámetros adicionales para la preservación de diversidad en la población, eliminando la dependencia de parámetros como el σ de sharing utilizado por el NSGA original.

- La asignación de valores de fitness en base a los niveles o rangos de no dominancia, se hereda del NSGA-II original, aunque se considera en el procedimiento de asignación los valores de distancia de crowding utilizados para evaluar la diversidad de las soluciones.

Un esquema del algoritmo NSGA-II, basado en la descripción de Deb y otros [DAPM00] es la que se muestra a continuación. Pueden apreciarse los operadores mencionados, utilizados para el ordenamiento no dominado, evaluación de la diversidad mediante la técnica de crowding y asignación de fitness.

Inicializar Población Inicial
generación = 0
Evaluar Población Inicial

MIENTRAS (no CriterioParada) ***HACER***
R = Padres ∪ Hijos
Frentes = ***Sorting No Dominado***(R)
NuevaPop = ϕ
i=1

MIENTRAS |NuevaPop| + |Frentes(i)| ≤ tampob ***HACER***
Calcular Distancia de Crowding (Frentes(i))
NuevaPop = NuevaPop ∪ Frentes(i)
i++
FIN MIENTRAS

Ordenación por Distancia (Frentes(i))
NuevaPop = NuevaPop ∪ Frentes(i)[1:(tampob - |NuevaPop|)
Hijos = ***Selección y Reproducción***(NuevaPop)
generación ++
P(generación) = NuevaPop

FIN MIENTRAS

Devolver la mejor Solución Hallada

Fig 4.10. Genético aplicado a la definición II

4.4.2. Nuestro NSGA II. Definición del sistema genético borroso

Igual que en el algoritmo genético explicado en la sección 3.4.2., hay que definir el esquema de codificación, el sistema de selección y los operadores géticos.

- Esquema de codificación: codificación real.
- Selección de la población intermedia: basada en ranking por dominancia y distancia de crowding
- Operador de cruce: max-min-aritmético [HILV95]
- Operador de mutación: no uniforme propuesto por Michalewicz [Mic96]

No obstante hay ciertas operaciones dentro del algoritmo NSGA II que deberán ser modificadas o adaptadas al caso particular de problemas que manejan datos imprecisos, como se verá en las secciones siguientes.

4.4.3. Esquema evolutivo

En este caso se trata de abordar problemas en los cuales los datos son imprecisos. Problemas en los que la evaluación producida por la nueva definición de Información Mutua propuesta en la definición II con aproximación bootstrap, produce un intervalo, como se ha mostrado en la sección 3.3.3.2. Por este motivo no se puede aplicar el algoritmo genético NSGA-II tal cual sino que hay que adaptar algunas partes del mismo como se verá en la sección 3.4.3.5.

Partiendo de una población inicial, se genera una población intermedia aplicando los operadores genéticos. Ambas poblaciones se unen y se ordenan los individuos de acuerdo con un ranking basado en frentes.

Para cada individuo se crea la lista de individuos a los que domina y se crean los diferentes frentes, de tal forma que el frente 0 contendrá los individuos no dominados por nadie, el frente 1 los individuos dominados por un individuo y así sucesivamente. Una vez ordenados por los frentes de esta manera, se seleccionan los individuos de la nueva población. El problema que surge es que haya que cortar uno de los frentes.

Si tenemos que separar un frente en dos partes, este frente debería estar ordenado de alguna forma para que el corte tenga coherencia .Si el frente tiene 10 individuos y no hay orden alguno y necesito solo 3 de ellos, ¿cuáles selecciono?. Para paliar este problema se utiliza la distancia de crowding, de tal forma que los individuos con mayor distancia dos a dos van los primeros.

En nuestro caso particular que es el manejo de datos imprecisos mediante intervalos surge la problemática de la comparación intervalar. En este caso, dos intervalos que se solapen mucho no se dominan y dos que se solapen poco si. En cualquier caso este problema se tratará de explicar en la sección 4.4.4.

4.4.4. La función de evaluación

Cada vez que se obtienen una población o bien se produce un cruce entre dos padres, hay que evaluar los individuos de la población de alguna manera. Para nuestro problema hemos utilizado la nueva definición de la medida de Información Mutua propuesta: definición II.

Para cada individuo de la población, que en nuestro caso será una definición de una partición borrosa, se calcula la Información Mutua con estimación boostrap vista en la sección 3.3.3.3 que aporta el conjunto de variables del conjunto de ejemplos con respecto a la clase.

Cuando el algoritmo genético termina, el resultado es una partición optimizada, aquella que nos proporciona un valor de Información Mutua mayor de las variables con respecto a la clase.

4.4.5. Adaptación de NSGA II al problema de tratamiento de datos imprecisos

El algoritmo NSGA-II se puede extender para el problema del tratamiento de datos imprecisos para que pueda encontrar un conjunto de soluciones no dominadas para un problema en el que la función objetivo es un intervalo.

Hay solo tres módulos en el algoritmo NSGA-II, donde la función fitness se supone que es un vector de números discretos: el operador de precedencia (dominancia), la ordenación no dominada de los individuos y la distancia de crowding. Nuestra extensión consistirá en definiciones alternativas para esos módulos.

El esquema del algoritmo NSGA II que se muestra a continuación, resalta aquellas partes del algoritmo sobre las que se han tenido que realizar las adaptaciones pertinentes.

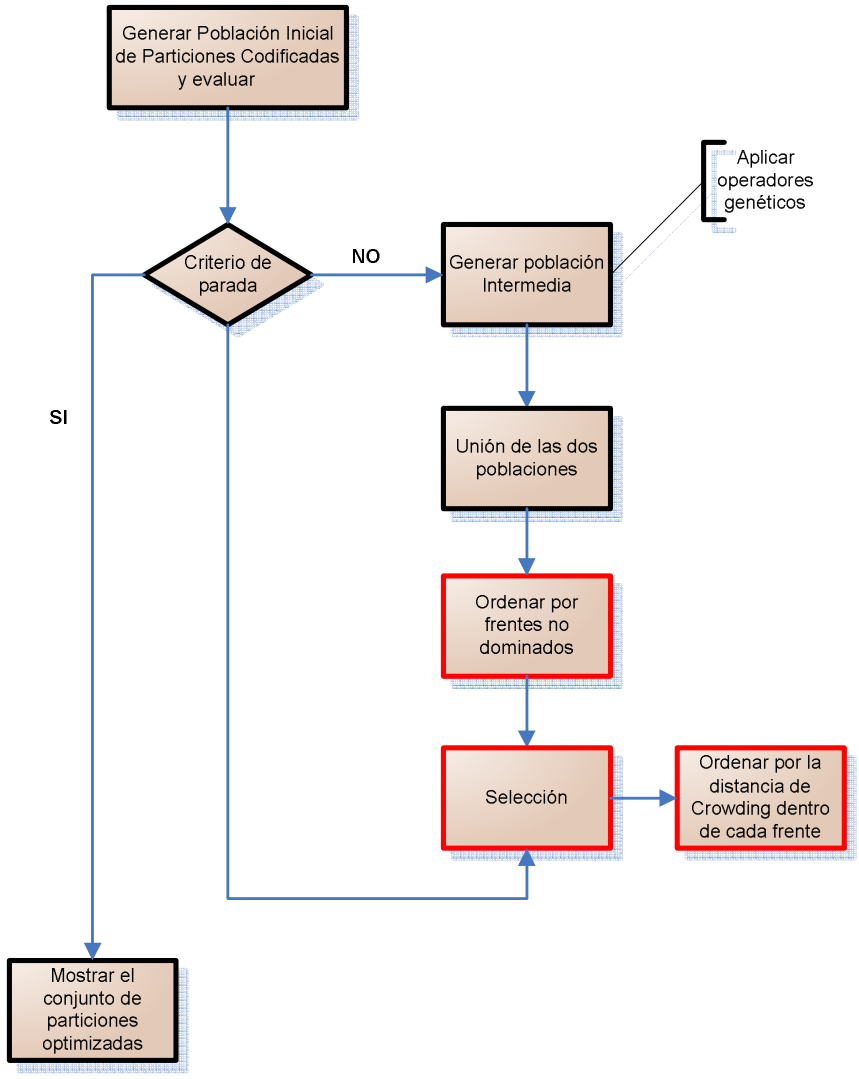


Fig 4.11. Algoritmo genético para datos imprecisos

4.4.5.1. Operador de precedencia

Un sistema borroso depende de un vector de x parámetros que tiene un fitness que denotaremos por θ. θ es desconocida excepto por la distribución de posibilidad dada por el fitness borroso $\tilde{F}_x$, en nuestro caso un intervalo.

Para determinar que individuo precede a otro, es necesario lanzar un procedimiento tal que dadas dos observaciones imprecisas $\widetilde{F}_{x1}$ y $\widetilde{F}_{x2}$ de dos valores de fitness θ_1 y θ_2, estime que la probabilidad de que $\theta_1 < \theta_2$ sea mayor que $\theta_1 \geq \theta_2$, esto es $\widetilde{F}_{x1} \prec \widetilde{F}_{x2}$. El criterio que proponemos se pude considerar como un caso especial de ranking borroso. Sin embrago lo que se pretende es encontrar aquellos casos en los que no sea tan evidente seleccionar un individuo u otro en base a los valores obtenidos.

Si se conociese la probabilidad conjunta $P((\theta_1, \theta_2) \mid (x_1, x_2))$ de dos individuos, esto se convertiría en un problema de decisión estadística pero desafortunadamente, la información que nos proporciona un fitness impreciso es mucho menor que la que nos proporciona una distribución de probabilidad. Por ejemplo podríamos utilizar,

$$\frac{P\{(\theta_1, \theta_2) : \theta_1 < \theta_2\}}{P\{(\theta_1, \theta_2) : \theta_1 \geq \theta_2\}} <> 1$$

Como ya dijimos, el fitness impreciso proporciona mucha menos información que una distribución de probabilidad, ya que este es tan solo una probabilidad superior, que domina la probabilidad posterior del fitness discreto.

En otras palabras, dado un individuo *x*, la información que tenemos sobre su fitness tiene un valor de θ limitado a,

$$\widetilde{F}_x(\theta) = P^*(\theta \mid x) \geq P(\theta \mid x)$$

por lo que la regla de decisión se convierte en

$$\frac{P_*(\{(\theta_1, \theta_2) : \theta_1 < \theta_2\} \mid x)}{P^*(\{(\theta_1, \theta_2) : \theta_1 \geq \theta_2\} \mid x)} > 1 \quad \textit{(1)}$$

También sabemos que $P^*(. \mid x)$ es una posibilidad para todo *x*, por lo que podemos decir que,

$$P^*(A \mid x) = \max\{P^*(\theta \mid x) : \theta \in A\}$$

Como $P^*(A) = 1 - P_*(A^c)$, la expresión (1) se reduce a

$$P^*(\{(\theta_1, \theta_2) : \theta_1 \geq \theta_2\} \mid x) < 1/2 \quad \textit{(2)}$$

por lo tanto para decidir cuando $\widetilde{F}_{x1} \prec \widetilde{F}_{x2}$ deberemos comprobar que,

$$\max\{P^*(\theta \mid x_1).P^*(\theta \mid x_2) : \theta_1 < \theta_2\} \geq 1/2$$
$$\max\{P^*(\theta \mid x_1).P^*(\theta \mid x_2) : \theta_1 \geq \theta_2\} < 1/2$$

con $P^*(\theta \mid x_1) = \widetilde{F}_{x1}$ y $P^*(\theta \mid x_2) = \widetilde{F}_{x2}$.

Hay que recalcar que, en este último caso, rechazar que $\theta_1 < \theta_2$ no implica que $\theta_1 \geq \theta_2$, porque puede ocurrir que la ecuación (2) dé un resultado mayor que $1/2$ y también que $P^*(\{(\theta_1, \theta_2) : \theta_1 < \theta_2\} \mid x) \geq 1/2$, y entonces concluiremos que la función de pertenencia borrosa $\widetilde{F}_{x1}$ y $\widetilde{F}_{x2}$ no contiene la suficiente información para apreciar diferencia significativas entre ellas. En particular, ocurrirá cuando $\widetilde{F}_{x1}$ y $\widetilde{F}_{x2}$ son intervalos no disjuntos. Hay que recalcar que la aplicación de la ecuación (1) para calcular el fitness intervalar, es numéricamente lo mismo que la llamada *dominancia fuerte*, propuesta en [Lim05].

Por otra parte la imposibilidad de distinguir entre intervalos con intersección no vacía es un problema importante. Nosotros vamos a mejorar esta problemática introduciendo un conocimiento a priori sobre la distribución de probabilidad del fitness. Admitiremos que el perfil de regla probable [WM99], esto es, que la probabilidad normalizada es

$$L(\theta) = \frac{P(x \mid \theta)}{\max\{P(x \mid \theta)\}} = P^*(\theta \mid x)$$

y también una distribución uniforme a priori, aplicando la regla de Bayes obtenemos que

$$P(A \mid x) = \frac{\sum_{\theta \in A} \widetilde{F}_x(\theta)}{\sum_{\theta \in \Theta} \widetilde{F}(\theta)}$$

que en realidad produce la regla de decisión

$$\frac{\sum_{\theta_1<\theta_2} \widetilde{F}_{x1}(\theta_1)\widetilde{F}_{x2}(\theta_2)}{\sum_{\theta_1,\theta_2\in\Theta_2} \widetilde{F}_{x1}(\theta_1)\widetilde{F}_{x2}(\theta_2)} > 1$$

El uso de una probabilidad a priori no es compatible con la interpretación de posibilidad. Como hemos visto en párrafos anteriores, hemos recuperado la probabilidad a posteriori a partir de un límite superior de si misma y una probabilidad a priori. Es lo mismo que normalizar la salida borrosa del modelo y asumir que las funciones de pertenencia son probabilidades.

Por el contrario, la combinación de un límite superior de una probabilidad a posteriori y una imprecisión a priori es significativa. Siguiendo con el enfoque de [Wal91], podemos suponer que la distribución a priori es una cierta familia P de probabilidades, e interpreta que $\widetilde{F}$ es una envoltura superior del conjunto de distribuciones de probabilidad posteriores que se alcanza cuando la función de probabilidad se combina con cada probabilidad a priori en la familia:

$$P^*(A \mid x) = \sup_{p\in P} \frac{\sum_{\theta\in A} L(\theta)P(\theta)}{\sum_{\theta\in\Theta} L(\theta)P(\theta)} \quad \textit{(3)}$$

y $\widetilde{F}_{x1} \prec \widetilde{F}_{x2}$ cuando

$$P_*(\{(\theta_1,\theta_2) : \theta_1 < \theta_2\} \mid x) > 1/2$$

La medida de la ecuación (3) es una verosimilitud. Nosotros solamente obtenemos una posibilidad en determinados casos particulares. Según [Wal99], la distribución de probabilidad con mas información que contienen a (3) y verificando algunas condiciones de regularidad es

$$\pi_0(\theta) = \frac{\sum_{\varphi: L(\varphi)\le L(\theta)} L(\varphi)}{\sum_{\varphi\in\Theta} L(\varphi)}$$

Podemos hacer $\widetilde{F}(\theta) = \pi_0(\theta)$ y resolverlo para *L*, y entonces sustituir *L* y la anterior en la ecuación (3). En el caso de que utilicemos el ECMB es inmediato, porque cuando $\widetilde{F}_x$ es un intervalo, entonces $L(\theta) = \pi_0(\theta) = \widetilde{F}_x(\theta)$.

El conjunto de conocimientos a priori puede abarca desde la ausencia de información $P^*(\theta) = 1$ y $P_*(\theta) = 0$ hasta la probabilidad a priori ya explicada con anterioridad. Nosotros usaremos una familia de probabilidades a priori que será lo bastante restrictiva como par asignar precedencias a los intervalos con intersección no vacía, pero no tan restrictiva como la probabilidad a priori uniforme.

4.4.5.2. Ordenación no dominada de los individuos

Ordenar la población con respecto a un objetivo es lo mismo que encontrar el mejor individuo para ese criterio. Una vez encontrado dicho individuo, podemos borrarlo y de forma recursiva repetir la búsqueda para obtener una población ordenada.

Encontrara el mejor individuo es un procedimiento de decisión estadístico que generaliza las reglas mostradas en la sección anterior. Nosotros podemos asegurar la probabilidad inferior del aserto "el i-esimo individuo tiene el mejor fitness de la población" mediante

$$M_i = P_*(\theta_i \text{ es el mínimo}) = \prod_{j=1}^{s} P_*(\theta_i < \theta_j) \quad i \neq j$$

Nosotros proponemos suponer que el mejor individuo es aquel que minimiza M_i. Para ordenar la población, el mejor individuo se borra y se vuelven a calcular el resto los valores de M_i. Entonces se obtiene el siguiente elemento como el nuevo mínimo de M_i. El proceso se repite hasta que todos los individuos son extraídos.

4.4.5.3. Distancia de Crowding

La distancia de crowding se utiliza para un gestionar un frente uniforme, haciendo que los individuos de las áreas mas densas sean menos candidatos a ser seleccionados. En el caso discreto, si los ***s*** individuos de la

población son ordenados tal que $\theta_i < \theta_{i+1}$, la densidad local del i-esimo individuo es aproximadamente

$$p_i = \frac{3}{s.(\theta_{i+1} - \theta_{i-1})}$$

porque el número de puntos en el volumen $[\theta_{i-1}, \theta_{i+1}]$ es tres. En otras palabras, la distancia de crowding es inversamente proporcional a la densidad de individuos en el espacio de fitness, basándonos en un criterio de dos vecinos

$$d_i = \frac{3}{s.p_i}$$

donde p_i es la densidad local del i-esimo individuo.

Para extender esta definición al caso intervalar, supongamos que los s individuos tienen un fitness $\theta_i \in I_i$. La densidad local esta acotada por

$$p_i \in \left[\frac{3}{s.V_i^{\max}}, \frac{3}{s.V_i^{\min}} \right]$$

donde $V_i^{\max}$ es el intervalo mas pequeño que contiene el fitness de I_i y otros dos individuos

$$I_i \subseteq V_i^{\max}, \qquad \#\{j : I_j \subseteq V_i^{\max}\} = 3$$

y $V_i^{\min}$ es el individuo mas pequeño que tiene una intersección no vacía con el fitness de los tres individuos, siendo I_1 uno de ellos

$$I_i \cap V_i^{\min} \neq 0, \qquad \#\{j : I_j \cap V_i^{\min} \neq 0\} = 3$$

Por lo tanto, la distancia de crowding asociada al i-esimo individuo es

$$d_i \in [\| V_i^{\min}, V_i^{\max} \|]$$

Desafortunadamente, esta generalización no produce buenos resultados, porque el límite superior depende demasiado de la incertidumbre del fitness que se va a comparar.

Para resolver este problema introducimos una métrica o medida entre valores imprecisos del fitness. Es deseable que la distancia de crowding entre dos intervalos $x \pm \varepsilon$ e $y \pm \varepsilon$ este dentro de los límites mencionados que no dependen de ε y es compatible con la definición para el caso discreto,

$$d(x \pm \varepsilon, y \pm \varepsilon) = | x - y |$$

Hay muchas métricas que satisfacen estas propiedades, que se puede definir. Se ha seleccionado la distancia de Haausdorff Dados dos conjuntos A y B, la distancia entre ellos se define como

$$d_H(A, B) = \max_{a \in A} \min_{b \in B}(a, b)$$

de tal forma que si A=[a_1,a_2] y B=[b_1,b_2] son intervalos,

$$d_H(A, B) = \max\{| a_1 - b_1 |, | a_2 - b_2 |\}$$

4.4.6. Conclusiones finales

Resumiendo un poco lo visto en la sección 4.4., hay que decir que por una parte se ha realizado un estudio de funcionamiento del algoritmo NSGA-II, así como del esquema genético del mismo, y por otra parte e ha realizado una adaptación algorítmica del NSGA-II para el caso de conjuntos de datos con imprecisión. Se ha mostrado como adaptar cada una de las partes necesarias para conseguir el objetivo deseado,

- La comparación entre valores de Información Mutua
- La distancia entre valores de Información Mutua
- La ordenación de valores no dominados.

En el capítulo 5 se mostrará el análisis numérico llevado a cabo, tanto desde el punto de vista de tablas con resultados como desde el punto de vista gráfico.

CAPITULO 5

Experimentación

5.1 Introducción

En esté capítulo se mostrará el conjunto de experimentos llevados a cabo para demostrar que hay una mejora significativa si se utiliza la definición borrosa de Información Mutua propuesta, enfocada a la optimización de particiones borrosas en el diseño de clasificadores borrosos.

Se experimentará tanto con datos con información precisa como con datos con información imprecisa o con perdida de información, para lo que demostraremos que esta especialmente indicada.

Tanto para datos imprecitos como para datos precisos, se siguen dos procesos: primeramente se introduce la relación de datasets utilizados en la experimentación, así como la relación de algoritmos de clasificación utilizados como herramienta para demostrar resultados y en secciones siguientes de muestra la experimentación realizada, así como las conclusiones de la misma. A continuación se muestra un esquema representativo.

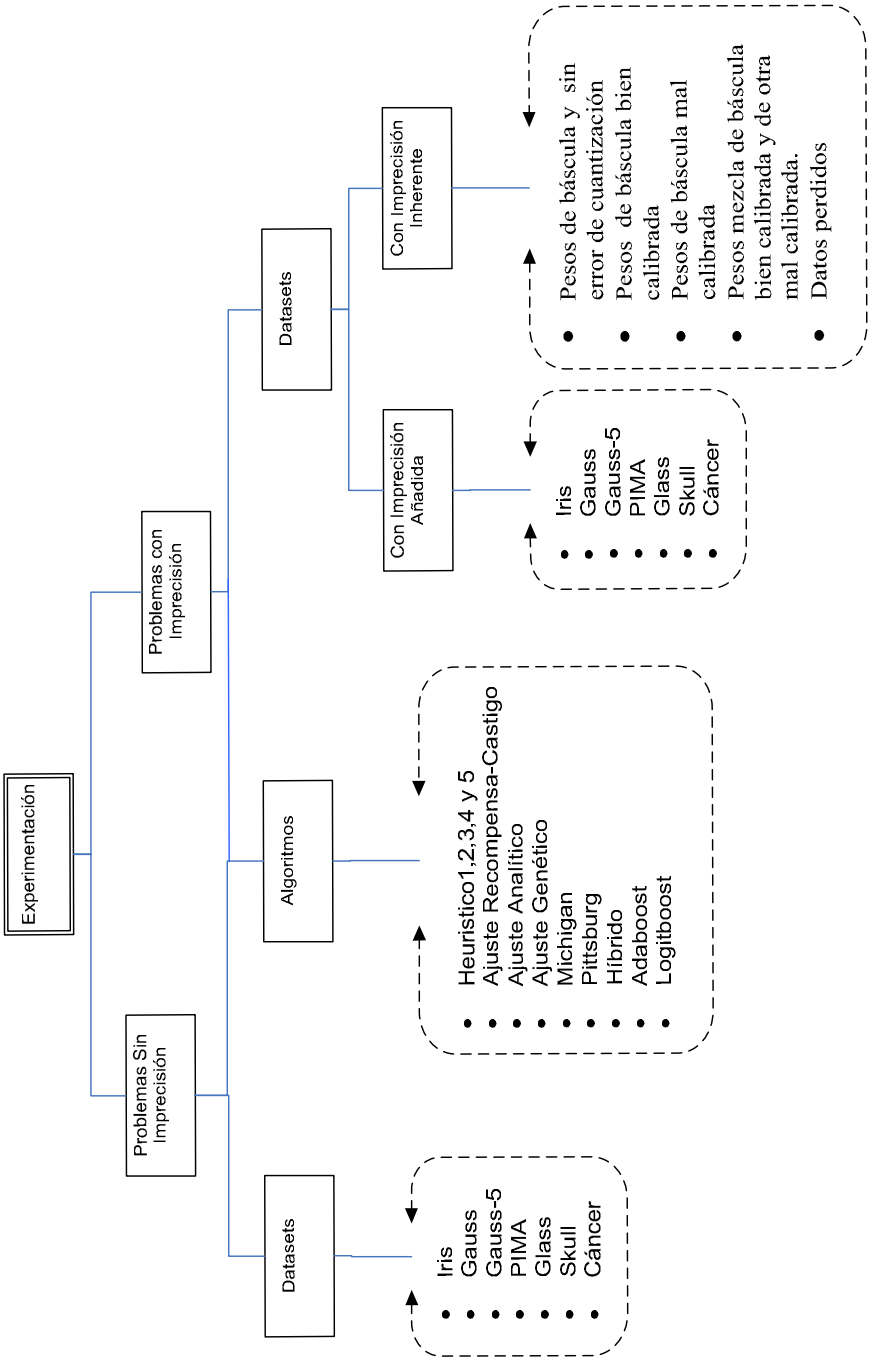


Fig 5.1. Esquema de la experimentación

5.2. Tipos de problemas

Los problemas utilizados para analizar el funcionamiento de los métodos propuestos, van a ser de dos tipos:

a) Problemas que no introducen imprecisión. Los conjuntos de datos utilizados serán datos continuos con información precisa en cada ejemplo. Conjuntos como PIMA o Iris son ejemplos de este tipo de problemas.

b) Problemas con imprecisión para los cuales consideramos nuestro método como el idóneo para obtener resultados y comparativas. Se utilizará alguno de los conjuntos de datos utilizados para datos precisos añadiéndole cierta imprecisión en algunos valores y además se proponen nuevos conjuntos de datos imprecisos e incluso con valores ausentes.

5.3. Datasets utilizados en problemas sin imprecisión

Dentro de ese primer grupo de problemas, vamos a trabajar con los conjuntos de ejemplos iris, gauss, gauss-5, PIMA, glass, skull y cáncer, utilizados con frecuencia en el análisis de este tipo de situaciones, y que pasaremos a describir en las subsecciones siguientes.

5.3.1. Iris

Iris es una base de datos de clasificación de plantas. Fue utilizada por primera vez por [Fis36] y propone la clasificación de tres clases de plantas, *setosa, virgínica* y *versicolor*, a través de cuatro variables predicativas cuyo rango se describe en la Tabla 5.1.

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

Tabla 5.1. Descripción de los atributos de IRIS

El conjunto de datos está formado por 150 muestras, 50 de cada una de las tres clases.

5.3.2. Gauss

Gauss [Hay99] es un conjunto de 4000 puntos tomados a partir de dos distribuciones Gaussianas bidimensionales solapadas y centradas en (0,0) y (2,0) con distintas matrices de covarianza. Cada uno de los ejemplos se describe, por lo tanto, a través de 2 variables cuyo rango se describe en la Tabla 5.2.

Variable	Descripción	Rango
1	Dimensión 1	[-5.326654, 9.144776]
2	Dimensión 2	[-7.51934, 7.347427]

Tabla 5.2. Descripción de los atributos de GAUSS

Los datos se reparten en dos clases 0 y 1 que se corresponden con cada una de las dos distribuciones. De los 4000 puntos, la mitad son de una distribución y la otra mitad de otra.

5.3.3. Gauss-5

Gauss-5 es un problema sintético de 750 puntos a partir de 5 distribuciones Gaussianas bidimensionales centradas en (0, 0), (−1,−1), (−1, 1), (1,−1), (1, 1). Cada uno de los ejemplos se describe, por lo tanto, través de 2 variables cuyo rango se describe en la Tabla 5.3.

Variable	Descripción	Rango
1	Dimensión 1	[-3.74877073, 4.38080242]
2	Dimensión 2	[-3.78346769, 3.90358026]

Tabla 5.3. Descripción de los atributos de GAUSS-5

Los datos se reparten en cinco clases de 0 a 4 que se corresponden con cada una de las cinco distribuciones. De los 750 puntos, 50 son de la distribución 1, 100 son de la distribución 2, 150 son de la distribución 3, 200 son de la distribución 4 y 250 son de la distribución 5.

5.3.4. PIMA

PIMA (Pima Indians Diabetes Database) es un conjunto de 768 muestras de un problema de diagnóstico en el que, a partir de un conjunto de valores de variables propuestas por la *Organización Mundial de la Salud*, se determina si el paciente sufre o no diabetes. Los datos fueron recogidos por investigadores del *Nacional Institute of Diabetes and Digestive* de EE.UU. en una población de mujeres indias de la tribu pima. Cada uno de los ejemplos se describe a través de ocho variables cuyo rango se describe en la Tabla 5.4.

Variable	Descripción	Rango
1	Número de embarazos	[0, 17]
2	Concentración de glucosa en plasma	[0, 199]
3	Presión diastólica de la sangre	[0, 122]
4	Grosor de los dobleces de la piel en los triceps	[0, 99]
5	Nivel de insulina	[0, 846]
6	Índice de masa corporal	[0, 67.1]
7	Estimación de la influencia genética	[0.08, 2.42]
8	Edad	[21, 81]

Tabla 5.4. Descripción de los atributos de PIMA

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

5.3.5. Glass

Glass es un conjunto de 214 muestras que consiste en clasificar diversos tipos del cristal. Está motivada por investigaciones criminológicas. Cada uno de los ejemplos se describe a través de ocho variables cuyo rango se describe en la Tabla 5.5.

Variable	Descripción	Rango
1	RI	[1.51115, 1.53393]
2	Na	[10.73, 17.38]
3	Mg	[0.0, 4.49]
4	Al	[0.29, 3.5]
5	Si	[69.81, 75.41]
6	K	[0.0, 6.21]
7	Ca	[5.43, 16.19]
8	Ba	[0.0, 3.15]
9	Fe	[0.0, 0.51]

Tabla 5.5. Descripción de los atributos de GLASS

Los datos se reparten en 6 clases, del 0 al 5, dependiendo del tipo de cristal encontrado. De las 214 muestras, 70 de de la clase 0, 76 son de la clase 1, 17 son de la clase 2, 13 son de la case 3, 9 son de la clase 4 y 29 son de la clase 5.

5.3.6. Skulls

Skulls es un conjunto de 150 muestras que consiste en la clasificación de cráneos. Cada uno de los ejemplos se describe mediante cuatro variables cuyo rango se describe en la Tabla 5.6.

Variable	Descripción	Rango
1	Variable1	[119, 148]
2	Variable2	[120, 145]
3	Variable3	[81, 114]
4	Variable4	[44, 60]

Tabla 5.6. Descripción de los atributos de SKULLS

Los datos se reparten en cinco clases que van de 0 a 4, dependiendo del tipo de cráneo. Las 150 muestras se reparten en 30 para cada una de las clases.

5.3.7. Cáncer

Cáncer (Wisconsin Breast Cancer) es un conjunto de 699 muestras que consiste la clasificación de un cáncer en maligno o benigno. Cada uno de los ejemplos se describe mediante nueve variables cuyo rango se describe en la Tabla 5.7.

Variable	Descripción	Rango
1	Variable1	[0.1, 1]
2	Variable2	[0.1, 1]
3	Variable3	[0.1, 1]
4	Variable4	[0.1, 1]
5	Variable5	[0.1, 1]
6	Variable6	[0.1, 1]
7	Variable7	[0.1, 1]
8	Variable8	[0.1, 1]
9	Variable9	[0.1, 1]

Tabla 5.7. Descripción de los atributos de CANCER

Los datos se reparten en dos clases, 0 y 1, dependiendo del tipo de cáncer diagnosticado. De las 699 muestras 458 son de clase 0 y 242 son de clase 1.

5.4. Algoritmos utilizados en la experimentación

Del resultado de la aplicación de nuestro método, y a partir de una partición uniforme inicial de los datos, pasando por un algoritmo de optimización, se obtiene la partición optimizada de la partición original. Para demostrar que la partición así obtenida es tan eficiente o mas que la inicial utilizaremos distintos algoritmos de clasificación.

Como se ha mencionado en el capítulo 1 vamos a utilizar sistemas de clasificación encontrados en la literatura para mostrar nuestros experimentos. A continuación se muestra todos ellos junto con sus abreviaturas:

- ***Heurístico 1 → HEU1***: En todas las reglas asociadas a celdas con soporte no nulo se emite una regla. El consecuente de la regla es la clase que maximiza la confianza. El peso de la regla es siempre 0.5.

- ***Heurístico 2 → HEU2***: El mismo algoritmo que el heurístico anterior, pero el peso de la regla es la confianza del consecuente.

- ***Heurístico 3 → HEU3***: El mismo algoritmo, con peso igual a la diferencia entre la confianza de la clase más frecuente y la media de las confianzas de las clases restantes.

- ***Heurístico 4 → HEU4***: El mismo algoritmo, con peso igual a la diferencia entre la confianza de la clase más frecuente y la confianza de la siguiente clase más frecuente.

- ***Heurístico 5 → HEU5***: El mismo algoritmo, con peso igual a la diferencia entre la confianza de la clase más frecuente y la suma de las confianzas de las clases restantes.

- ***Ajuste por recompensa-castigo → REWP***: Partiendo de la solución dada por el algoritmo heurístico de tipo 1, se ajustan iterativamente los pesos de todas las reglas mediante el algoritmo de recompensa-castigo: para cada ejemplo, el peso de la regla ganadora se aumenta si la clasificación es correcta y se disminuye en caso contrario. El proceso continúa, con modificaciones gradualmente menores, hasta que se alcanza un equilibrio. A continuación se eliminan de la base de conocimiento las reglas que no hayan ganado en ningún ejemplo del conjunto de entrenamiento.

- ***Ajuste analítico → ANAL***: Partiendo de la solución dada por el algoritmo heurístico de tipo 1, se ajustan iterativamente los pesos de todas las reglas mediante el algoritmo de aprendizaje analítico. Para cada ejemplo, si la regla ganadora es incorrecta, se prueba a rebajar su peso y si ha mejorado, se conserva esta modificación. En caso contrario, se prueba a aumentar el peso de la regla más prometedora (la regla de mayor importancia cuyo consecuente sea compatible con el ejemplo estudiado) y se admite la modificación sólo en el caso de que el error de entrenamiento disminuya. El proceso se repite hasta que no haya cambios en una iteración. Por último, se eliminan todas las reglas que no hayan ganado en ninguno de los ejemplos.

- ***Ajuste genético → GENS***: Se elige el subconjunto de reglas que minimiza el error de entrenamiento mediante un algoritmo genético con codificación binaria.

- ***Algoritmo Michigan → MICH***: Un algoritmo genético estilo Michigan, donde cada individuo es una regla y la población es la base de conocimiento. En todos los ejemplos se ha usado una población de tamaño 25 y se ha dejado evolucionar el sistema durante 1000 generaciones.

- ***Algoritmo Pittsburg → PITT***: Un algoritmo genético tipo Pittsburgh, donde cada individuo es una base de conocimiento completa. Población de tamaño 50, 25 reglas por individuo y 50 generaciones.

- ***Algoritmo Hibrido → HYBR***: Algoritmo híbrido de tipo Pittsburgh con mutación Michigan: con alta probabilidad, se aplica a los descendientes del cruce una operación de mutación que se corresponde con un cruce Michigan aplicado a un único individuo.

- ***Adaboost → ADAB***: Generación iterativa de 25 reglas borrosas mediante el algoritmo fuzzy Adaboost. A diferencia de los algoritmos anteriores, se emplea inferencia por suma de votos.

- ***Logitboost → LOGI***: Generación iterativa de 10 reglas de tipo 3 mediante el algoritmo fuzzy Logitboost. También se utiliza inferencia por suma de votos.

Intuitivamente, el efecto de la optimización de las funciones de pertenencia debe ser más marcado en los algoritmos que emplean inferencia con un único ganador, dado que los algoritmos estadísticos (Adaboost y Logitboost) pueden combinar más fácilmente el efecto de varias reglas y crear regiones de decisión más complejas, a costa de la interpretabilidad del resultado.

Los algoritmos heurísticos generan bases de conocimiento de un tamaño muy elevado en los problemas con un alto número de entradas (cancer, glass, pima) lo que los hace poco interesantes desde un punto de vista práctico. Los algoritmos genéticos de aprendizaje han sido limitados a 25 reglas en todos los problemas, por lo cual sus errores pueden ser llamativamente altos en comparación con los algoritmos heurísticos. Por último, se ha decidido limitar, como solución de compromiso, el algoritmo Logitboost a 10 reglas de tipo III (con tantos pesos como clases en cada consecuente). En general, cualquiera de los algoritmos genéticos podría ser optimizado en mayor medida para cada problema, pero en este estudio estamos interesados en el efecto del ajuste de las particiones en el error y no en comparar la eficacia de los algoritmos de aprendizaje.

Por último, destacar que se incluirán los errores tanto sobre el conjunto de entrenamiento como sobre el conjunto de prueba, para detectar si la causa en el descenso de la eficiencia de ciertos algoritmos de aprendizaje se debe a un sobreentrenamiento y no es atribuible, por tanto, a la optimización genética de las particiones.

5.5. Experimentos con datos completos, sin imprecisión

Los problemas que se plantean constan de un conjunto de ejemplos, cada uno de los cuales esta formado por atributos, que determinan la clase a la que pertenece el ejemplo. Cada uno de esos atributos viene definido por un rango de tipo numérico, dentro del cual se ubican los diferentes valores que puede tomar los atributos.

Para poder trabajar con los datos, un primer paso es el de la discretización de los mismos. A cada valor numérico se le asigna una etiqueta lingüística que representa un conjunto borroso. Dichos conjuntos borrosos pueden venir determinados por un experto o bien pueden ser calculados de alguna forma manual o automática. En el caso de que el conjunto borroso venga determinado por un experto, no hay nada que decir a no ser que se utilice algún método de ajuste de particiones que demuestre y justifique que el ajuste realizado es mejor que lo que decía el experto. En el caso de que las particiones se realicen de forma manual, será nuestro criterio el que decida que partición es la mejor. Podemos realizar diferentes tipos de particiones distintas en función del número de etiquetas lingüísticas, de la distribución de los datos,…

Sabemos por lo contado en el capitulo 1 que el número de etiquetas lingüísticas viene dado por el problema que se este tratando. No obstante el número mínimo de etiquetas lingüísticas utilizado es de 3. En un primer momento pensaremos en realizar una distribución uniforme de los datos. A la partición uniforme inicial se le aplicará un proceso genético que utilizará la definición II de Información Mutua con estimación bootstrap para llegar a obtener una partición optimizada a partir de la original.

El conjunto de ejemplos o dataset se particionará en ejemplos de entrenamiento y ejemplos de prueba realizando 10 conjuntos distintos tanto para entrenamiento como para prueba. Los experimentos se realizarán tanto sobre el conjunto de entrenamiento como sobre el conjunto de prueba para cada uno de los 10 conjuntos distintos y para cada algoritmo enumerado en la sección 5.4. Para cada dataset y por cada algoritmo se calcula el error medio de clasificación, que es lo que se representa en las tablas y

gráficamente en los boxplot. Se mostrará tanto el error medio de clasificación para la partición uniforme como para la optimizada, tanto en entrenamiento como en prueba.

Desde el punto de vista del algoritmo genético se muestran los valores de los distintos parámetros utilizados en la experimentación.

- Numero de etiquetas lingüísticas por variable: 3
- Número de caminos usados en aproximación Monte Carlo: 1000
- Umbral de pertenencia : 0
- Probabilidad de mutación: 0.05
- Probabilidad de cruce: 0.05
- Numero máximo de generaciones: 50

5.5.1. Dataset Iris

Como puede comprobarse en los boxplots, en el problema Iris ninguno de los algoritmos de aprendizaje ha visto mejorada de forma significativa su capacidad de acertar en entrenamiento. Hay mejoras en el error medio de entrenamiento en 4 de los 13 algoritmos, y mejoras en prueba en 8 de los 13 algoritmos. Asimismo, es destacable que las diferencias en el error de prueba de los 13 algoritmos de aprendizaje son mucho menos acusadas tras realizar el ajuste genético.

	Uniforme		Optimizada	
	Entrenamiento	**Prueba**	**Entrenamiento**	**Prueba**
HEU1	**0.027**	**0.027**	0.040	0.040
HEU2	**0.029**	**0.033**	0.040	0.040
HEU3	0.061	0.060	**0.040**	**0.040**
HEU4	0.067	0.067	**0.040**	**0.040**
HEU5	0.067	0.067	**0.040**	**0.040**
REWP	**0.016**	0.047	0.040	**0.040**
ANAL	**0.027**	**0.033**	0.040	0.040
GENS	0.053	0.067	**0.033**	**0.060**
MICH	**0.037**	0.047	0.040	**0.040**
PITT	**0.033**	0.060	0.039	**0.047**
HYBR	**0.033**	**0.047**	0.050	0.060
ADAB	**0.019**	0.047	0.028	**0.040**
LOGI	**0.026**	**0.040**	0.031	0.047

Tabla 5.8. Experimentos con el dataset IRIS con partición uniforme y optimizada

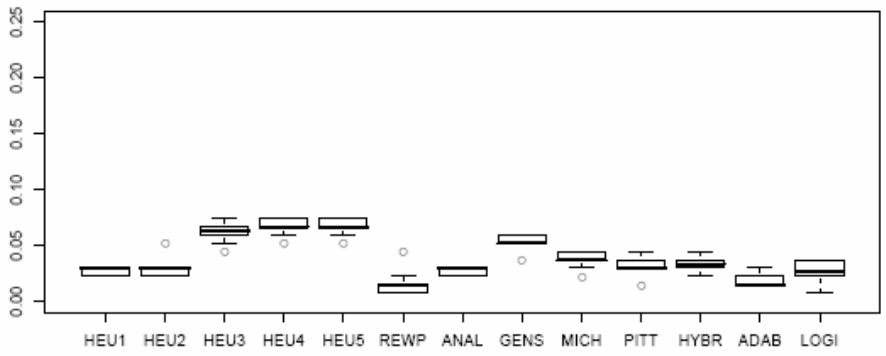


Fig 5.2. Errores en entrenamiento con partición uniforme

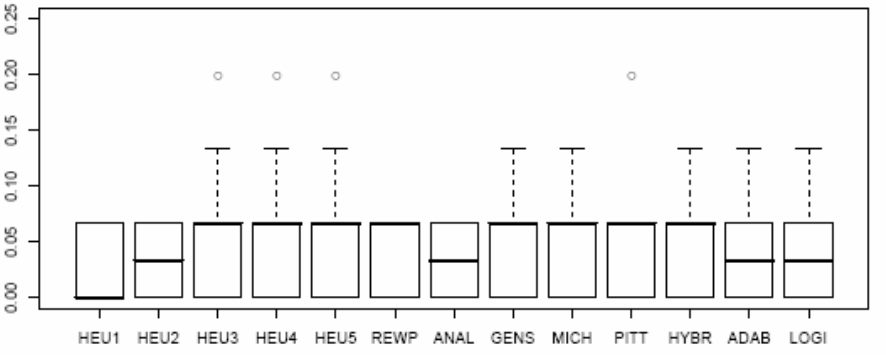


Fig 5.3. Errores en prueba con partición uniforme

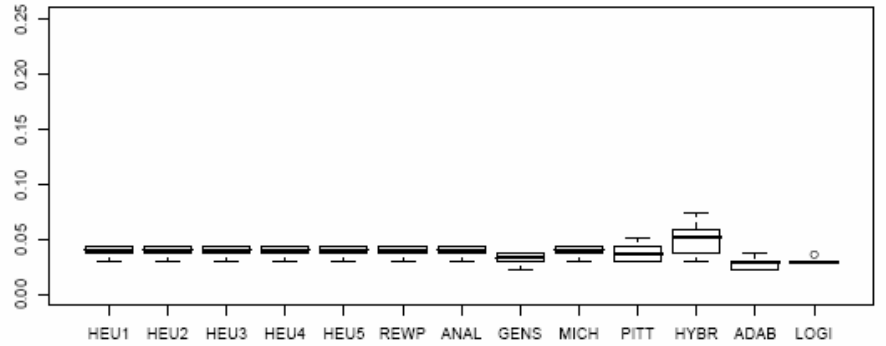


Fig 5.4. Errores en entrenamiento con partición optimizada

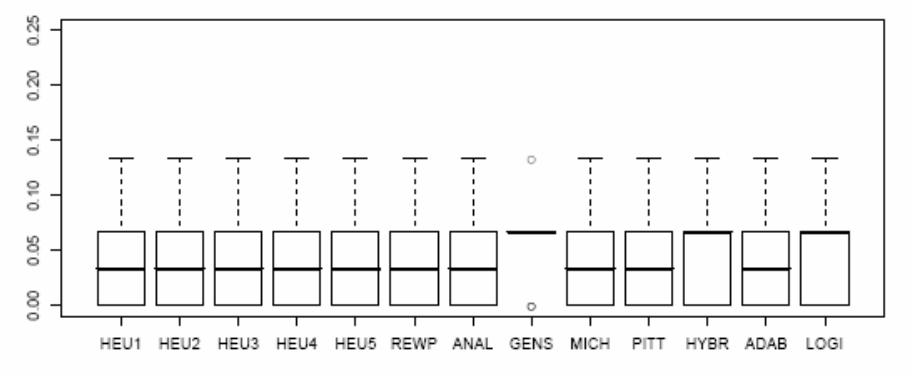


Fig 5.5. Errores en prueba con partición optimizada

5.5.2. Dataset Gauss

El dataset Gauss es un problema de dos clases, con una alta densidad de ejemplos. No es de esperar que la elección de las particiones sea relevante en los clasificadores con inferencia por suma de votos. Las particiones con Información Mutua maximizada sí que conducen a un incremento notable de potencia en los clasificadores basados en inferencia por máximo ganador. El ajuste de pertenencias ha mejorado el error, tanto en entrenamiento como en prueba, de 9 de los 13 clasificadores.

	Uniforme		Optimizada	
	Entrenamiento	**Prueba**	**Entrenamiento**	**Prueba**
HEU1	0.45	0.45	**0.22**	**0.22**
HEU2	0.43	0.43	**0.22**	**0.22**
HEU3	0.26	0.27	**0.22**	**0.22**
HEU4	0.26	0.27	**0.22**	**0.22**
HEU5	0.26	0.27	**0.22**	**0.22**
REWP	0.30	0.30	**0.22**	**0.22**
ANAL	**0.20**	**0.20**	0.22	0.22
GENS	**0.21**	**0.21**	0.22	0.22
MICH	0.31	0.31	**0.22**	**0.22**
PITT	0.31	0.31	**0.22**	**0.22**
HYBR	0.27	0.27	**0.22**	**0.22**
ADAB	**0.21**	**0.21**	0.50	0.50
LOGI	**0.20**	**0.20**	0.22	0.22

Tabla 5.9. Experimentos con el dataset GAUSS con partición uniforme y optimizada

Se observa un empeoramiento muy significativo en el caso de Adaboost pero en general se mantiene un error bastante uniforme entre todos los métodos mejorando en la mayoría de ellos.

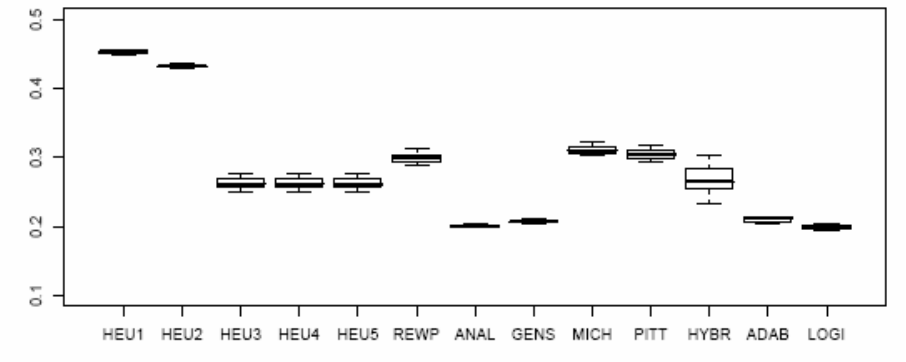


Fig 5.6. Errores en entrenamiento con partición uniforme

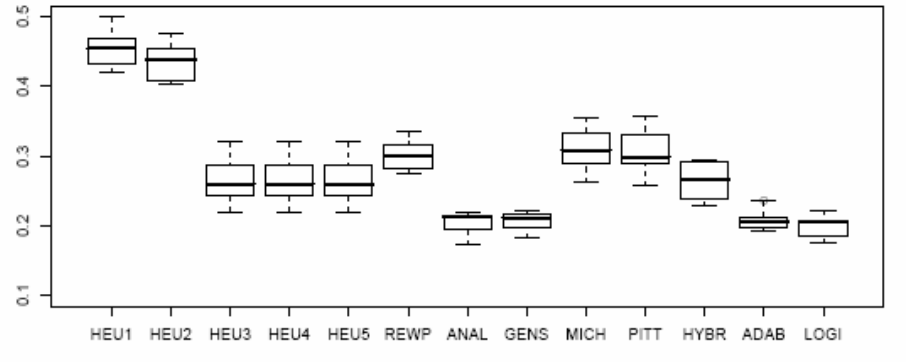


Fig 5.7. Errores prueba con partición uniforme

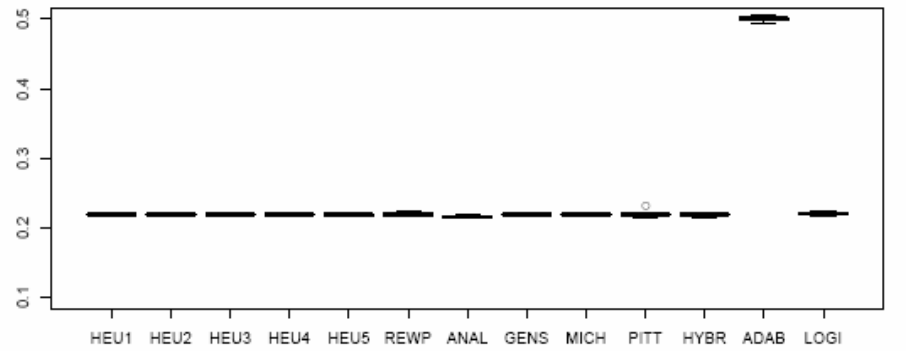


Fig 5.8. Errores en entrenamiento con partición optimizada

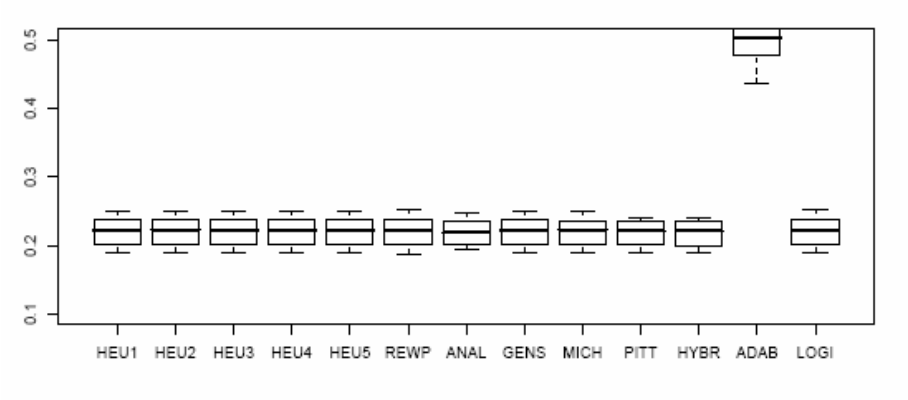


Fig 5.9. Errores prueba con partición optimizada

Se muestra además, los casos concretos del heurístico 1, Michigan y Pittsburgh, en el que se observa como ha mejorado el error de clasificación de forma sustancial cuando se realizan los experimentos con la partición optimiza, obtenida a partir de nuestro método, frente a la partición uniforme inicial.

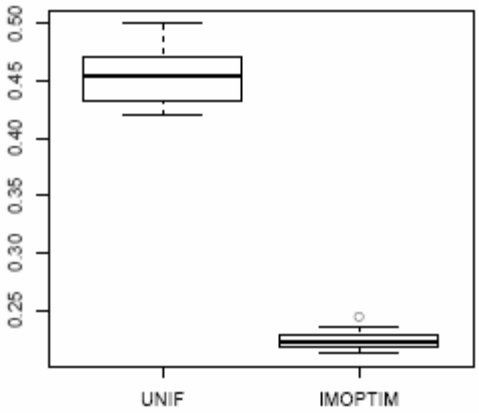


Fig 5.10. Detalle de la mejora en prueba del algoritmo Heuristico1

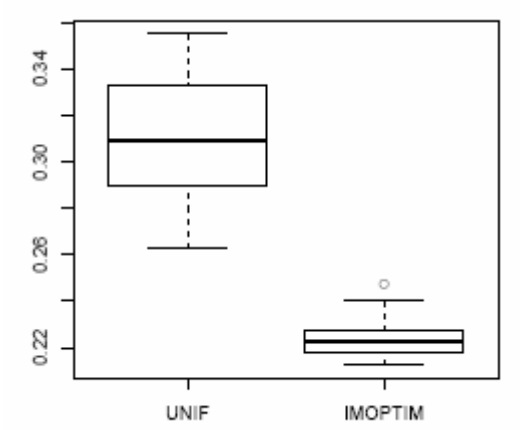


Fig 5.11. Detalle de la mejora en prueba del algoritmo Michigan

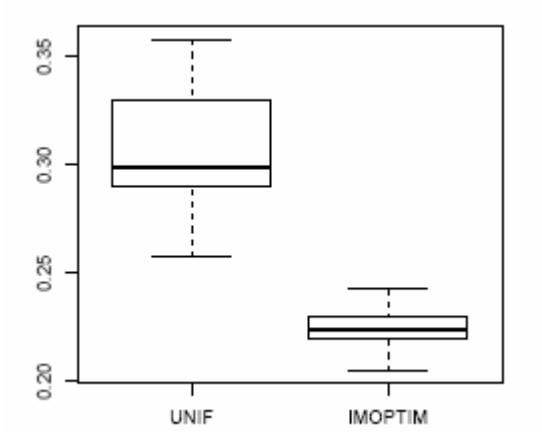


Fig 5.12. Detalle de la mejora en prueba del algoritmo Pittsburgh

5.5.3. Dataset Gauss-5

En el dataset Gauss-5 se da la misma situación que en el problema anterior, si bien la densidad de ejemplos es menor y hay 5 clases diferentes, lo cual supone un problema para algunos algoritmos heurísticos. Al haber dos variables y estar limitado a 3 el número de términos lingüísticos, es de esperar que la asignación de pesos a las reglas tenga una gran influencia en el aprendizaje.

En particular, los únicos métodos que consiguen un error cercano al óptimo son los basados en inferencia en suma de votos y el heurístico con

ajuste analítico. Como puede comprobarse en la tabla siguiente, el ajuste de pertenencias ha producido mejoras tanto en el error de entrenamiento como en el de prueba en los 13 problemas. Es destacable que las diferencias en el error tanto en prueba como en entrenamiento de los 13 algoritmos de aprendizaje son mucho menos acusadas tras realizar el ajuste genético.

	Uniforme		Optimizada	
	Entrenamiento	Prueba	Entrenamiento	Prueba
HEU1	0.55	0.55	**0.31**	**0.33**
HEU2	0.52	0.52	**0.31**	**0.33**
HEU3	0.49	0.49	**0.31**	**0.33**
HEU4	0.45	0.45	**0.31**	**0.33**
HEU5	0.39	0.39	**0.32**	**0.32**
REWP	0.43	0.44	**0.31**	**0.33**
ANAL	0.31	0.31	**0.30**	**0.31**
GENS	0.41	0.41	**0.32**	**0.32**
MICH	0.56	0.57	**0.31**	**0.32**
PITT	0.54	0.55	**0.31**	**0.32**
HYBR	0.52	0.52	**0.31**	**0.32**
ADAB	0.31	0.32	**0.31**	**0.32**
LOGI	0.31	0.32	**0.31**	**0.32**

Tabla 5.10. Experimentos con el dataset GAUSS-5 con partición uniforme y optimizada

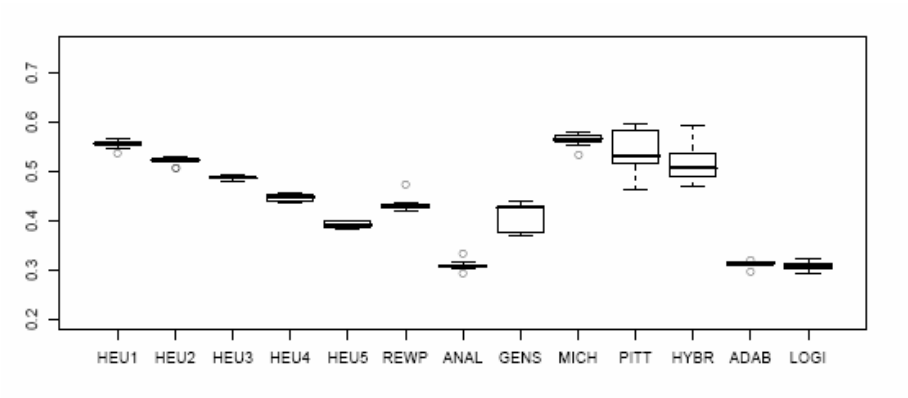


Fig 5.13. Errores en entrenamiento con partición uniforme

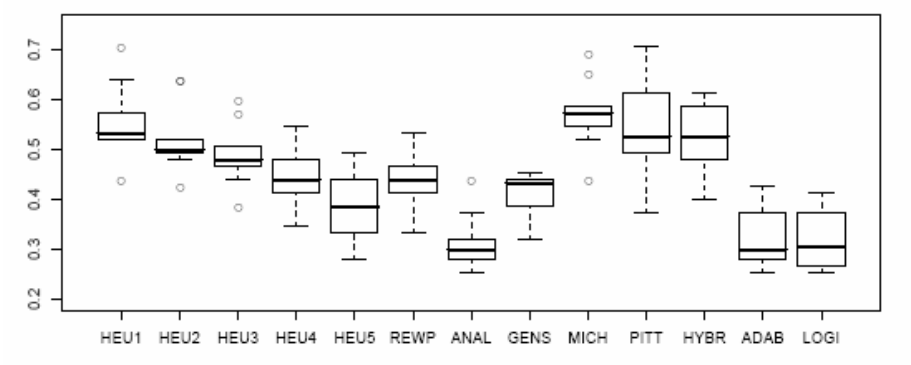


Fig 5.14. Errores en prueba con partición uniforme

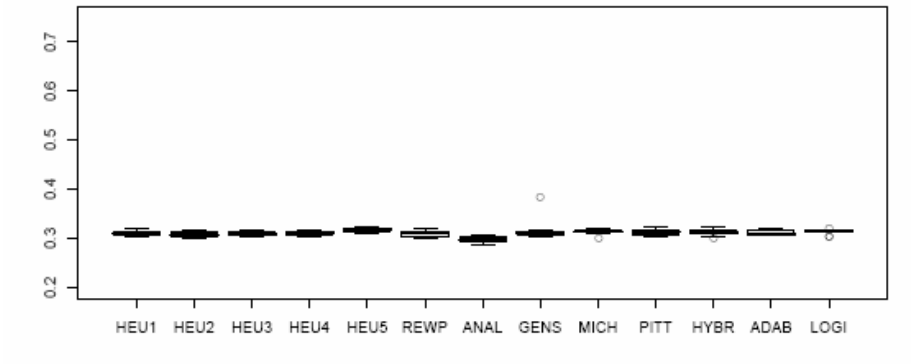


Fig 5.15. Errores en entrenamiento con partición optimizada

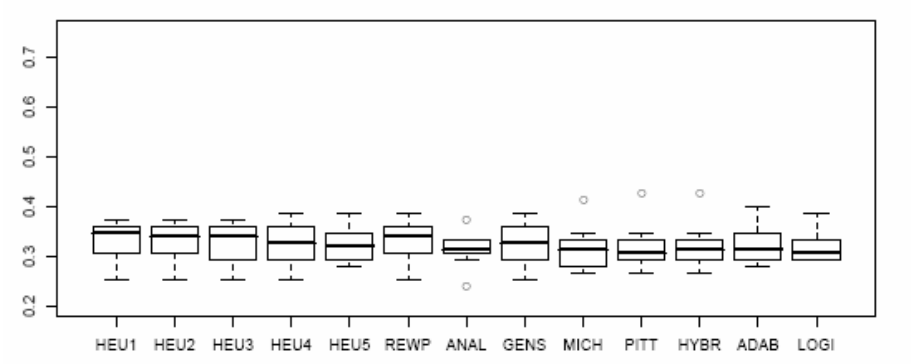


Fig 5.16. Errores en prueba con partición optimizada

Se muestra además, los casos concretos del heurístico 1, Michigan y Pittsburgh, en el que se observa como ha mejorado el error de clasificación de forma sustancial cuando se realizan los experimentos con la partición optimizada, obtenida a partir de nuestro método, frente a la partición uniforme inicial.

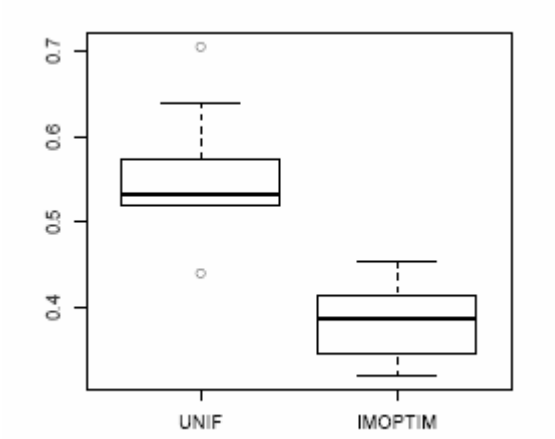


Fig 5.17. Detalle de la mejora en prueba del algoritmo Heuristico1

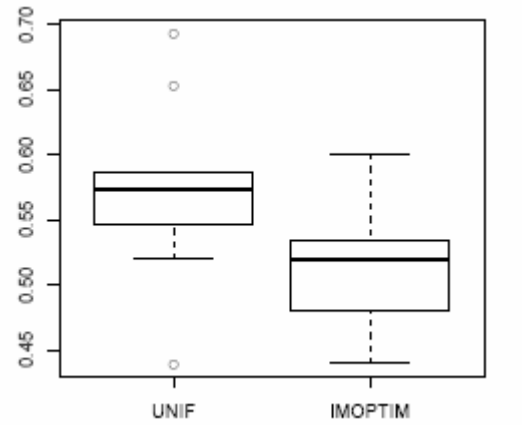


Fig 5.18. Detalle de la mejora en prueba del algoritmo Michigan

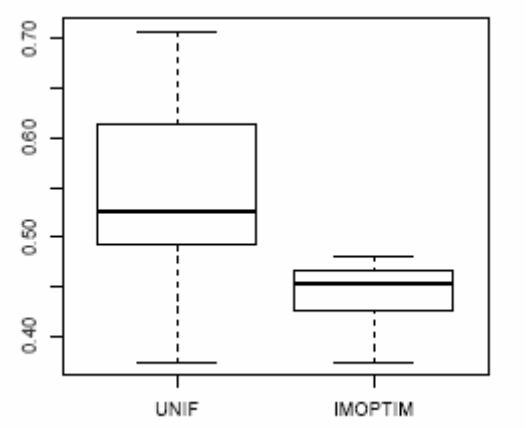


Fig 5.19. Detalle de la mejora en prueba del algoritmo Pittsburg

5.5.4. Dataset Pima

En el dataset Pima la densidad de ejemplos es aproximadamente uniforme en todas las variables, por lo que la partición óptima no es muy distante de la uniforme. Si bien hay mejoras en el error de entrenamiento en 12 de los 13 métodos, y en prueba en 9 de los 13, ninguna de las diferencias es significativa.

	Uniforme		Optimizada	
	Entrenamiento	Prueba	Entrenamiento	Prueba
HEU1	0.25	0.28	**0.18**	**0.26**
HEU2	0.24	0.27	**0.17**	**0.25**
HEU3	0.23	0.25	**0.16**	**0.25**
HEU4	0.23	0.25	**0.16**	**0.25**
HEU5	0.23	0.25	**0.16**	**0.25**
REWP	0.20	**0.26**	**0.13**	0.27
ANAL	0.21	0.28	**0.16**	**0.27**
GENS	0.20	0.36	**0.14**	**0.26**
MICH	0.35	0.35	**0.35**	**0.35**
PITT	0.26	0.28	**0.24**	**0.28**
HYBR	0.23	**0.27**	**0.22**	0.28
ADAB	**0.23**	**0.25**	0.32	0.34
LOGI	0.21	**0.23**	**0.21**	0.24

Tabla 5.11. Experimentos con el dataset PIMA con partición uniforme y optimizada

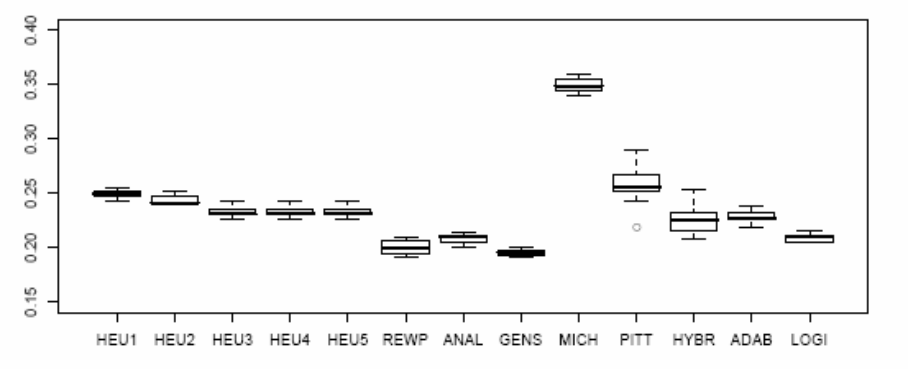


Fig 5.20. Errores de entrenamiento con partición uniforme

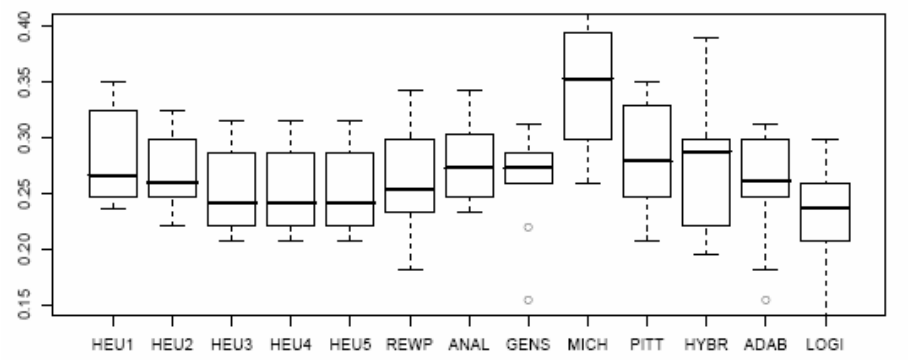


Fig 5.21. Errores de prueba con partición uniforme

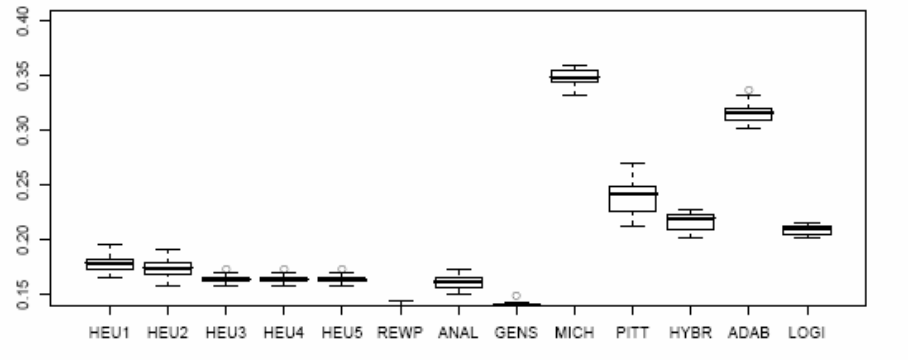


Fig 5.22. Errores de entrenamiento con partición optimizada

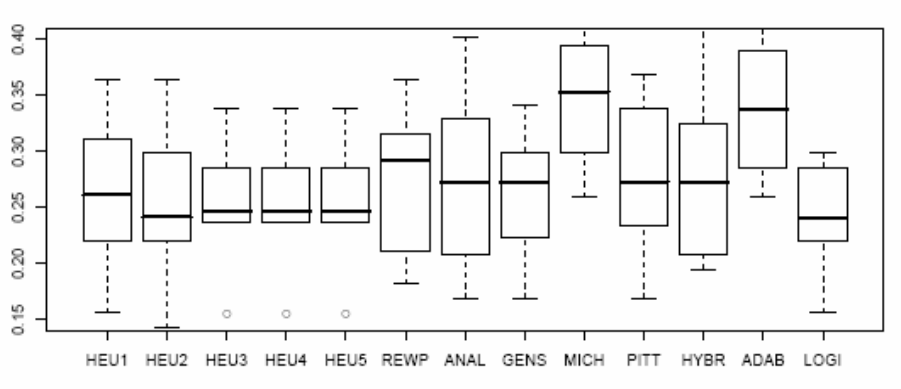


Fig 5.23. Errores de prueba con partición optimizada

5.5.5. Dataset Glass

El dataset Glass es un problema complejo en el sentido de que trabajamos con muchos atributos y esto dificulta el proceso. No se aprecian mejoras significativas tanto en entrenamiento como en prueba ya que tan solo 7 de los 13 métodos mejoran el error con la partición optimizada.

	Uniforme		Optimizada	
	Entrenamiento	**Prueba**	**Entrenamiento**	**Prueba**
HEU1	**0.32**	**0.38**	0.36	0.41
HEU2	**0.31**	**0.37**	0.33	0.39
HEU3	**0.30**	**0.37**	0.32	0.38
HEU4	**0.27**	0.36	0.28	**0.35**
HEU5	0.26	0.35	**0.26**	**0.35**
REWP	0.29	0.37	**0.24**	**0.35**
ANAL	**0.26**	**0.37**	0.31	0.43
GENS	0.24	0.36	**0.22**	**0.30**
MICH	0.46	0.49	**0.36**	**0.39**
PITT	0.36	**0.37**	**0.33**	0.43
HYBR	0.35	0.43	**0.29**	**0.43**
ADAB	0.23	0.43	**0.22**	**0.32**
LOGI	**0.27**	**0.32**	0.30	0.38

Tabla 5.12. Experimentos con el dataset GLASS con partición uniforme y optimizada

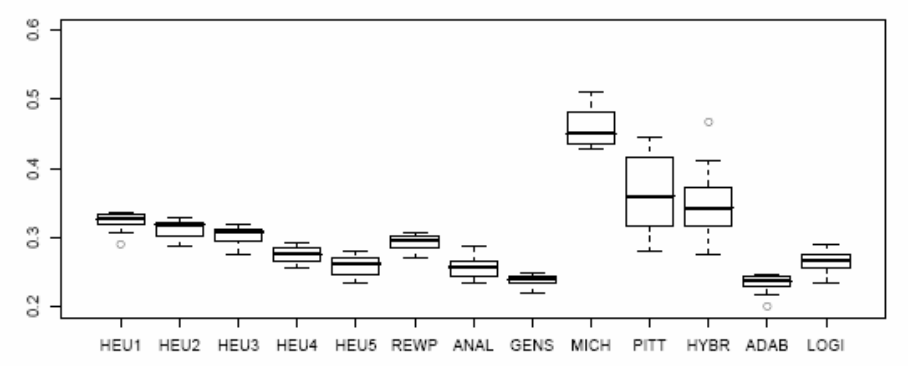


Fig 5.24. Errores en entrenamiento con partición uniforme

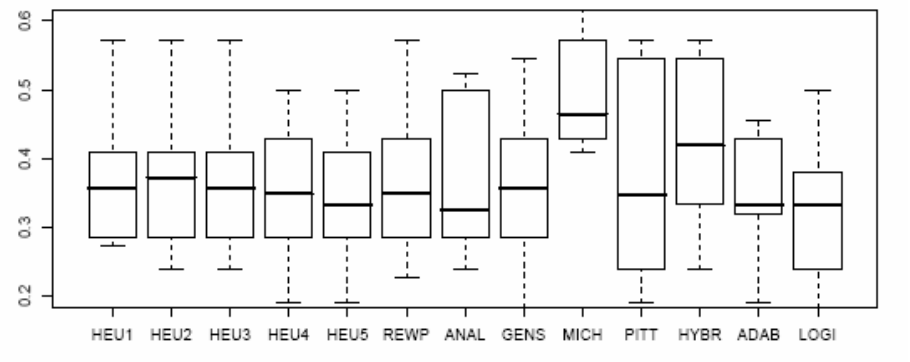


Fig 5.25. Errores en prueba con partición uniforme

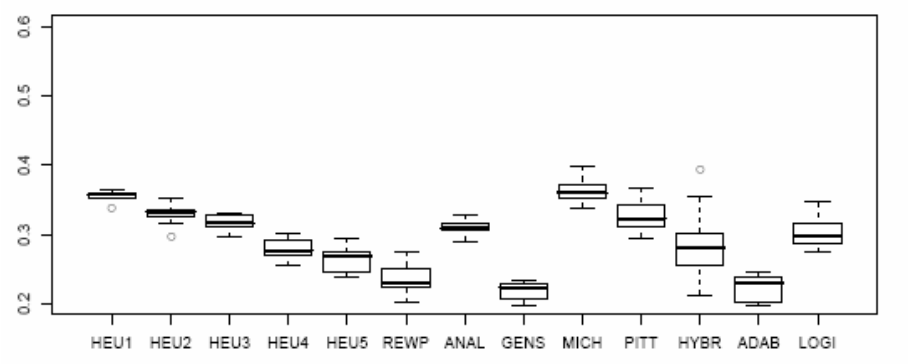


Fig 5.26. Errores en entrenamiento con partición optimizada

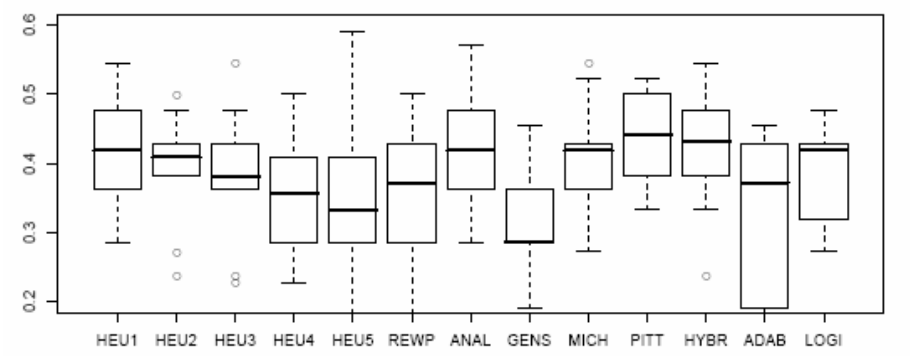


Fig 5.27. Errores en prueba con partición optimizada

4.5.6. Dataset Skulls

El dataset Skulls tiene un comportamiento un tanto inusual ya que, en entrenamiento no hay mejoras con ninguno de los métodos, en cambio en prueba 11 de los 13 métodos obtienen unos resultados ligeramente mejores con la partición optimizada frente a la partición uniforme.

	Uniforme		Optimizada	
	Entrenamiento	Prueba	Entrenamiento	Prueba
HEU1	**0.64**	0.85	0.74	**0.79**
HEU2	**0.64**	0.86	0.74	**0.79**
HEU3	**0.60**	0.84	0.72	**0.79**
HEU4	**0.60**	0.83	0.71	**0.78**
HEU5	**0.66**	0.81	0.71	**0.73**
REWP	**0.62**	0.86	0.72	**0.79**
ANAL	**0.53**	0.81	0.62	**0.75**
GENS	**0.51**	0.81	0.60	**0.71**
MICH	**0.64**	**0.83**	0.73	0.84
PITT	**0.63**	0.81	0.70	**0.77**
HYBR	**0.62**	**0.81**	0.74	0.84
ADAB	**0.50**	0.75	0.60	**0.74**
LOGI	**0.59**	0.75	0.62	**0.71**

Tabla 5.13. Experimentos con el dataset SKULLS con partición uniforme y optimizada

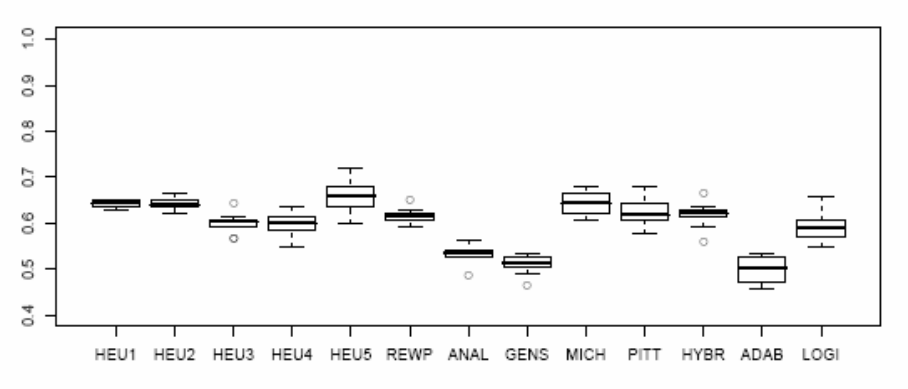


Fig 5.28. Errores en entrenamiento con partición uniforme

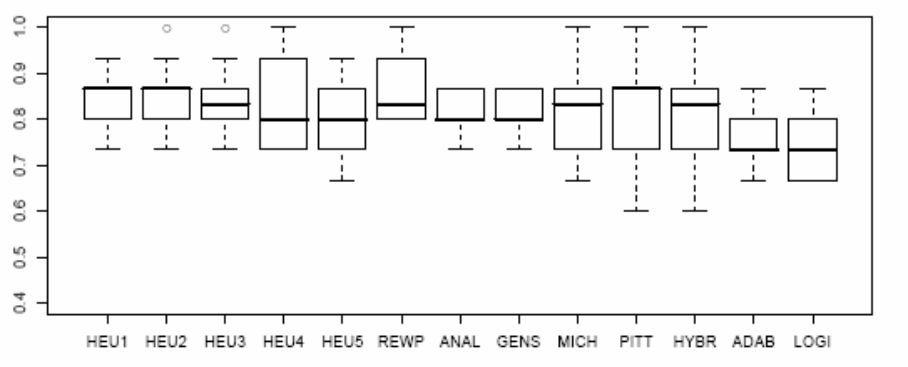


Fig 5.29. Errores en prueba con partición uniforme

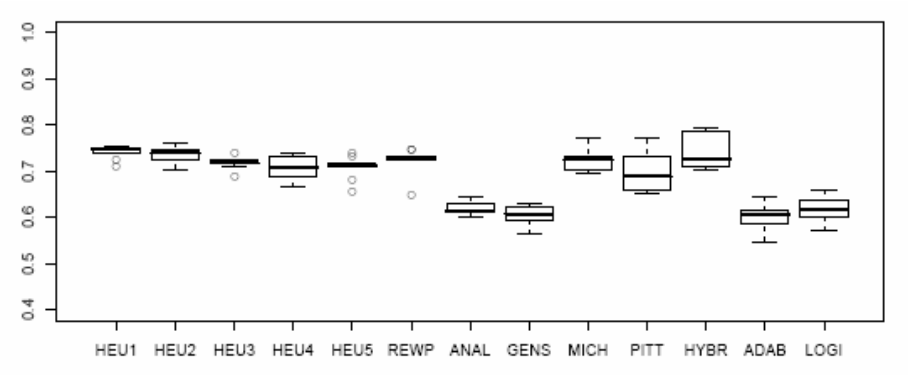


Fig 5.30. Errores en entrenamiento con partición optimizada

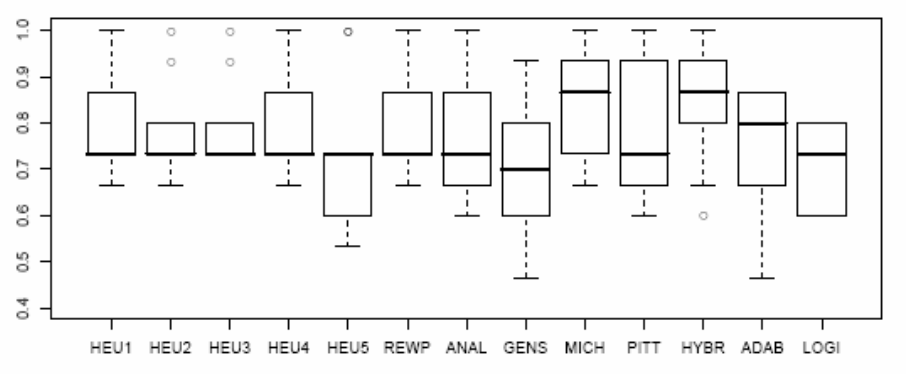


Fig 5.31. Errores en prueba con partición optimizada

5.5.7. Dataset Cancer

Este problema es trivial para los clasificadores lineales, pero en cambio precisa de un número muy elevado de reglas en los algoritmos heurísticos. A diferencia de los datasets precedentes la simple selección de una pequeña fracción de estas reglas no conduce a resultados aceptables, como puede comprobarse en los resultados del algoritmo de selección genética. Asimismo, el algoritmo Adaboost no ha conseguido obtener una base de conocimiento de menos de 25 reglas capaz de clasificar el problema con precisión aceptable, a diferencia de los restantes algoritmos de aprendizaje genéticos, y en especial del algoritmo Logitboost, capaz de resolver el problema con 10 reglas. Con la partición optimizada se consiguen mejoras en el entrenamiento en 8 de los 13 métodos, y mejoras en prueba en 11 de los 13 métodos.

	Uniforme		Optimizada	
	Entrenamiento	Prueba	Entrenamiento	Prueba
HEU1	0.0178	0.040	**0.0168**	**0.030**
HEU2	0.0176	0.039	**0.0153**	**0.031**
HEU3	**0.0170**	0.037	0.0173	**0.031**
HEU4	**0.0170**	0.037	0.0173	**0.031**
HEU5	**0.0170**	0.037	0.0173	**0.031**
REWP	0.0103	0.087	**0.0099**	**0.039**
ANAL	0.0178	0.081	**0.0168**	**0.040**
GENS	**0.0089**	0.346	0.0114	**0.029**
MICH	**0.0383**	**0.043**	0.0563	0.062
PITT	0.0684	0.077	**0.0332**	**0.037**
HYBR	0.0251	**0.036**	**0.0203**	0.039
ADAB	0.1671	0.205	**0.0919**	**0.102**
LOGI	0.0269	0.033	**0.0232**	**0.027**

Tabla 5.14. Experimentos con el dataset CANCER con partición uniforme y optimizada

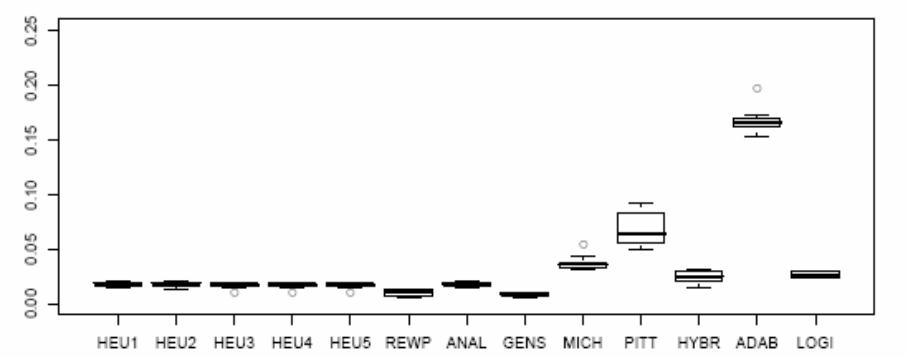


Fig 5.32. Errores en entrenamiento con partición uniforme

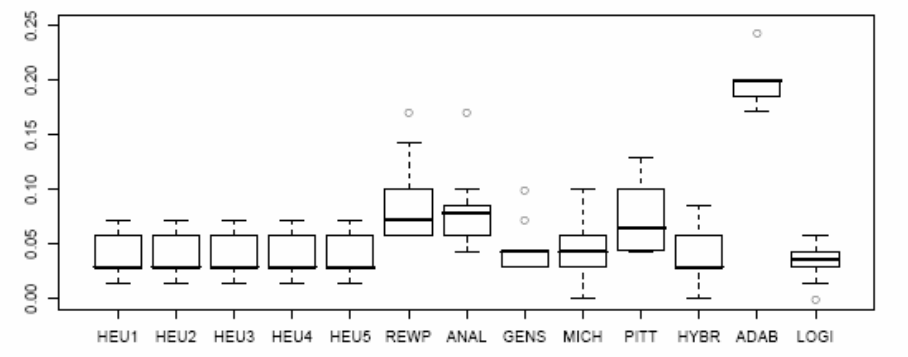


Fig 5.33. Errores en prueba con partición uniforme

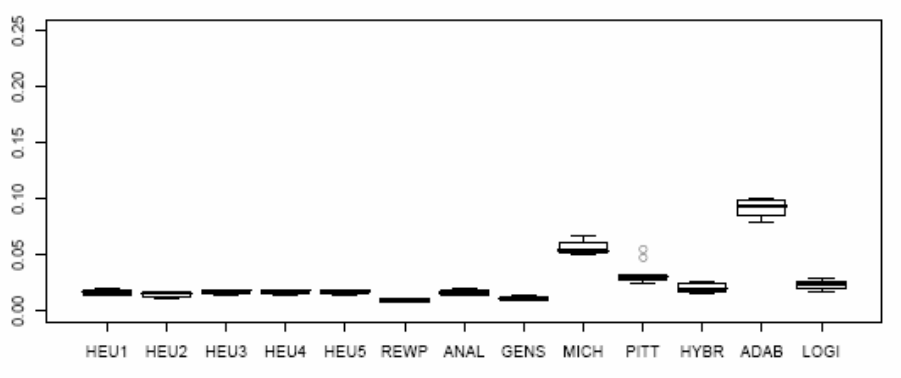


Fig 5.34. Errores en entrenamiento con partición optimizada

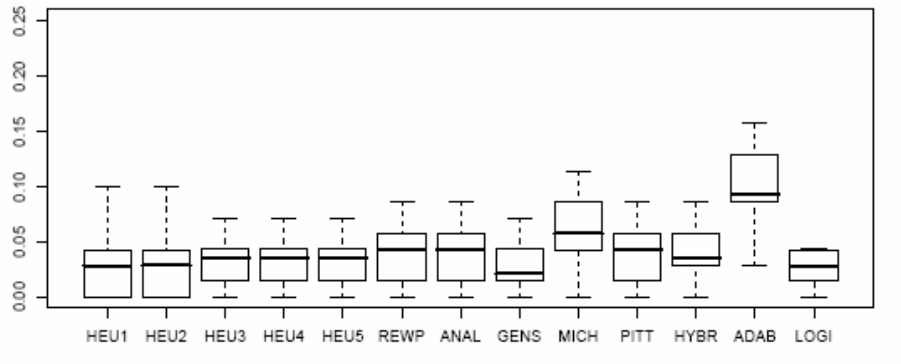


Fig 5.35. Errores en prueba con partición optimizada

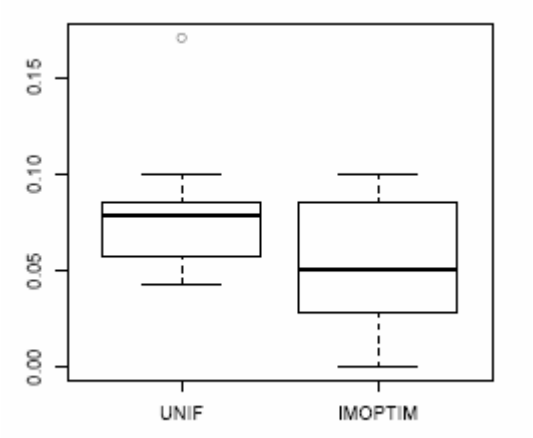


Fig 5.36. Detalle de la mejora en prueba del algoritmo de Ajuste Analítico

5.6. Trabajando con datos con imprecisión

Dentro del segundo tipo de problemas a los que se puede aplicar nuestra definición de Información Mutua, abordaremos un tipo de problemas que son bastante comunes en la vida real, y que son aquellos que introducen imprecisión en los ejemplos de partida.

Para nuestra experimentación hemos realizado dos tipos de ensayos:

a) Añadir imprecisión a conjuntos de ejemplos que no tenían imprecisión

b) Crear nuestros propios conjuntos de ejemplos imprecisos.

Nosotros intentaremos ver hasta que punto es importante la imprecisión para que un sistema de clasificación sea eficiente.

Los problemas que se plantean constan de un conjunto de ejemplos, cada uno de los cuales esta formado por atributos, que determinar una clase a la que pertenece el ejemplo. La principal característica de los valores de los atributos es que, al contrario de los que ocurría en el caso de datos precisos, su valor, para un ejemplo determinado y un atributo específico, no viene definido por un valor numérico concreto, sino que cada valor de un atributo es un intervalo.

A1	A2	Clase
[0.75,0.85]	[0.45,0.55]	1

Esta es una muestra de un ejemplo con imprecisión, donde los valores de los atributos son intervalos y no valores concretos.

A1	A2	Clase
0.588977	[0, 1]	0

Y esta otra es una muestra de un ejemplo que en el atributo 2 tiene un valor desconocido, que se representa con el intervalos [0,1].

Realizaremos nuestros experimentos introduciendo información imprecisa en los conjuntos de ejemplos utilizados en le experimentación sin imprecisión y luego plantearemos otros tipos de conjuntos de ejemplos que manejen información imprecisa.

Para este tipo de conjuntos de ejemplos utilizaremos una partición uniforme de tres etiquetas lingüísticas, como se utilizó para el caso de datos completos y sin imprecisión.

A la partición uniforme inicial se le aplicará un proceso genético, en este caso multiobjetivo, que utilizará a definición de información mutua para llegar a obtener una partición optimizada a partir de la original.

Se dividirá el conjunto de ejemplos en ejemplos de entrenamiento y ejemplos de test realizando 10 conjuntos distintos para nuestros experimentos. Las pruebas se realizarán tanto sobre el conjunto de entrenamiento como sobre el conjunto de test.

5.6.1. Conjunto de datos con imprecisión añadida

En este caso se han tomado los conjuntos de ejemplos utilizados en la experimentación de datos sin imprecisión y se le ha introducido una imprecisión del 1%

A continuación veremos los resultados de error medio para los distintos clasificadores tanto para la partición uniforme como para la optimizada tanto en entrenamiento como en prueba para cada uno de los datasets.

Asimismo se muestra de forma gráfica los errores medios para que se pueda observar mejor los resultados obtenidos.

5.6.1.1. Dataset Iris

Como puede comprobarse en los boxplots, en el problema Iris hay mejoras en el error medio de entrenamiento en 5 de los 13 algoritmos, y mejoras en prueba en 7 de los 13.

	Uniforme		Optimizada Imprecisa	
	Entrenamiento	Prueba	Entrenamiento	Prueba
HEU1	**0.027**	**0.027**	0.040	0.040
HEU2	**0.029**	**0.033**	0.040	0.040
HEU3	0.061	0.060	**0.040**	**0.040**
HEU4	0.067	0.067	**0.040**	**0.040**
HEU5	0.067	0.067	**0.040**	**0.040**
REWP	**0.016**	0.047	0.040	**0.040**
ANAL	**0.027**	**0.033**	0.040	0.040
GENS	0.053	0.067	**0.033**	**0.047**
MICH	**0.037**	0.047	0.040	**0.047**
PITT	**0.033**	**0.060**	0.062	0.073
HYBR	**0.033**	**0.047**	0.064	0.067
ADAB	0.019	0.047	**0.013**	**0.040**
LOGI	**0.026**	**0.040**	0.028	0.047

Tabla 5.15. Experimentos con el dataset IRIS con partición uniforme y optimizada imprecisa

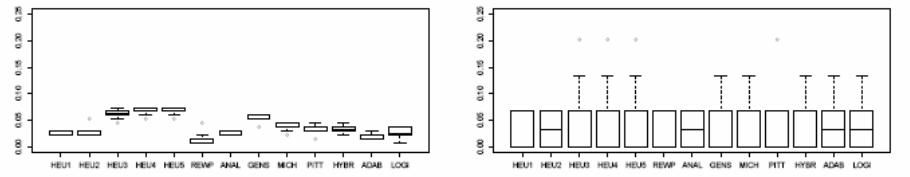


Fig 5.37. Errores en entrenamiento y prueba con partición uniforme

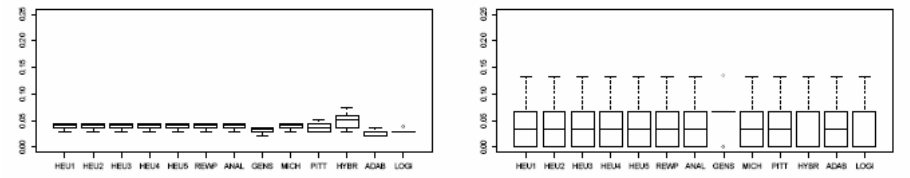


Fig 5.38.Errores en entrenamiento y prueba con partición optimizada

El comportamiento es similar al observado con datos precisos, como se puede observar en los boxplots que se muestran.

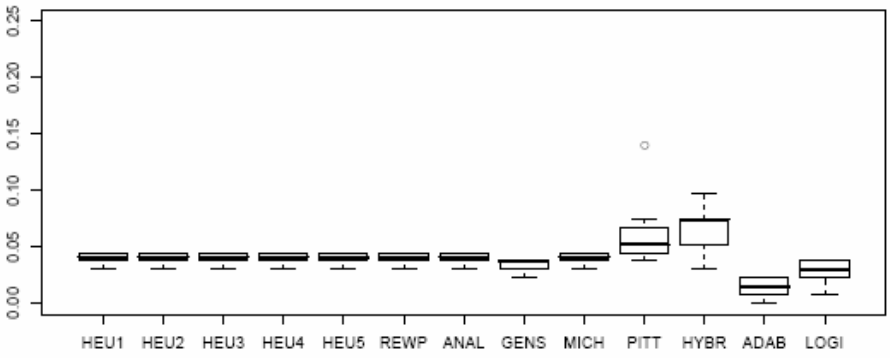


Fig 5.39. Errores en entrenamiento con partición optimizada imprecisa

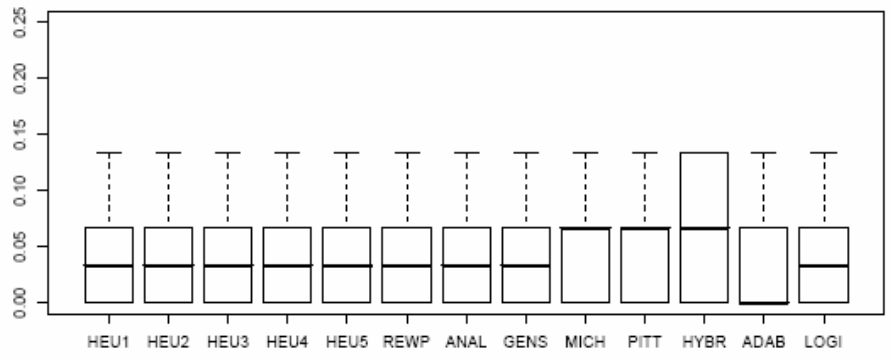


Fig 5.40. Errores en prueba con partición optimizada imprecisa

5.6.1.2. Dataset Gauss

El dataset Gauss es un problema de dos clases, con una alta densidad de ejemplos. No es de esperar que la elección de las particiones sea relevante en los clasificadores con inferencia por suma de votos. Las particiones con Información Mutua maximizada sí que conducen a un incremento notable de potencia en los clasificadores basados en inferencia por máximo ganador. El ajuste de pertenencias ha mejorado el error, tanto en entrenamiento como en prueba, de 9 de los 13 clasificadores. Además se observa que el error se mantiene uniforme tanto en entrenamiento como en prueba.

	Uniforme		Optimizada Imprecisa	
	Entrenamiento	Prueba	Entrenamiento	Prueba
HEU1	0.45	0.45	**0.22**	**0.22**
HEU2	0.43	0.43	**0.22**	**0.22**
HEU3	0.26	0.27	**0.22**	**0.22**
HEU4	0.26	0.27	**0.22**	**0.22**
HEU5	0.26	0.27	**0.22**	**0.22**
REWP	0.30	0.30	**0.22**	**0.22**
ANAL	**0.20**	**0.20**	0.22	0.23
GENS	**0.21**	**0.21**	0.22	0.22
MICH	0.31	0.31	**0.22**	**0.22**
PITT	0.31	0.31	**0.22**	**0.22**
HYBR	0.27	0.27	**0.22**	**0.22**
ADAB	**0.21**	**0.21**	0.22	0.23
LOGI	**0.20**	**0.20**	0.22	0.22

Tabla 5.16. Experimentos con el dataset GAUSS con partición uniforme y optimizada imprecisa

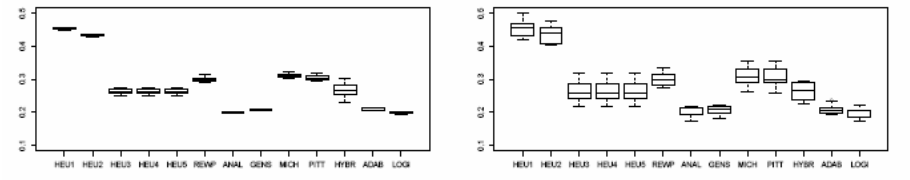


Fig 5.41. Errores en entrenamiento y prueba con partición uniforme

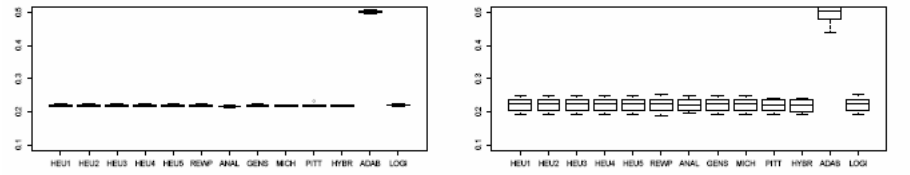


Fig 5.42. Errores en entrenamiento y prueba con partición optimizada

En este caso el comportamiento es idéntico al obtenido con datos precisos, por lo que se deduce que nuestro método es adecuado para este tipo de problemas.

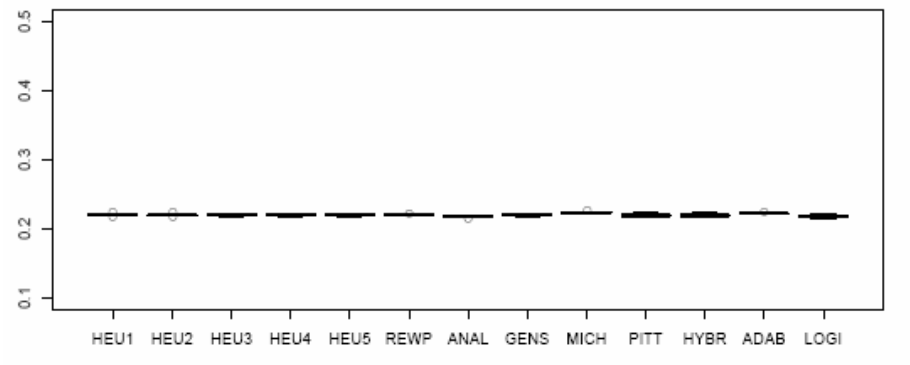


Fig 5.43. Errores en entrenamiento con partición optimizada imprecisa

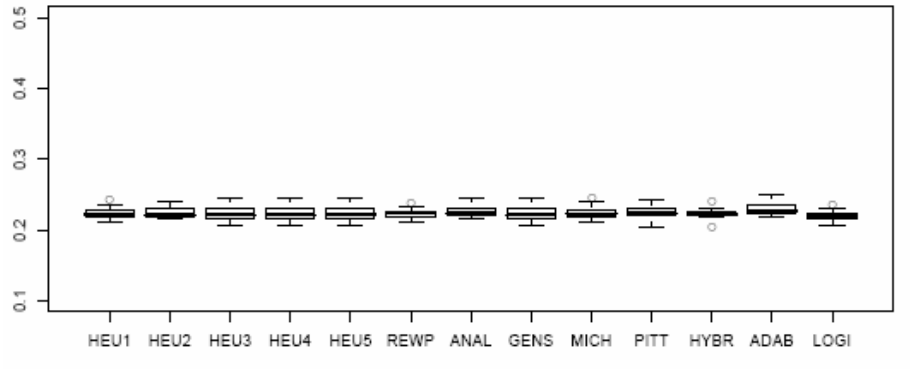


Fig 5.44. Errores en prueba con partición optimizada imprecisa

5.6.1.3. Dataset Gauss-5

En el dataset Gauss-5 se da la misma situación que el problema anterior, si bien la densidad de ejemplos es menor y hay 5 clases diferentes, lo cual supone un problema para algunos algoritmos heurísticos. Al haber dos variables y estar limitado a 3 el número de términos lingüísticos, es de esperar que la asignación de pesos a las reglas tenga una gran influencia en el aprendizaje.

Tras la optimización se han producido mejoras en el error de entrenamiento en los 9 de los 13 problemas, y esas mejoras han repercutido en el error de prueba en 12 de los 13 algoritmos.

En este caso no hay mejora en todos los métodos como ocurría con datos precisos.

	Uniforme		Optimizada Imprecisa	
	Entrenamiento	Prueba	Entrenamiento	Prueba
HEU1	0.55	0.55	**0.39**	**0.39**
HEU2	0.52	0.52	**0.38**	**0.38**
HEU3	0.49	0.49	**0.38**	**0.38**
HEU4	0.45	0.45	**0.38**	**0.38**
HEU5	0.39	0.39	**0.37**	**0.37**
REWP	0.43	0.44	**0.37**	**0.37**
ANAL	**0.31**	**0.31**	0.36	0.36
GENS	**0.41**	0.41	0.42	**0.41**
MICH	0.56	0.57	**0.50**	**0.52**
PITT	0.54	0.55	**0.43**	**0.45**
HYBR	0.52	0.52	**0.42**	**0.44**
ADAB	**0.31**	0.32	0.33	**0.33**
LOGI	**0.31**	0.32	0.32	**0.33**

Tabla 5.17. Experimentos con el dataset GAUSS-5 con partición uniforme y optimizada imprecisa

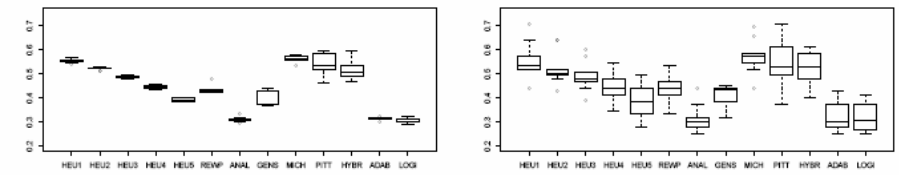


Fig 5.45. Errores en entrenamiento y prueba con partición uniforme

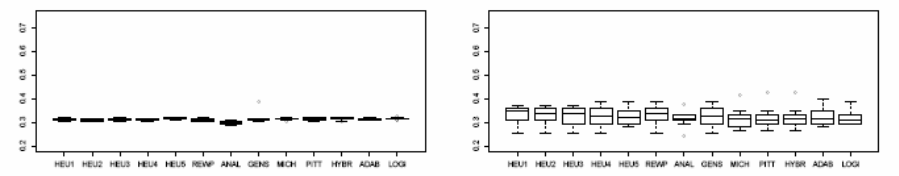


Fig 5.46. Errores en entrenamiento y prueba con partición optimizada

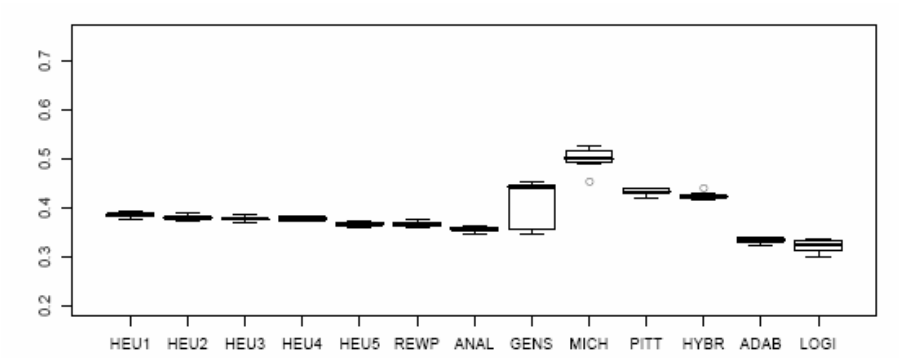


Fig 5.47. Errores en entrenamiento con partición optimizada imprecisa

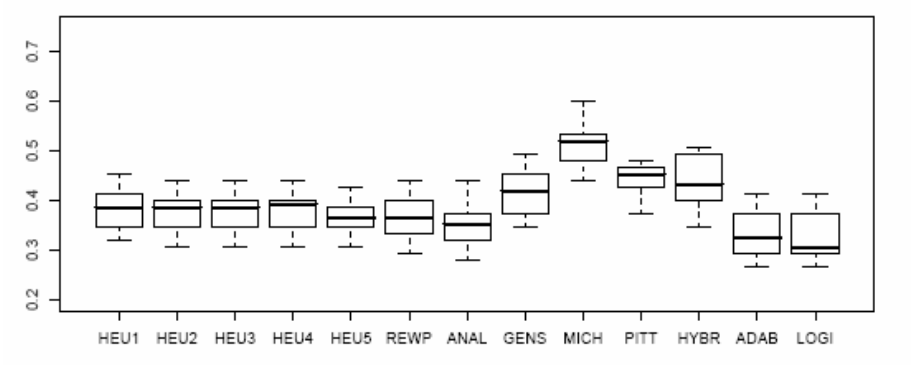


Fig 5.48. Errores en prueba con partición optimizada imprecisa

5.6.1.4. Dataset Pima

En el dataset Pima la densidad de ejemplos es aproximadamente uniforme en todas las variables. Si bien hay mejoras en el error de entrenamiento en 11 de los 13 métodos, y en prueba en 6 de los 13, ninguna de las diferencias es significativa.

	Uniforme		Optimizada Imprecisa	
	Entrenamiento	Prueba	Entrenamiento	Prueba
HEU1	0.25	**0.28**	**0.22**	0.29
HEU2	0.24	**0.27**	**0.21**	0.28
HEU3	0.23	**0.25**	**0.19**	0.26
HEU4	0.23	**0.25**	**0.19**	0.26
HEU5	0.23	**0.25**	**0.19**	0.26
REWP	0.20	0.26	**0.18**	**0.26**
ANAL	0.21	0.28	**0.19**	**0.27**
GENS	0.20	0.36	**0.17**	**0.26**
MICH	0.35	0.35	**0.34**	**0.35**
PITT	0.26	0.28	**0.24**	**0.26**
HYBR	0.23	0.27	**0.22**	**0.26**
ADAB	**0.23**	**0.25**	0.26	0.30
LOGI	**0.21**	**0.23**	0.22	0.24

Tabla 5.18. Experimentos con el dataset PIMA con partición uniforme y optimizada imprecisa

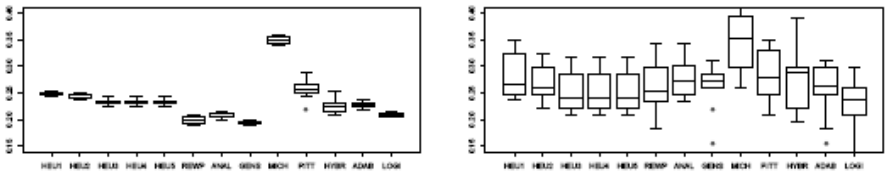


Fig 5.49. Errores en entrenamiento y prueba con partición uniforme

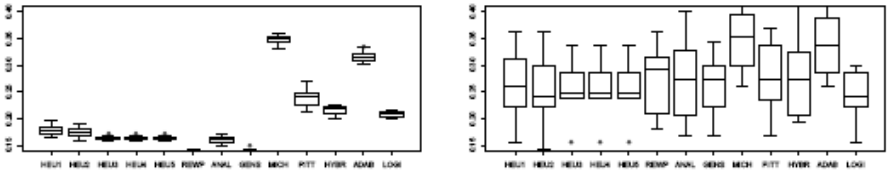


Fig 5.50. Errores en entrenamiento y prueba con partición optimizada

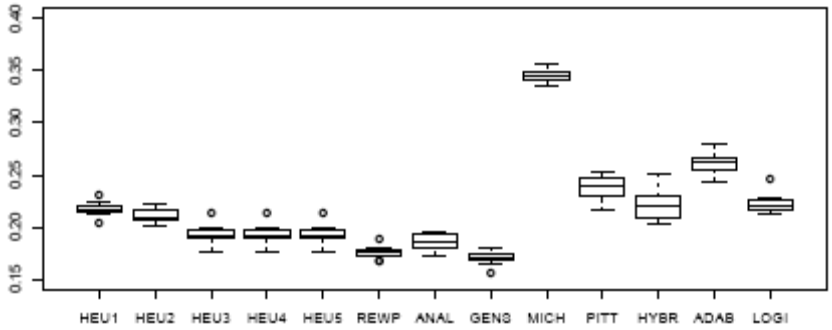


Fig 5.51. Errores en entrenamiento con partición optimizada imprecisa

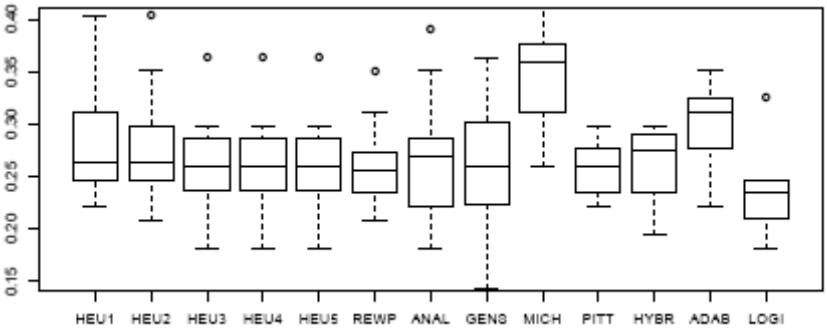


Fig 5.52. Errores en prueba con partición optimizada imprecisa

5.6.1.5. Dataset Glass

A pesar de ser un dataset un tanto especial, se observa que hay mejoras en 10 de los 13 métodos en entrenamiento y en 9 de los 13 en prueba, aunque las mejoras no sean muy significativas.

	Uniforme		Optimizada Imprecisa	
	Entrenamiento	Prueba	Entrenamiento	Prueba
HEU1	**0.32**	**0.38**	0.36	0.41
HEU2	**0.31**	**0.37**	0.34	0.38
HEU3	**0.30**	0.37	0.33	**0.37**
HEU4	0.27	0.36	**0.27**	**0.35**
HEU5	0.26	0.35	**0.25**	**0.35**
REWP	0.29	0.37	**0.25**	**0.33**
ANAL	0.26	0.37	**0.25**	**0.33**
GENS	0.24	0.36	**0.21**	**0.35**
MICH	0.46	0.49	**0.38**	**0.41**
PITT	0.36	**0.37**	**0.36**	0.43
HYBR	0.35	0.43	**0.33**	**0.43**
ADAB	0.23	0.43	**0.22**	**0.31**
LOGI	0.27	**0.32**	**0.27**	0.33

Tabla 5.19. Experimentos con el dataset GLASS con partición uniforme y optimizada

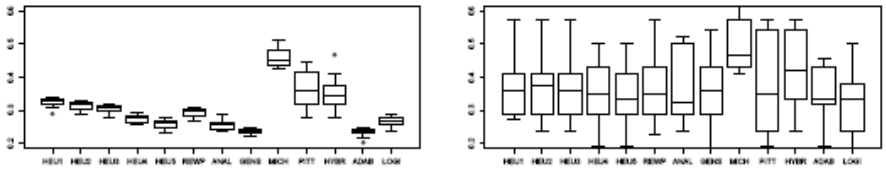


Fig 5.53. Errores en entrenamiento y prueba con partición uniforme

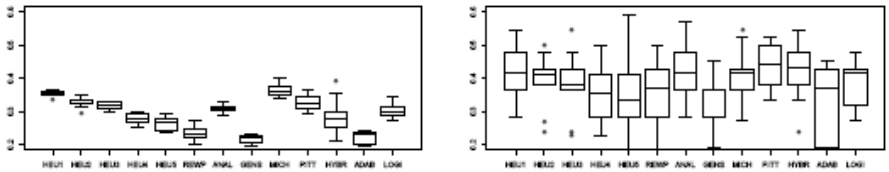


Fig 5.54. Errores en entrenamiento y prueba con partición optimizada

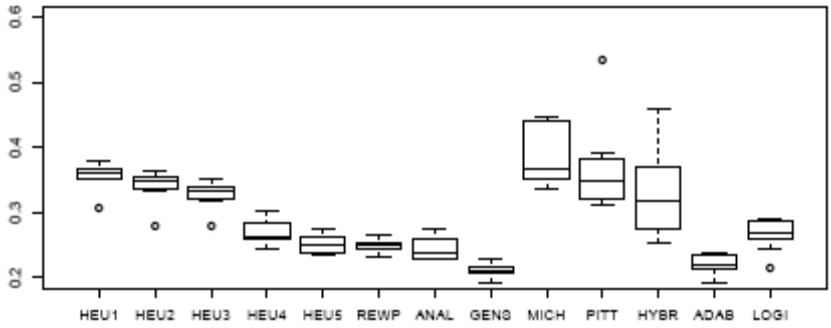


Fig 5.55. Errores en entrenamiento con partición optimizada imprecisa

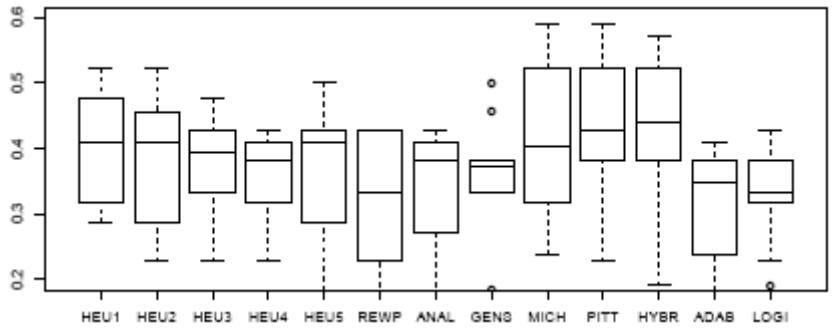


Fig 5.56. Errores en prueba con partición optimizada imprecisa

5.6.1.6. Dataset Skulls

El comportamiento de skulls es un poco inusual ya que vemos que en entrenamiento no ha habido mejora alguna pero en prueba han mejorado todos los métodos menos cuatro, aunque dicha mejora no es muy significativa.

Como ocurría con datos sin imprecisión, no se observan mejoras en entrenamiento aunque si en prueba.

	Uniforme		Optimizada Imprecisa	
	Entrenamiento	Prueba	Entrenamiento	Prueba
HEU1	**0.64**	0.85	0.70	**0.76**
HEU2	**0.64**	0.86	0.70	**0.77**
HEU3	**0.60**	0.84	0.70	**0.80**
HEU4	**0.60**	0.83	0.69	**0.80**
HEU5	**0.66**	0.81	0.69	**0.77**
REWP	**0.62**	0.86	0.70	**0.78**
ANAL	**0.53**	0.81	0.61	**0.81**
GENS	**0.51**	0.81	0.59	**0.79**
MICH	**0.64**	**0.83**	0.71	0.85
PITT	**0.63**	0.81	0.68	**0.86**
HYBR	**0.62**	**0.81**	0.66	0.85
ADAB	**0.50**	**0.75**	0.58	0.77
LOGI	**0.59**	**0.75**	0.63	0.78

Tabla 5.20. Experimentos con el dataset SKULLS con partición uniforme y optimizada

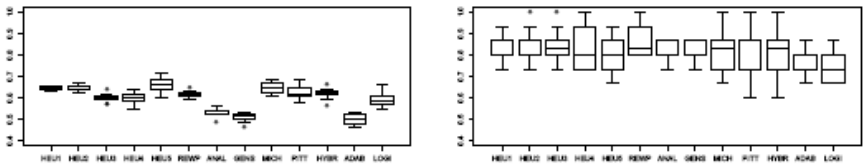


Fig 5.57. Errores en entrenamiento y prueba con partición uniforme

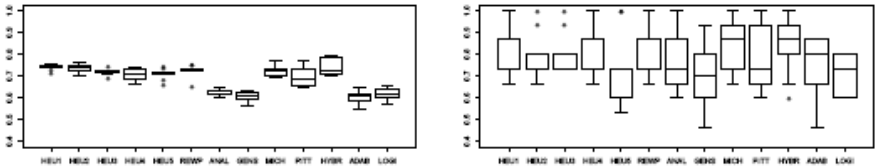


Fig 5.58. Errores en entrenamiento y prueba con partición optimizada

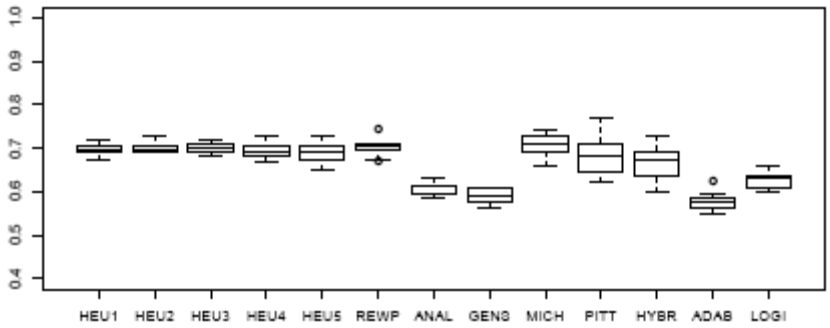


Fig 5.59. Errores en entrenamiento con partición optimizada imprecisa

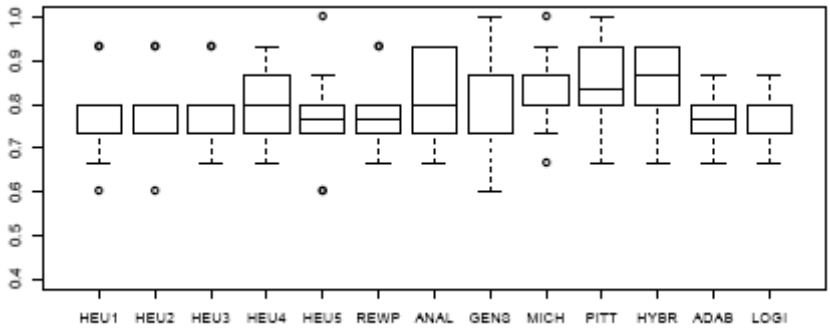


Fig 5.60. Errores en prueba con partición optimizada imprecisa

5.6.1.7. Dataset Cancer

Este problema es trivial para los clasificadores lineales, pero en cambio precisa de un número muy elevado de reglas en los algoritmos heurísticos. Con la partición optimizada se consiguen mejoras en el entrenamiento en tan solo 4 de los 13 métodos, y mejoras en prueba en tan sólo 5 de los 13 métodos.

	Uniforme		Optimizada Imprecisa	
	Entrenamiento	Prueba	Entrenamiento	Prueba
HEU1	**0.0178**	**0.040**	0.020	0.041
HEU2	**0.0176**	**0.039**	0.018	0.042
HEU3	**0.0170**	**0.037**	0.018	0.039
HEU4	**0.0170**	**0.037**	0.018	0.039
HEU5	**0.0170**	**0.037**	0.018	0.039
REWP	**0.0103**	0.087	0.016	**0.056**
ANAL	**0.0178**	0.081	0.020	**0.051**
GENS	**0.0089**	**0.346**	0.014	0.041
MICH	**0.0383**	**0.043**	0.046	0.049
PITT	0.0684	0.077	**0.034**	**0.047**
HYBR	0.0251	**0.036**	**0.025**	0.043
ADAB	0.1671	0.205	**0.072**	**0.086**
LOGI	0.0269	0.033	**0.025**	**0.030**

Tabla 5.21. Experimentos con el dataset CANCER con partición uniforme y optimizada imprecisa

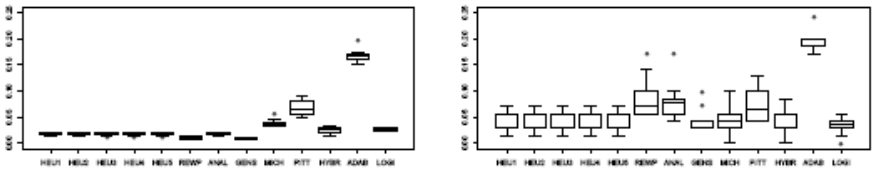


Fig 5.61. Errores en entrenamiento y prueba con partición uniforme

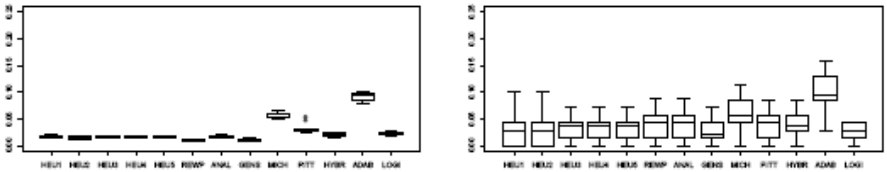


Fig 5.62. Errores en entrenamiento y prueba con partición optimizada

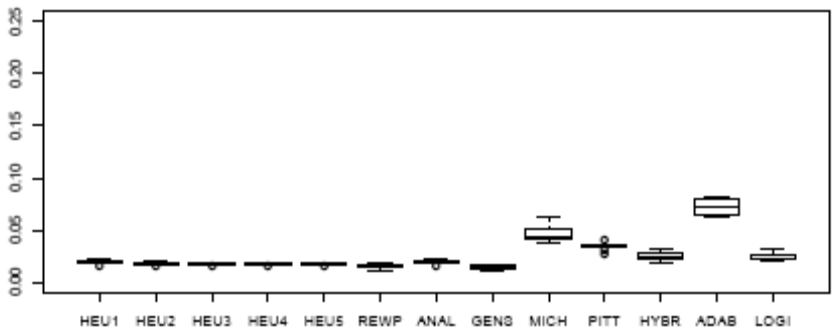


Fig 5.63. Errores en entrenamiento con partición optimizada imprecisa

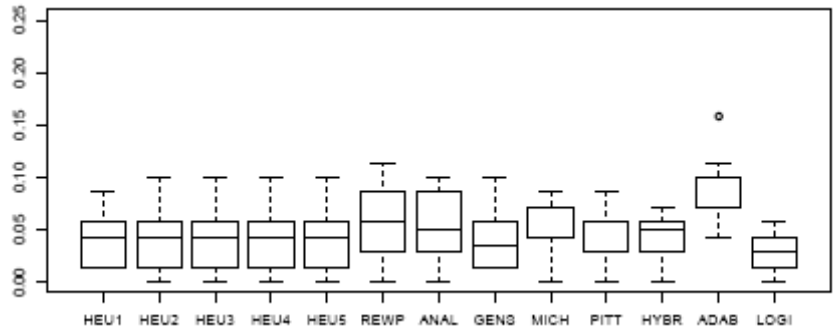


Fig 5.64. Errores en prueba con partición optimizada imprecisa

5.6.2. Conjuntos de datos imprecisos

En el apartado anterior vimos la experimentación asociada a conjuntos de datos imprecisos, de tal forma que dicha imprecisión se forzó de alguna manera.

En este punto de la experimentación se han realizado diferentes pruebas sobre distintos conjuntos de datos imprecisos con características distintas, que hemos creado a partir de pesos de básculas. Son datos reales de situaciones reales sin forzar ningún tipo de imprecisión.

Se han construido distintos datasets con pesos obtenidos a partir de una báscula que en cada caso tiene unas características distintas.

- Dataset de pesos obtenidos a partir de una báscula que nos proporciona pesos sin error de cuantización.

- Dataset de pesos obtenidos a partir de una báscula bien calibrada.
- Dataset de pesos obtenidos a partir de una báscula mal calibrada.
- Dataset de pesos obtenido de la mezcla de pesos de una báscula bien calibrada y de otra mal calibrada. Obtenidos a partir de dos básculas diferentes.
- Dataset con datos perdidos.

5.6.2.1. Báscula sin error de cuantización

En este caso se trabaja con un conjunto de ejemplo donde se presentan pesos obtenidos de una báscula sin error.

Se observa que tanto en prueba como en entrenamiento se obtienen mejoras significativas. Además, después del ajuste genético se observa una uniformidad del error obtenido en la mayor parte de los métodos. Existe mejora en 11 de los 13 métodos tanto en entrenamiento como en prueba.

	Uniforme		Optimizada	
	Entrenamiento	**Prueba**	**Entrenamiento**	**Prueba**
HEU1	0.43	0.47	**0.28**	**0.28**
HEU2	0.40	0.45	**0.28**	**0.28**
HEU3	0.31	0.36	**0.28**	**0.28**
HEU4	0.31	0.36	**0.28**	**0.28**
HEU5	0.31	0.36	**0.28**	**0.28**
REWP	**0.27**	0.31	0.28	**0.28**
ANAL	**0.26**	0.29	0.28	**0.28**
GENS	0.28	0.29	**0.25**	**0.25**
MICH	0.47	0.48	**0.28**	**0.29**
PITT	0.43	0.43	**0.30**	**0.34**
HYBR	0.42	0.43	**0.29**	**0.33**
ADAB	**0.12**	0.12	**0.16**	0.16
LOGI	**0.20**	0.20	**0.24**	0.25

Tabla 5.22 Experimentos con el dataset de pesos de una báscula sin error de cuantización

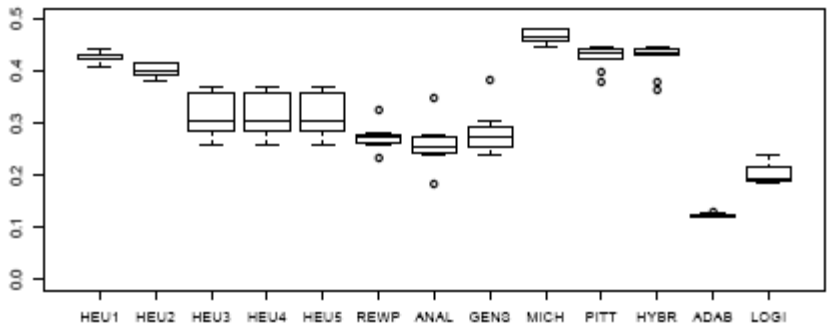


Fig 5.65. Errores en entrenamiento con partición uniforme

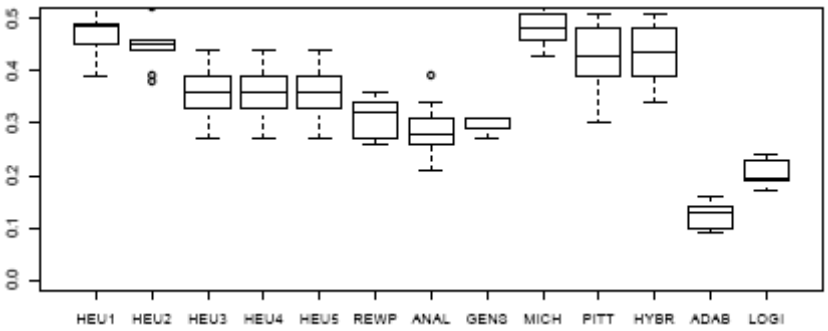


Fig 5.66. Errores en prueba con partición uniforme

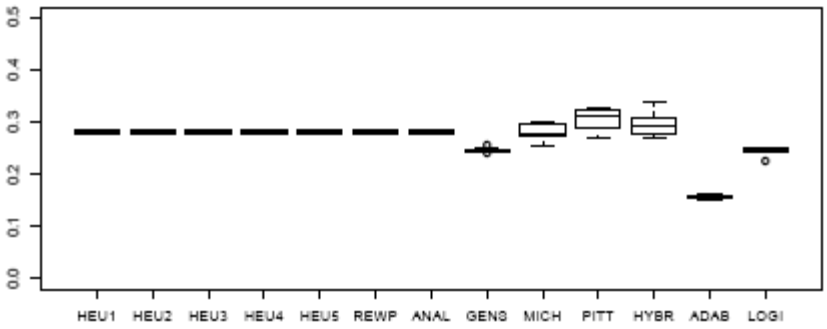


Fig 5.67. Errores en entrenamiento con partición optimizada

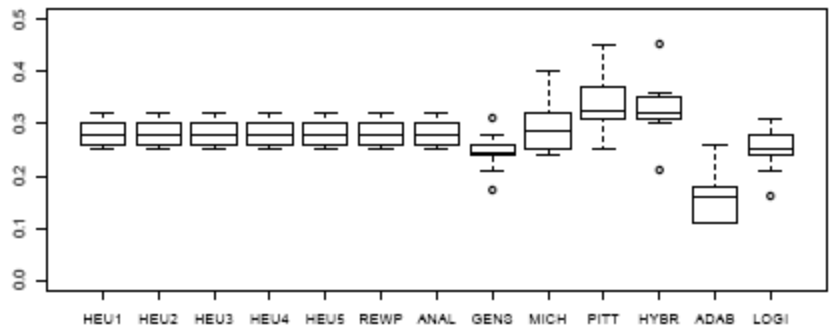


Fig 5.68. Errores en prueba con partición optimizada

5.6.2.2. Báscula bien calibrada

En este caso se trata de un conjunto de ejemplos con pesos obtenidos de una báscula bien calibrada donde el peso varía en un intervalo definido. Valores entre x-0.5 y x+0.5.

Existe una mejora en algunos casos no muy significativa en 9 de los 13 métodos en entrenamiento y en 10 de los 13 métodos en prueba. Se observa una uniformidad del error tanto en entrenamiento como en error después del ajuste genético.

	Uniforme		Optimizada	
	Entrenamiento	Prueba	Entrenamiento	Prueba
HEU1	0.43	0.47	**0.30**	**0.30**
HEU2	0.40	0.45	**0.30**	**0.30**
HEU3	0.31	0.36	**0.30**	**0.30**
HEU4	0.31	0.36	**0.30**	**0.30**
HEU5	0.31	0.36	**0.30**	**0.30**
REWP	**0.27**	0.31	0.29	**0.29**
ANAL	**0.26**	0.29	0.29	**0.29**
GENS	0.28	**0.29**	**0.32**	0.32
MICH	0.47	0.48	**0.30**	**0.30**
PITT	0.43	0.43	**0.30**	**0.30**
HYBR	0.42	0.43	**0.30**	**0.30**
ADAB	**0.12**	**0.12**	0.35	0.35
LOGI	**0.20**	**0.20**	0.27	0.28

Tabla 5.23 Experimentos con el dataset de pesos de una báscula bien calibrada

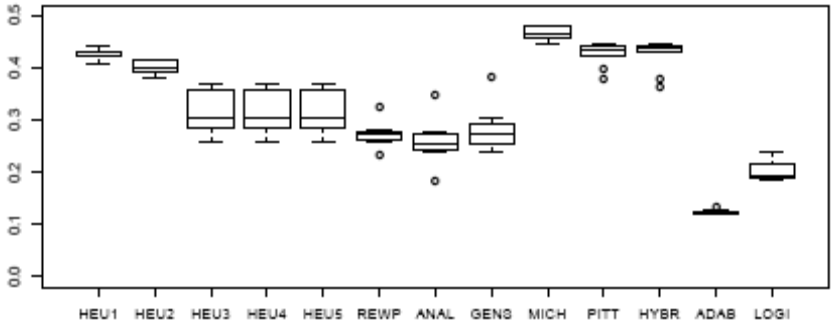


Fig 5.69. Errores en entrenamiento con partición uniforme

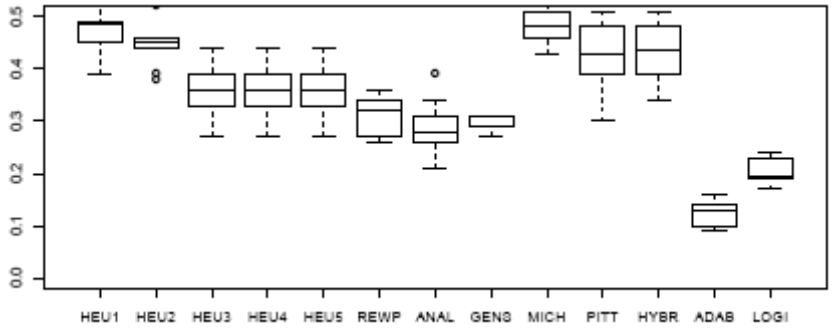


Fig 5.70. Errores en prueba con partición uniforme

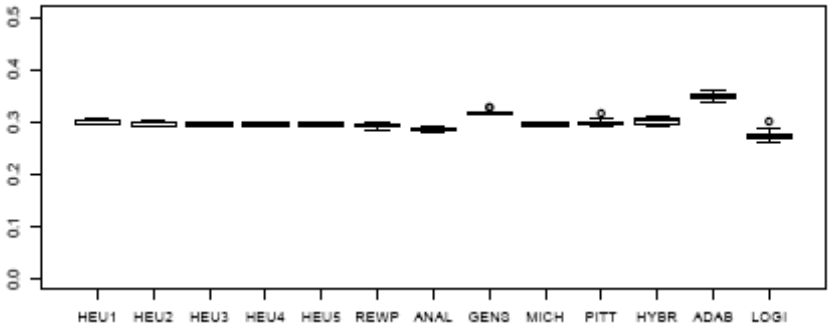


Fig 5.71. Errores en entrenamiento con partición optimizada

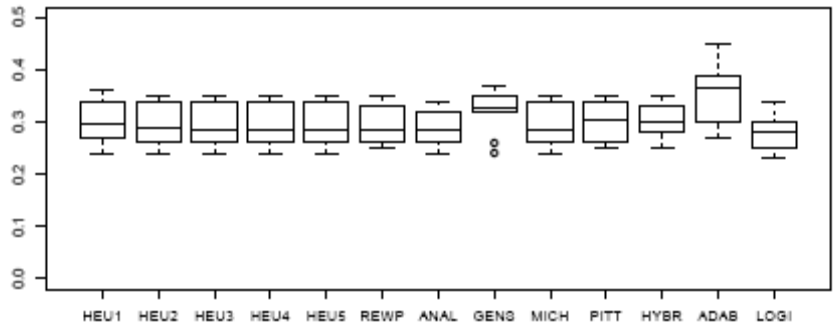


Fig 5.72. Errores en prueba con partición optimizada

5.6.2.3. Báscula mal calibrada

En este tipo de ejemplo el error en las mediciones no se puede predecir con exactitud como ocurría en el caso anterior ya que la báscula esta mal calibrada. Valores entre x-0.1 y x+0.9.

Se observa una mejora significativa en 10 de los 13 métodos tanto en entrenamiento como en prueba. Asimismo, se observa como el error se mantiene uniforme después del ajuste genético tanto en entrenamiento como en prueba.

	Uniforme		Optimizada	
	Entrenamiento	Prueba	Entrenamiento	Prueba
HEU1	0.42	0.45	**0.29**	**0.29**
HEU2	0.42	0.45	**0.29**	**0.29**
HEU3	0.35	0.39	**0.29**	**0.29**
HEU4	0.35	0.39	**0.29**	**0.29**
HEU5	0.35	0.39	**0.29**	**0.29**
REWP	0.35	0.38	**0.29**	**0.29**
ANAL	0.30	0.33	**0.29**	**0.29**
GENS	**0.28**	**0.29**	0.29	0.30
MICH	0.44	0.46	**0.29**	**0.29**
PITT	0.42	0.41	**0.29**	**0.29**
HYBR	0.42	0.42	**0.29**	**0.29**
ADAB	**0.22**	**0.24**	0.38	0.39
LOGI	**0.22**	**0.23**	0.29	0.29

Tabla 5.24 Experimentos con el dataset de pesos de una báscula mal calibrada

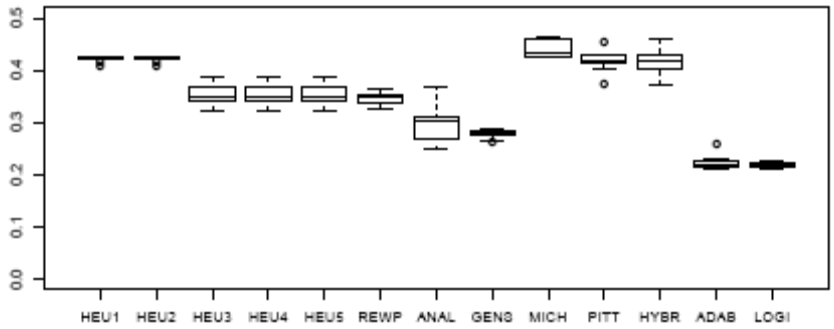


Fig 5.73. Errores en entrenamiento con partición uniforme

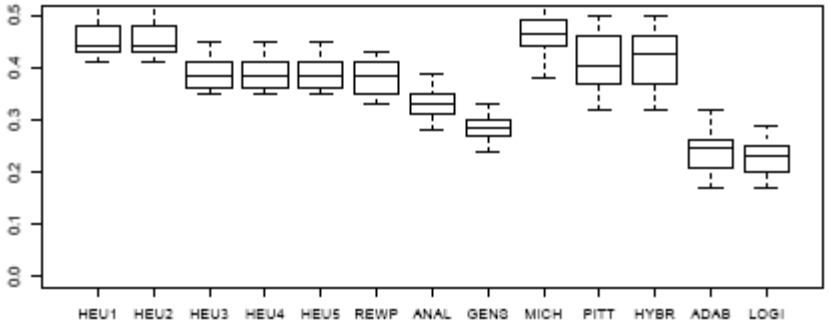


Fig 5.74. Errores en prueba con partición uniforme

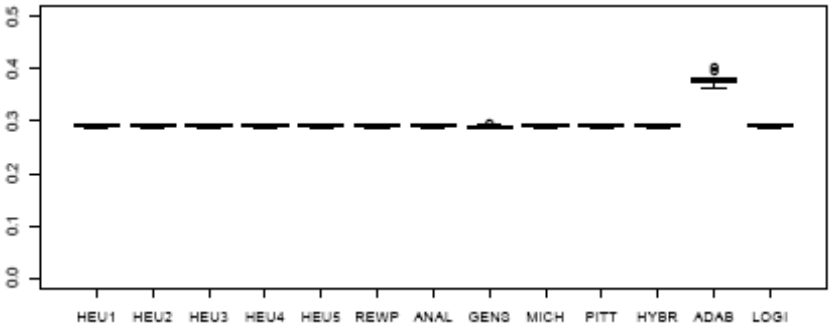


Fig 5.75. Errores en entrenamiento con partición optimizada

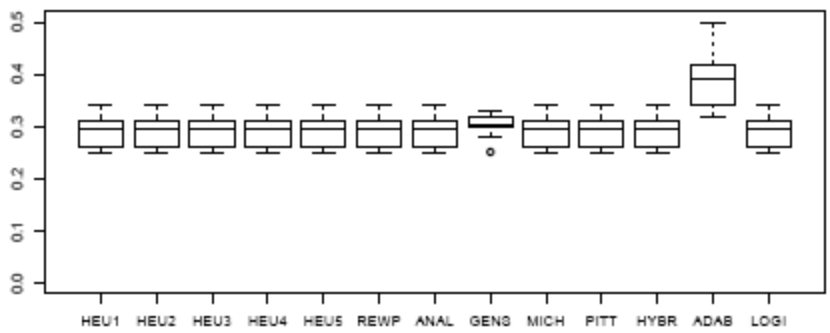


Fig 5.76. Errores en prueba con partición optimizada

5.6.2.4. Dos básculas diferentes

En el caso de ejemplos de mediciones de pesos de dos básculas diferentes, una bien calibrada y otra mal calibrada, se observa nuevamente una mejora significativa en 9 de los 13 métodos evaluados tanto en entrenamiento como en prueba. Curiosamente la mejora se produce en los mismos métodos tanto en prueba como en entrenamiento.

Como ocurría en los otros cosos de datos imprecisos, parece observarse cierta uniformidad en el error después de ajuste genético tanto en entrenamiento como en prueba.

	Uniforme		Optimizada	
	Entrenamiento	Prueba	Entrenamiento	Prueba
HEU1	0.43	0.47	**0.31**	**0.34**
HEU2	0.43	0.47	**0.31**	**0.34**
HEU3	0.33	0.36	**0.31**	**0.34**
HEU4	0.33	0.36	**0.31**	**0.34**
HEU5	0.33	0.36	**0.31**	**0.34**
REWP	**0.29**	**0.31**	0.33	0.36
ANAL	0.32	0.34	**0.31**	**0.34**
GENS	**0.25**	**0.26**	0.33	0.37
MICH	0.45	0.46	**0.35**	**0.36**
PITT	0.42	0.42	**0.37**	**0.37**
HYBR	0.43	0.43	**0.37**	**0.37**
ADAB	**0.20**	**0.21**	0.38	0.38
LOGI	**0.20**	**0.20**	0.30	0.31

Tabla 5.25 Experimentos con el dataset de pesos de dos básculas diferentes

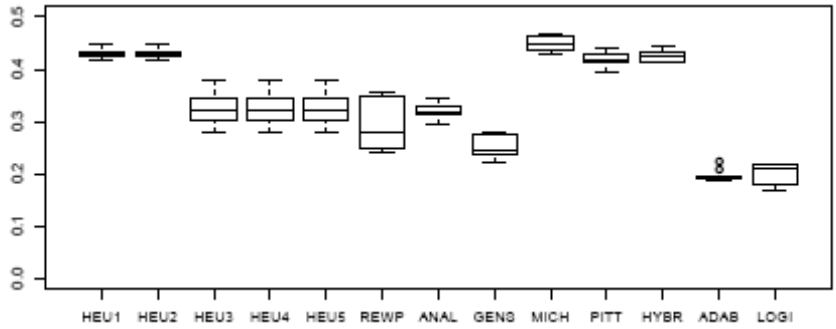


Fig 5.77. Errores en entrenamiento con partición uniforme

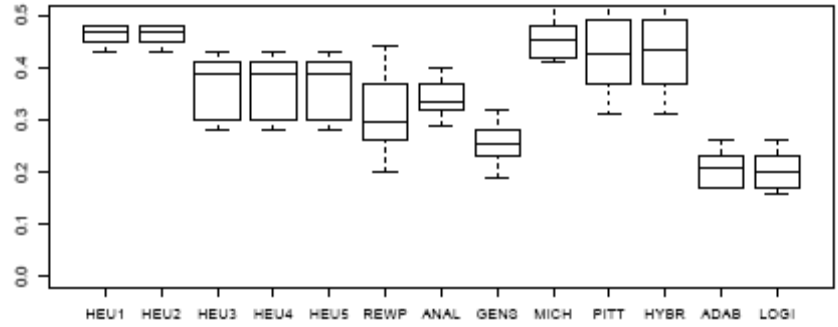


Fig 5.78. Errores en prueba con partición uniforme

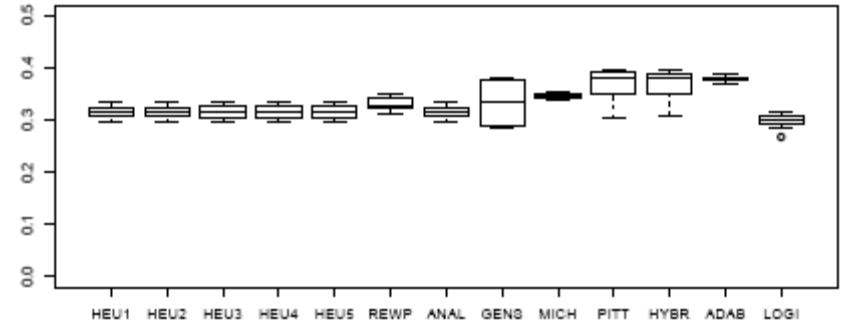


Fig 5.79. Errores en entrenamiento con partición optimizada

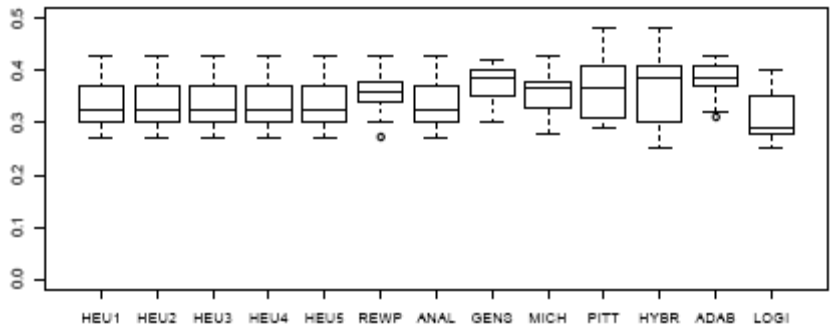


Fig 5.80. Errores en prueba con partición optimizada

5.6.2.5. Datos perdidos

En este caso se trata de un dataset donde el 5% de los datos son datos perdidos. Los valores perdidos se reemplazan por un conjunto borroso cuya función de pertenencia vale 1 en todas las etiquetas lingüísticas. En este caso observa que la mejora es significativa en todos los métodos tanto en entrenamiento como en prueba.

	Uniforme		Optimizada	
	Entrenamiento	Prueba	Entrenamiento	Prueba
HEU1	0.42	0.45	**0.24**	**0.24**
HEU2	0.40	0.43	**0.24**	**0.24**
HEU3	0.31	0.34	**0.24**	**0.24**
HEU4	0.31	0.34	**0.24**	**0.24**
HEU5	0.31	0.34	**0.24**	**0.24**
REWP	0.34	0.36	**0.24**	**0.24**
ANAL	0.25	0.27	**0.24**	**0.24**
GENS	0.30	0.32	**0.30**	**0.30**
MICH	0.47	0.48	**0.25**	**0.25**
PITT	0.43	0.43	**0.24**	**0.25**
HYBR	0.41	0.44	**0.26**	**0.26**
ADAB	0.31	0.31	**0.30**	**0.31**
LOGI	0.23	0.23	**0.20**	**0.20**

Tabla 5.26 Experimentos con el dataset de pesos de dos básculas diferentes

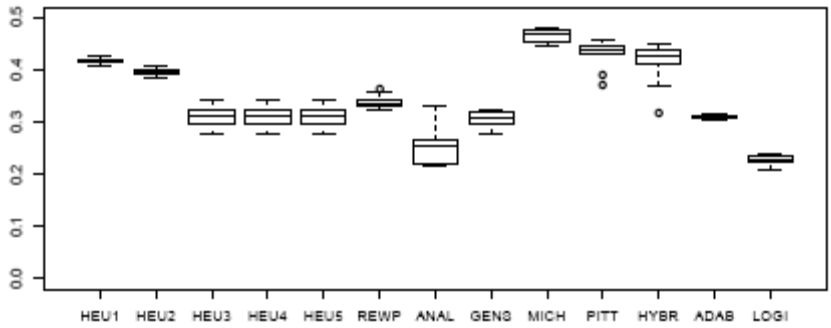


Fig 5.81. Errores en entrenamiento con partición uniforme

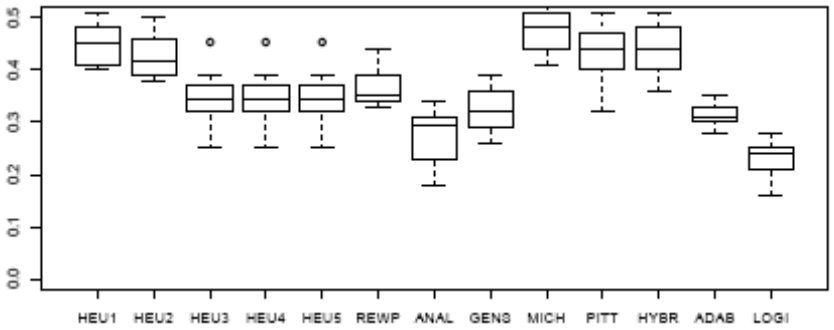


Fig 5.82. Errores en prueba con partición uniforme

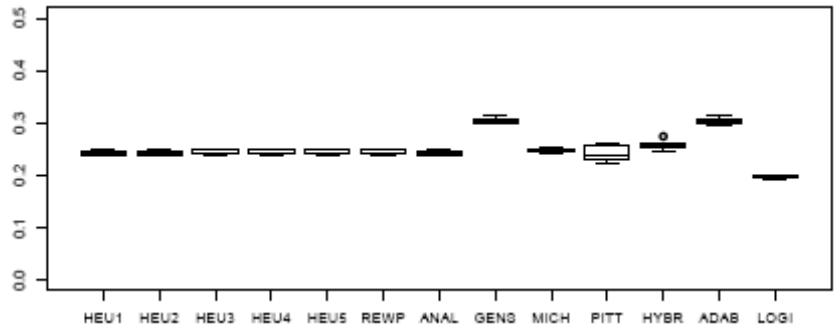


Fig 5.83. Errores en entrenamiento con partición optimizada

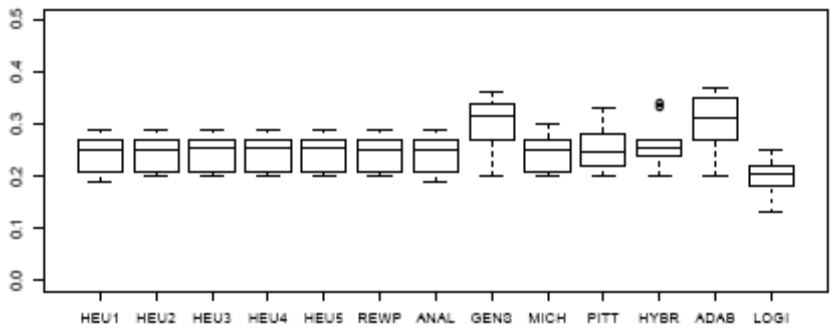


Fig 5.84. Errores en prueba con partición optimizada

5.7. Conclusiones finales

Se ha realizado un estudio experimental de dos tipos de datasets, aquellos que manejan datos sin imprecisión y aquellos que manejan datos incompletos o imprecisos. En ambos casos, se ha demostrado, que la utilización de la nueva definición de Información Mutua con aproximación bootstrap en un problema concreto que es la optimización de particiones lingüísticas, ha producido los resultados esperados y en cualquier caso coherentes. En algunos casos no se obtiene una mejora significativa pero si que se obtiene una uniformidad en los resultados.

CAPITULO 6

Comentarios finales

6.1. Introducción

Dedicaremos esta sección a la presentación de un resumen de los resultados obtenidos y conclusiones que esta memoria puede aportar. Presentaremos las publicaciones asociadas a esta tesis y comentaremos algunos aspectos sobre trabajos futuros que siguen la línea aquí expuesta y sobre otras líneas de investigación que se pueden derivar.

6.2. Resumen y conclusiones

Hemos definido una nueva medida de Información Mutua aplicada al diseño de particiones borrosas, que con la ayuda de un algoritmo genético diseñado para optimización de particiones, nos proporciona una partición optimizada a partir de una partición uniforme, tanto para datos sin imprecisión, como para datos con incompletos con imprecisión.

Hemos realizado un estudio de diferentes clasificadores, que nos van a servir como herramienta para demostrar que la nueva partición optimizada, mediante nuestro nuevo método, nos proporciona errores de clasificación iguales o mejores que los que nos proporcionaba la partición uniforme, en la mayor parte de los casos y en cualquier caso coherentes.

En general se observa una mejora de error medio cuando se realiza un ajuste genético de la partición uniforme de partida. La mejora, en algunos casos no es significativa, pero podemos demostrar en cualquier caso que nuestro método no empeora los resultados.

En el caso de conjuntos de ejemplos sin imprecisión, hemos seleccionado siete datasets utilizados comúnmente en procesos de clasificación. Se han considerado particiones uniformes de tamaño tres y el ajuste genético, mediante nuestra definición de Información Mutua, nos proporciona unos resultados que en general mejoran el error de clasificación en la mayor parte de los métodos. Se observan mejoras significativas en el error de la prueba de todos los algoritmos que utilizan el método de la regla ganadora, y se observa también que los algoritmos que usan la suma de votos dependen menos de la partición.

Por otra parte las diferencias entre los clasificadores borrosos dependen notablemente de la partición borrosa. La optimización genética iguala el error de los datasets de prueba y obtiene una uniformidad en

resultados entre los diferentes métodos de clasificación basados en reglas borrosas.

Si la imprecisión es añadida a datasets que no tenían imprecisión, los resultados obtenidos vienen a ser parecidos a los resultados de datasets sin imprecisión. La mejora fundamental se produce cuando los conjuntos de datos utilizados son problemas reales que conllevan imprecisión en si mismos. En estos casos la mejora del error medio es significativa y además se obtiene una uniformidad de error después del ajuste genético.

Se observa que una selección adecuada de la partición borrosa puede mejorar el comportamiento de la mayoría de los clasificadores bajo datos imprecisos, según lo considerado en los datasets empleados para datos imprecisos.

En el caso especial de valores perdidos se obtiene una mejora de todos los métodos. Pero en este caso hay que tener en cuenta que una cantidad masiva de valores perdidos puede incurrir en información no relevante para la clasificación.

Para finalizar se puede concluir que si bien es cierto que nuestro método no es universal y que dependiendo del problema será mas o menos viable su aplicación, si que podemos demostrar que esta nueva definición de Información Mutua propuesta mejora en algunos casos los sistemas de clasificación para los cuales esta indicada.

6.3. Publicaciones asociadas a la tesis

De la realización de esta tesis se ha conseguido realizar dos publicaciones.

- Sánchez, L., Suárez, M.R. y Couso, I. A fuzzy definition of Mutual Information with application to the design of Genetic Fuzzy Classifiers. International Conference on Machine Intelligence (ACIDCA-ICMI05). Tozeur (Tunisia, 2005) 602-609

- Sánchez, L., Suárez, M.R. y Couso, I. Possibilistic mutual information between fuzzy random variables applied to the genetic estimation of the most informative fuzzy partitions from incomplete data.

6.4. Líneas de investigación futuras

A continuación, consideraremos algunas líneas de trabajo futuras a partir de las conclusiones obtenidas en esta memoria.

La definición de Información Mutua aportada tiene más aplicaciones que la que se plantea en este documento, relacionada con la obtención de la partición optima. Una de las líneas de investigación que se pretende llevar a cabo es la aplicación de la definición de Información Mutua a la selección de características lingüísticas, tema del que hay muchos estudios con los que tendremos que compararnos. Se pretende aplicar la nueva definición tanto para selección de características con datos sin imprecisión como con datos incompletos o con imprecisión.

Por otra parte también se puede realizar un estudio relacionado con la forma en que se mejora la selección de características cuando la partición resultante es una partición optimizada y no uniforme tanto para datos sin imprecisión como para datos con imprecisión.

APENDICES

APENDICE A

A.1. α-corte

Un α-corte es una forma directa de pasar de conjuntos borrosos a conjuntos clásicos [KY95]. Dado un número α perteneciente a [0,1] y un conjunto borroso A, definimos el α-corte de A con el conjunto A_α cuya función característica se define como:

$$\gamma A_\alpha(x) = \begin{cases} 1 & si \mu_A(x) \geq \alpha \\ 0 & en\_otro\_caso \end{cases}$$

En definitiva, el α-corte se compone de aquellos elementos cuyo grado de pertenencia es mayor o igual a un umbral dado α.

Y así, se puede hablar de α-cortes estrictos si

$$\gamma A_\alpha(x) = \begin{cases} 1 & si \mu_A(x) > \alpha \\ 0 & en\_otro\_caso \end{cases}$$

Cualquier conjunto borroso A se puede representar mediante la unión de sus α-cortes de la siguiente manera:

$$\mu_A(x) = \max_{\alpha \in [0,1]} [\alpha . \gamma A_\alpha(x)]$$

A.2. Números borrosos

Un número borroso es una cantidad cuyo valor es impreciso, no como en el caso de los números ordinarios cuyo valor es exacto, ya que es un único valor.

Cualquier número borroso se puede interpretar como una función cuyo dominio sea un sistema o conjunto especificado, normalmente el conjunto de los números reales, cuyo rango va de 0 a 1 sin incluir los números negativos.

A cada valor numérico, en el dominio, se le asigna un grado de pertenencia al mismo, donde 0 representa el grado de pertenencia mas pequeño y 1 es el grado pertenencia mayor.

En muchos aspectos, los números borrosos representan la realidad, de forma más fidedigna, que los números clásicos u ordinarios. Supongamos, por ejemplo, que vamos conduciendo a lo largo de una carretera donde el límite de velocidad es 120 km/h. Intentamos mantener la velocidad a 120 km/h, pero el coche no tiene velocidad de crucero o control de velocidad y esta varía constantemente. Si se representa la velocidad de forma gráfica para esos instantes en los que varia y durante varios minutos y además se traza una línea que une esos puntos representados, se conseguirá una función que se parecerá a alguna de las curvas que se muestran mas abajo.

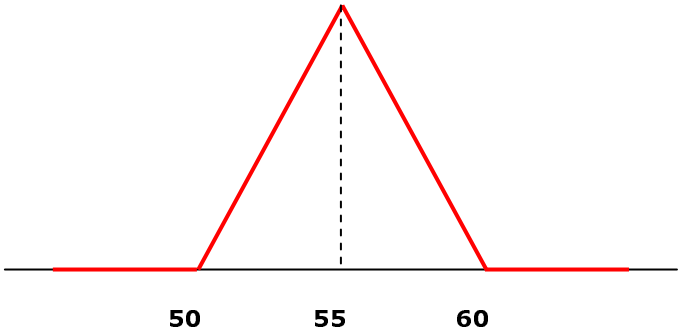


Fig A.1. Número borroso triangular

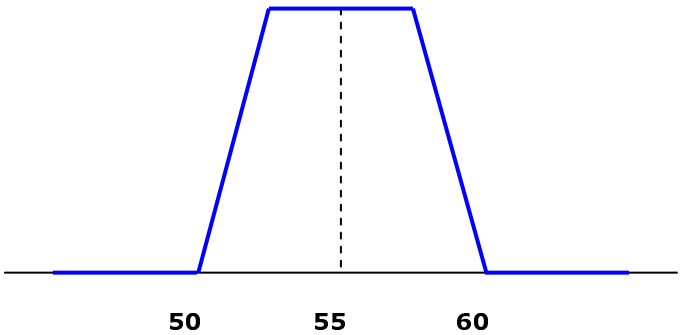


Fig A.2. Número borroso trapezoidal

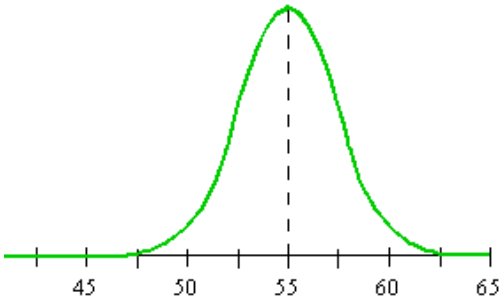


Fig A.3. Número borroso en forma de campana

La curva roja, representa un número borroso triangular, la curva azul muestra un número borroso trapezoidal y la curva verde muestra un número borroso en forma de campana.

Estas tres funciones, conocidas como funciones de pertenencia, son todas ellas convexas (comienza en 0, alcanzan el máximo, y luego vuelven a tender a cero). Sin embargo, algunos números borrosos tienen funciones de pertenencia cóncava, irregular e incluso caótica.

No hay restricciones en la forma de la curva, siempre y cuando cada valor en el dominio se corresponda con uno y sólo un grado de pertenencia, y este grado de partencia nunca sea menor que 0 ni mayor que 1.

Los números borrosos se utilizan en estadística, en programación, en ingeniería y en ciencias experimentales.

En definitiva, la realidad nos dice que todos los fenómenos en el universo físico tienen un grado de incertidumbre inherente.

APENDICE B

B.1. Definición de información mutua

Un clasificador se puede considerar como un sistema que reduce la incertidumbre inicial, para ser definido de forma más precisa en un momento posterior mediante la información contenida en el vector de entrada.

En el caso ideal la incertidumbre final será cero pero en las aplicaciones del mundo real la incertidumbre final será alta por al menos dos razones distintas:

- tenemos una información de partida escasa o insuficiente
- la forma de trabajar del clasificador no es la óptima.

En el segundo caso la información de la que se dispone puede ser suficiente para resolver ambigüedades pero se pierde información debido a que:

- hay poco entrenamiento,
- a que los datos con aproximados,
- e incluso a errores.

Esto último se puede resolver considerando ejemplos adicionales de entrenamiento, mas tiempo de entrenamiento e incluso considerando otros algoritmos de procesamiento, pero la carencia de de información debe ser detectada lo antes posible en el proceso de desarrollo, porque en este caso, la única solución es añadir mas características, cuanto mas informativas mejor.

La teoría de la información de Shannon [SW49] propone una formulación adecuada para cuantificar los conceptos expuestos con anterioridad.

Si las probabilidades para las distintas clases son $P_c(c)$ con $c = 1, \cdots, N_c$, la incertidumbre inicial en la clase de salida se puede medir mediante la entropía de Shannon [Ash65]:

$$H(C) = -\sum_{c=1}^{N_c} P_c(c) \log P_c(c)$$

mientras que la incertidumbre media después de conocer el vector de características f con $f = 1, \cdots, N_f$, es la entropía condicionada:

$$H(C/F) = -\sum_{f=1}^{N_f} P_f(f) \left(\sum_{c=1}^{N_c} P_{c,f}(c/f) \log P_{c,f}(c/f) \right)$$

donde $P_{c,f}(c/f)$ es la probabilidad condicionada para la clase c, dado el vector f.

Si el vector de características esta constituido por variables continuas, la suma se sustituirá por una integral y las probabilidades por las densidades de probabilidad correspondientes.

Por lo general, la entropía condicionada será menor o igual que la entropía inicial, debido a la información que una variable aporta a la otra. Es igual, si y solo si hay independencia entre las características y la clase, es decir, si la densidad de probabilidad conjunta es el producto de las densidades individuales.

$$P_{c,f}(c, f) = P_c(c) P_f(f)$$

En definitiva, la cantidad de información aportada por la cual se decrementa la incertidumbre es por definición, la **información mutua** entre las variables y la clase

$$I(C; F) = H(C) - H(C/F)$$

Esta función es simétrica con respecto a C y F, y se puede transformar de la siguiente manera:

$$I(C; F) = I(F; C) = \sum_{c,f} P_{c,f}(c, f) \log \frac{P_{c,f}(c, f)}{P_c(c) P_f(f)}$$

La información mutua es, por lo tanto, la cantidad de información aportada mediante la cual el conocimiento suministrado por el vector de características decrementa la incertidumbre sobre la clase. Por lo general, la incertidumbre condiciona es menor que la suma de las incertidumbres individuales y se podría demostrar la siguiente relación:

$$H(C;F) = H(C) + H(F) - I(C,F)$$

o bien:

$$I(C,F) = H(C) + H(F) - H(C;F)$$

Una representación gráfica que ilustra lo que hemos explicado y que muestra la relación entre la entropía de dos variables aleatorias sería la siguiente:

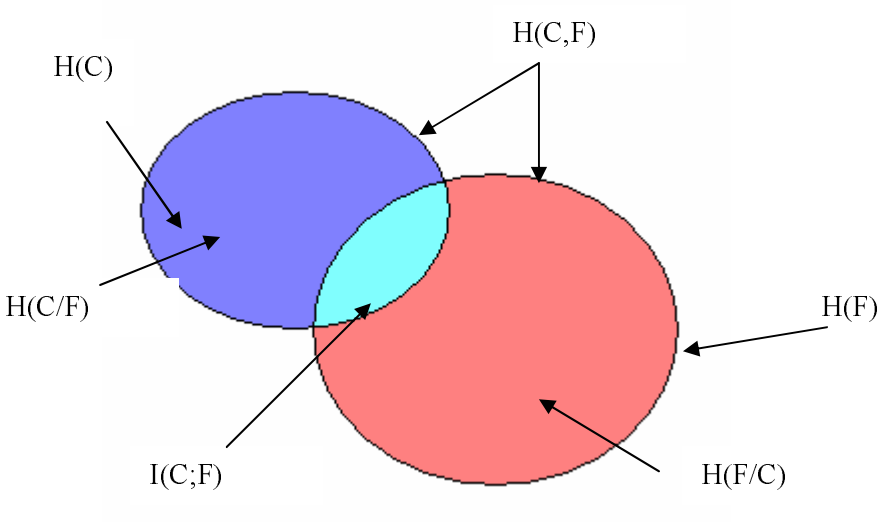


Fig B.1. Representación gráfica de la entropía e IM

Y se puede deducir que:

- Si C y F son variables aleatorias independientes, la incertidumbre de una no depende de la incertidumbre de la otra. Por lo tanto, $I(C;F) = 0$.

- En términos generales, el condicionamiento de una variable sobre la otra quita incertidumbre de esta variable. Es decir, que $H(C/F) \leq H(C)$, y también $H(F/C) \leq H(F)$.

Aunque la principal aplicación de la Teoria de la Información esta relacionada con el ruido producido en los canales de comunicación, este concepto se ha aplicado a distintos campos. Así por ejemplo veremos su aplicación en problemas como la selección de características lingüísticas o bien selección de la mejor partición borrosa para el diseño de un clasificador que sea optimo.

B.1.1. Información mutua entre variables discretas aleatorias

Para ilustrar esta sección utilizaremos un pequeño ejemplo mediante el cual se podrá entender todo el proceso.

Supongamos que disponemos de cinco ejemplos relacionados con pesos de frutas. Cada uno de los ejemplos tiene un atributo que llamaremos "peso"; vamos a tener pesos de frutas.

Suponemos además que dicho atributo continuo va a tener tres etiquetas lingüísticas, "pequeño", "mediano" y "grande". A cada una de esas etiquetas lingüísticas se le puede asignar un intervalo que represente un conjunto de valores y dicho intervalo vendrá dado por un experto. A cada ejemplo le corresponderá una única etiqueta lingüística ya que no hay solapamientos. En nuestro ejemplo presentamos dos posibles alternativas

1) pequeño=[90,100), mediano=[100,110), grande=[110,120]

y

2) pequeño=[90,110), mediano=[110,115), grande=[115,120]

que dependerán de los datos de entrada para el atributo en estudio.

Tenemos que discriminar entre tres clases: manzana, pera y plátano. Para diseñar un clasificador basado en reglas, utilizaremos un ejemplo de cinco piezas, cuyos pesos y clases se muestran en la Tabla B.1.

Ejemplo	Peso	Clase
1	111	Pera
2	96	Manzana
3	116	Pera
4	91	Plátano
5	101	Manzana

Tabla B.1. Muestra de 5 ejemplos

Si utilizamos las etiquetas lingüísticas (discretización) para representar los datos numéricos correspondientes al atributo "peso" para cada una de las alternativas, se obtiene lo que se presenta en la tabla2.

Ejemplo	Peso	Alternativa1	Alternativa2	Clase
1	111	Grande	Mediano	Pera
2	96	Pequeño	Pequeño	Manzana
3	116	Grande	Grande	Pera
4	91	Pequeño	Pequeño	Plátano
5	101	Mediano	Pequeño	Manzana

Tabla B.2. Muestra de 5 ejemplos discretizados

El objetivo final será escoger el mejor diseño entre dos alternativas que se han propuesto.

Las entropías de las variables "Alternativa1", "Alternativa2" y "Clase" son respectivamente,

$$H_1 = 0.4\log 0.4 + 0.2\log 0.2 + 0.4\log 0.4 = 1.0549$$
$$H_2 = 0.9503$$
$$H_{Clase} = 1.0549$$
$$H_{1,Clase} = 1.3322$$
$$H_{2,Clase} = 1.3322$$

Con las entropías calculadas se puede aplicar la formula de información mutua propuesta en el apartados anteriores y se obtienen los siguientes resultados,

$$\boxed{IM_{1,Clase} = H_1 + H_{Clase} - H_{1,Clase} = 0.7776}$$

$$IM_{2,Clase} = H_2 + H_{Clase} - H_{2,Clase} = 0.6730$$

que nos llevan a concluir que la primera alternativa aporta más información del atributo peso sobre la clase y por lo tanto es la seleccionada para el problema propuesto.

B.1.2. Información mutua entre una variable aleatoria y un conjunto aleatorio

Supongamos de nuevo, que disponemos de cinco ejemplos relacionados con pesos de frutas. Cada uno de los ejemplos tiene un atributo que llamaremos "peso" y como en el caso anterior el atributo continuo va a tener tres etiquetas lingüísticas, "pequeño", "mediano" y "grande",

En este caso no nos dan el valor del estado del sistema, pero si un conjunto que contiene este valor, es decir, para cada ejemplo no nos proporciona un peso concreto, sino una aproximación (ver tabla B.3). En otras palabras, tenemos una tolerancia en las medidas numéricas, y además cuando el valor del estado está cerca de los límites de un elemento de la partición no podemos asignarle una única etiqueta, sino dos.

Ejemplo	Peso	Clase
1	110±1	Pera
2	96±1	Manzana
3	116±1	Pera
4	91±1	Plátano
5	101±1	Manzana

Tabla B.3. Muestra de 5 ejemplos con imprecisión

Además, no conocemos los valores de las frecuencias, pero podemos obtener los conjuntos de las contienen. No podemos estimar la información mutua para un ejemplo pero si podemos encontrar un conjunto que la contiene:

$$\bar{MI}(A,B) = \{\Sigma a_i \log a_i + \Sigma b_i \log b_i - \Sigma\Sigma c_i \log c_i \ : \ a_i \in \Gamma_{a_i}, b_i \in \Gamma_{b_i}, c_i \in \Gamma_{c_i}\}$$

Este conjunto se formará por todas las estimaciones de información mutua compatible con nuestro conocimiento.

Dadas las dos alternativas propuestas por el experto humano,

1) pequeño=[90,100), mediano=[100,110), grande=[110,120]

y

2) pequeño=[90,110), mediano=[110,115), grande=[115,120]

y como en el caso anterior, podemos construir una Tabla Bon la correspondencia entre valor numérico y etiqueta lingüística para cada una de las alternativas como se muestra en la Tabla 5.

Ejemplo	Peso	Alternativa1	Alternativa2	Clase
1	110±1	Grande o Mediano	Pequeño o Mediano	Pera
2	96±1	Pequeño	Pequeño	Manzana
3	116±1	Grande	Grande	Pera
4	91±1	Pequeño	Pequeño	Plátano
5	101±1	Mediano	Pequeño	Manzana

Tabla B.4. Muestra de 5 ejemplos con imprecisión discretizados

El objetivo será final será escoger el mejor diseño entre dos que se han propuesto. En cada una de las alternativas hay dos conjuntos de estimaciones que son compatibles con ejemplo.

Alternativa 1

pequeño=[90,100), mediano=[100,110), grande=[110,120]

Alternativa1				
Ejemplo	Peso	Estimación1	Estimación2	Clase
1	110±1	Grande	Mediano	Pera
2	96±1	Pequeño	Pequeño	Manzana
3	116±1	Grande	Grande	Pera
4	91±1	Pequeño	Pequeño	Plátano
5	101±1	Mediano	Mediano	Manzana

Tabla B.5. Muestra de 5 ejemplos con imprecisión discretizados. Alternativa 1

$$H_{E1} = 0.4\log 0.4 + 0.2\log 0.2 + 0.4\log 0.4 = 1.0549$$
$$H_{E2} = 1.0549$$
$$H_{Clase} = 1.0549$$
$$H_{E1,Clase} = 1.3322$$
$$H_{E2,Clase} = 1.6094$$

Con las entropías calculadas se puede aplicar la formula de información mutua propuesta en los apartados anteriores y se obtienen los siguientes resultados,

$$IM_{E1,Clase} = H_{E1} + H_{Clase} - H_{E1,Clase} = 0.7776$$

$$IM_{E2,Clase} = H_{E2} + H_{Clase} - H_{E2,Clase} = 0.5004$$

Alternativa2

pequeño=[90,110), mediano=[110,115), grande=[115,120]

Nuevamente se presentar se pueden realizar dos estimaciones posibles:

Alternativa2				
Ejemplo	**Peso**	**Estimación1**	**Estimación2**	**Clase**
1	110±1	Pequeño	Mediano	Pera
2	96±1	Pequeño	Pequeño	Manzana
3	116±1	Grande	Grande	Pera
4	91±1	Pequeño	Pequeño	Plátano
5	101±1	Pequeño	Pequeño	Manzana

Tabla B.6. Muestra de 5 ejemplos con imprecisión discretizados. Alternativa 2

$$H_{E1} = 0.8\log 0.8 + 0.2\log 0.2 = 0.5004$$
$$H_{E2} = 0.9503$$
$$H_{Clase} = 1.0549$$
$$H_{E1,Clase} = 1.3322$$
$$H_{E2,Clase} = 1.3322$$

Con las entropías calculadas se puede aplicar la formula de información mutua propuesta en el apartados anteriores y se obtienen los siguientes resultados,

$$IM_{E1,Clase} = H_{E1} + H_{Clase} - H_{E1,Clase} = 0.2231$$

$$IM_{E2,Clase} = H_{E2} + H_{Clase} - H_{E2,Clase} = 0.6730$$

Operando con la primera alternativa, la información mutua es 0.7776, en cambio la segunda alternativa da un valor de 0.6730, además podemos decir que la cantidad de información mutua que la entrada discreta aporta a la clase esta en el conjunto $\overline{IM} = \{0.6730, 0.7776\}$. Hay que tener en cuenta que si el segundo ejemplo estuviese en los límites, obtendríamos cuatro conjuntos de estimaciones y así sucesivamente.

B.1.3. Información mutua entre una variable aleatoria y una variable aleatoria difusa.

Recordamos que en la sección anterior estimamos un conjunto de valores de información mutua dando etiquetas a los ejemplos, esto es, "el peso es grande o mediano". Ahora queremos generalizarlo al caso borroso, donde los valores numéricos son transformados en un conjunto borroso de etiquetas de la forma, "el peso es grande con seguridad 0.8" y "el peso es mediano con seguridad 0.2".

En este último caso según [SSC05], todo lo que queramos estimar sobre información mutua sufre una restricción borrosa. Para cada α-corte se obtiene un problema como el de la sección anterior, cuya solución se puede apilar para formar una salida borrosa. El proceso se muestra a continuación.

Vamos a evaluar la información mutua entre la variable lingüística "peso" (entendiendo "variable lingüística" en el sentido borroso), como se define en la Fig B.2., y la variable discreta "clase".

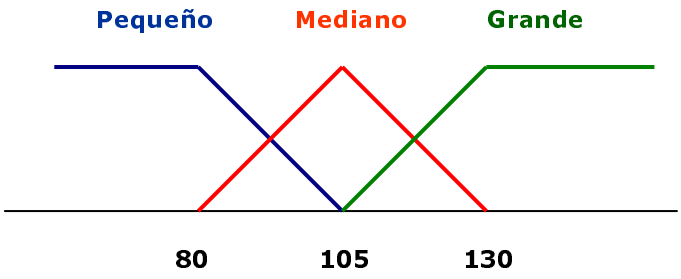


Fig B.2. Conjunto borroso "peso"

Para nuestro problema, suponemos cinco ejemplos con pesos de frutas y la clase a la que pertenece.

Ejemplo	Valor	Clase
1	108	Pera
2	100.5	Manzana
3	110.5	Pera
4	98	Plátano
5	103	Manzana

Tabla B.7. Muestra de 5 ejemplos

Para cada ejemplo tenemos un valor de la función de pertenencia. Cada valor del atributo peso pertenecerá con un grado a lo sumo a dos particiones siempre y cuando el valor no coincida con la pertenencia máxima, en cuyo caso pertenece a un partición con grado máximo.

Ejemplo	Valor	Pequeño	Mediano	Grande	Clase
1	108	0	0.88	0.12	Pera
2	100.5	0.18	0.82	0	Manzana
3	110.5	0	0.78	0.22	Pera
4	98	0.28	0.72	0	Plátano
5	103	0.08	0.92	0	Manzana

Tabla B.8. Ejemplos con las pertenencias a cada variables lingüística del conjunto borroso

Como un valor del atributo puede pertenecer como máximo a dos particiones hay tantas combinaciones como dos elevado al número de ejemplos del problema a tratar. En este caso hay 2^5=32 combinaciones distintas de las etiquetas discretas “pequeño”, “mediano” y “grande” que son compatibles con esos valores. Por ejemplo una de ellas sería la siguiente:

Ejemplo	Valor	Variable lingüística	Pertenencia	Clase
1	108	Mediano	0.88	Pera
2	100.5	Pequeño	0.18	Manzana
3	110.5	Mediano	0.78	Pera
4	98	Pequeño	0.28	Plátano
5	103	Pequeño	0.08	Manzana

Tabla B.9. Una de las alternativas con sus pertenencias

para los cuales el valor de información mutua entre la clase y la variable lingüística es 0.673, y la pertenencia del valor 0.673 a la información mutua borrosa no será menor de 0.08 (mínimo α-corte).

Repitiendo el proceso para cada una de las 32 combinaciones posibles, se obtiene 32 valores de información mutua junto con su pertenencia mínima. Agrupamos los valores de mínimos α-cortes, de tal forma que para cada valor distinto de Información Mutua tenemos distintos valores de α-corte. Para cada valor distinto nos quedamos con el máximo α-corte y lo representamos como la suma de α-corte/IM:

Estimación=0.72/0+0.22/0.22+0.18/0.40+0.28/0.50+
0.22/0.67+0.12/0.78+ 0.12/1.06.

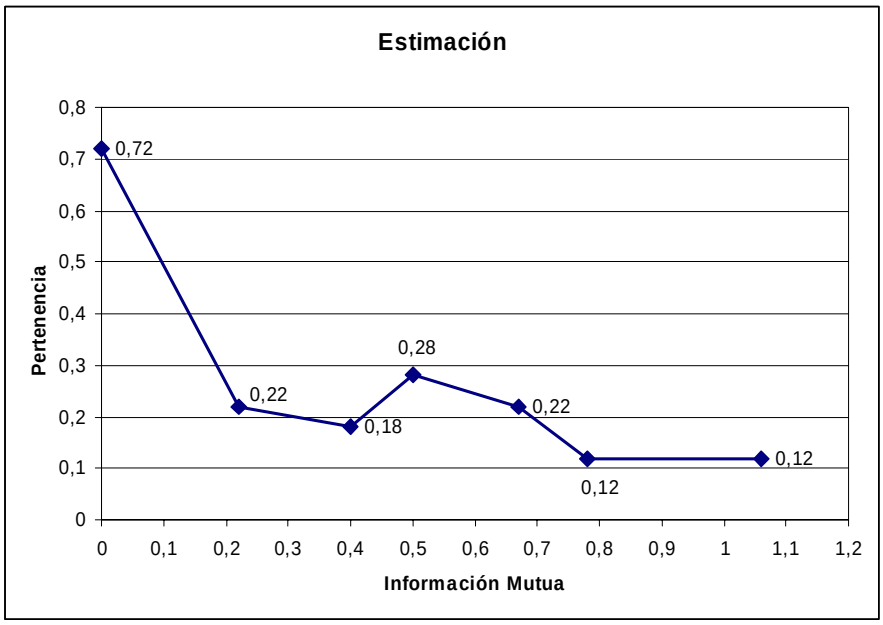


Fig B.3. Relación entre pertenencia e IM de forma gráfica

Distintas particiones tendrán distintos valores de información mutua fuzzy. Por ejemplos, si el conjunto fuzzy "pequeño", "mediano" y "grande" se definen respectivamente como

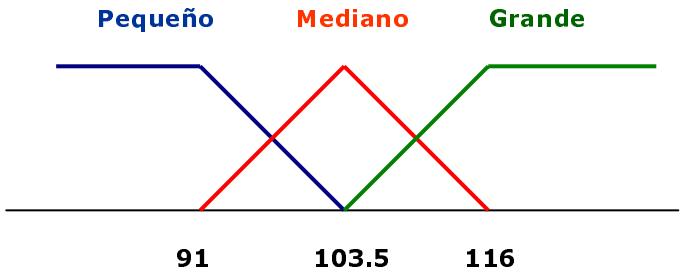


Fig B.4. Conjunto borroso "peso": nueva definición

la estimación es:

Estimación=0.40/0.50+0.40/0.67+0.60/0.78+0.40/1.05

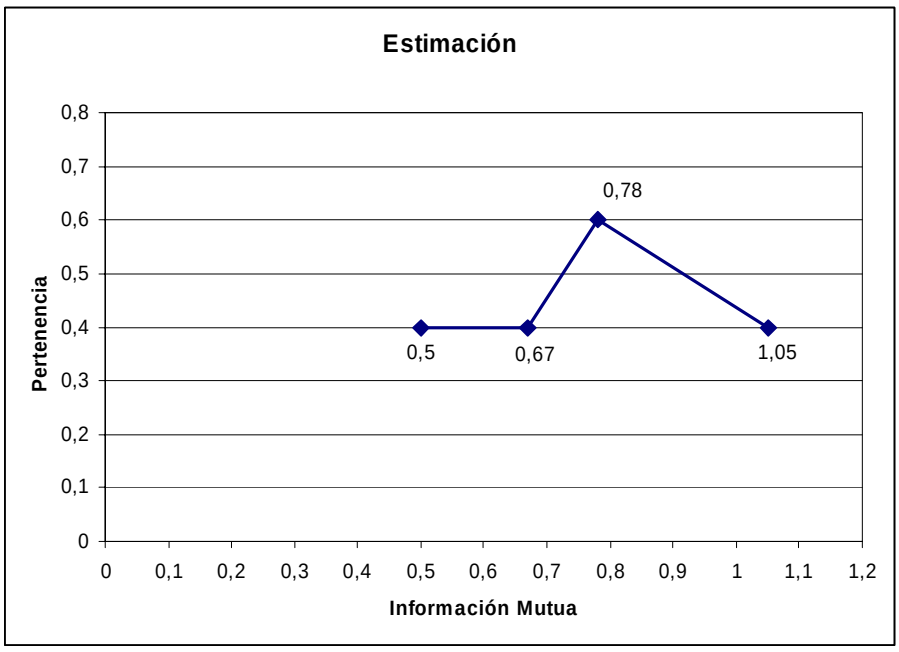


Fig B.5. Relación entre pertenencia e IM de forma gráfica para el segundo conjunto borroso

Lo que nos falta ahora es determinar cual de las dos particiones es la óptima. Para ello existen técnicas que nos permiten comparar dos números borrosos. Nosotros utilizamos la técnica del centroide:

$$\frac{\sum \mu(x_i)x}{\sum x_i}$$

la cual nos dice que la segunda definición dada de las particiones es mejor.

Hay que recalcar que, si nos hubiésemos restringido a la combinación mas compatible de las 32,

Ejemplo	Valor	Variable lingüística	Pertenencia	Clase
1	108	Mediano	0.88	Pera
2	100.5	Mediano	0.82	Manzana
3	110.5	Mediano	0.78	Pera
4	98	Mediano	0.72	Plátano
5	103	Mediano	0.92	Manzana

Tabla B.10. Una de las alternativas con sus pertenencias

la información mutua para este caso puntual es 0. Así, si hubiésemos tomado este caso para estimar la información mutua, hubiésemos concluido (erróneamente) que la partición lingüística pierde completamente la dependencia con respecto a la clase numérica.

Es un sentido informal, los valores de información mutua con pertenencias mas bajas informan sobre la cantidad de información aportada por la entrada que es menos cubierta por la partición borrosa, las cuales pueden ser relevantes para el diseño de un clasificador.

APENDICE C

Una de las experimentaciones realizadas fue con un fichero con 20 ejemplos y tres variables. Cada variable es un número real comprendido entre 0 y 1 que se genera de forma aleatoria. La clase tiene dos valores posibles 0 y 1.

Se realiza una comparación entre la densidad estimada para la primera variable del problema con la información mutua basada en conjuntos aleatorios, y las estimaciones bootstrap con 5, 8, 10 y 12 puntos (500 remuestras).

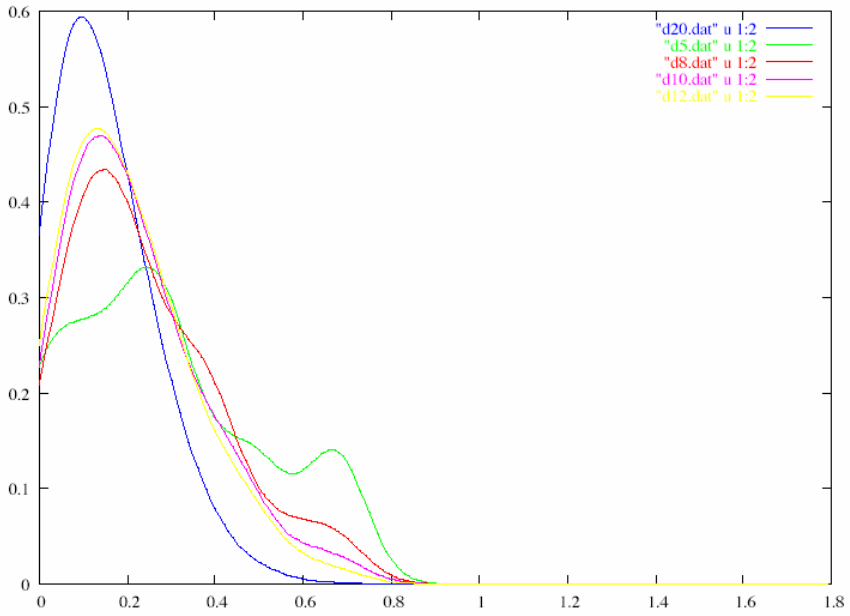


Fig C.1. Densidad estimada para distintos tamaños de conjuntos

La estimación bootstrap de la densidad no parece converger al valor deseado (las curvas con 10 y 12 puntos están muy próximas entre sí y lejanas de la curva con los 20 ejemplos).

Se realiza también estimaciones Monte Carlo de la función de densidad de la IM para la variable 1 del problema (con 20 ejemplos), con 1000, 10000 y 20000 iteraciones.

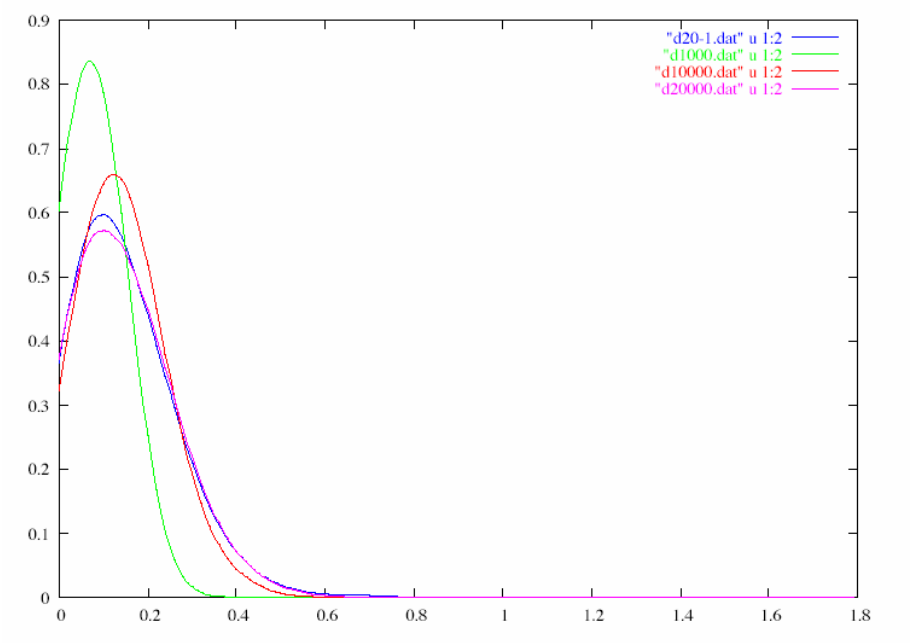


Fig C.2. Estimación Monte Carlo con distinto número de iteraciones

La aproximación de 20000 iteraciones es casi idéntica a la estimación completa y lleva 10 veces menos cálculos.

Bibliografía

[ACW06] Au, W.-H., Chan, K. C. C., Wong, A. K. C. A Fuzzy Approach to Partitioning Continuous Attributes for Classification. *IEEE Transactions On Knowledge And Data Engineering, vol. 18, no. 5.*

[AMSTV96] Agrawal, R., Mannila, H., Srikant, R., Toivonen, H. y Verkamo, A.I. Fast Discovery of Association Rules. *In Fayyad, U. M., Piatetsky –Shapiro G., Smyth P., Uthurusamy R. eds., Advances in knowledge Discovery & Data Mining, 307-328. AAAI Press, Menlo Park.*

[AS94] Agrawal, R., Srikant, R. Fast algorithms for Mining Association Rules. *In Proc. of 20th International Conference on Very Large Data Bases*, 487-499.

[Ash65] Ash, R. Information Theory. *Dover.*

[Bat94] Battiti, R. Using Mutual Information for Selecting Features in Supervised Neural Net Learning. *IEEE Transation on Neural networks, vol. 5 no. 4.*

[Bil95] Bilgic¸, T. Measurement Theoretic Frameworks for Fuzzy Set Theory with Applications to Preference Modelling, *PhD thesis, University of Toronto, Dept. of Industrial Engineering Toronto Ontario M5S 1A4 Canada.*

[BKC96] Bonissone, P. P., Khedkar, P. S. y Chen, Y. Genetic algorithms for automated tuning of fuzzy controllers: A transportation application. *In Proc. Fifth IEEE International Conference of Fuzzy Systems (FUZZ-IEEE'96), New Orleans, USA, pp. 674-680.*

[Boo82] Booker, L. B. Intelligent behaviour as an adaptation to the task environment. *Ph. D. thesis, Department of Computer and Communication Science. University of Michigan.*

[BS91] Bäck, T. y Schwefel, H. P. Extended selection mechanisms in genetic algorithms. *In Proc. Forth International Conference on Genetic Algorithms (ICGA'91), páginas 2-9. San Diego, EE.UU.*

[BWY93] Bollmann-Sdorra, P., Wong, S. K. M. & Yao, Y. Y. A measurement theoretic axiomatization of fuzzy sets, *Fuzzy Sets and Systems 60(3): 295–307.*

[CHGP98] Cordon, O., Herrera, F., González, A. y Pérez, R. Encouraging cooperation in the genetic iterative rule learning approach for qualitative modelling. *In J. Kacprzyk and L. A. Zadeh (Eds.), Computing with Words in Intelligent/Information Systems 2, pp.95-117. Physica-Verlag.*

[CHHM01] Cordon, O., Herrera, Hoffmann, F. y Magdalena, L. Genetic fuzzy systems: evolutionary tuning and learning of fuzzy knowledge bases. *World Scientific Publishers, Singapore.*

[CJH99] Cordon, O., Jesus, M.J. del, Herrera, F. A proposal on Reasoning Methods in fuzzy rule-base classification systems. *International Journal of approximate reasoning, 20, 21-45.*

[CJHL99] Cordon, O., Jesus, M. J. del, Herrera, F. y Lozano, M. MOGUL: A methodology to obtain genetic fuzzy rule-based systems under the iterative rule learning approach. *International Journal of Intelligent Systems. 14(11), 1123-1153.*

[CS87] Chameau, J. L. y Santamarina, J. C. Membership functions part I: Comparing method of measurement, *International Journal of Approximate Reasoning 1: 287–301.*

[CWC95] Ching, J. Y., Wong, A. K. C. y Chan, K. C. C. Class-Dependent Discretization for Inductive Learning from Continuous and Mixed-Mode Data. *IEEE Trans. Pattern Analysis and Machine Intelligence, vol. 17, no. 7, pp. 641-651, July.*

[DAPM00] Deb K., Agrawal S., Pratab A., Meyarivan T. A Fast Elitist Non-Dominated Sorting Genetic Algorithm for Multi-Objective Optimization: NSGA-II, *Proceedings of the Parallel Problem Solving from Nature VI Conference, pages 849-858. Springer.*

[Din67] Dinkelbach, W. On nonlinear fractional programming. *Managament Science, 13 pp. 492-498.*

[DKS95] Dougherty, J., Kohavi, R. y Sahami, M. Supervised and Unsupervised Discretization of Continuous Features. *Proc. 12th Int'l Conf. MachineLearning, pp. 194-202.*

[DP91] Dubois, D. y Prade, H. Fuzzy sets in approximate reasoning, part 1: inference with possibility distributions. *Fuzzy sets and Systems, vol. 40, pp. 143-202.*

[FF93] Fonseca, C.M. y Fleming, P.J. Genetic Algorithm for Multiobjective Optimization: Formulation, Discussion and Generalization. *In S. Forest (ed.), Proceedings of th Third International Conference on Genetic Algorithms. Morgan Kaufmann Publishers, pp. 416-423.*

[FHT98] Friedman, J., Hastie, T., Tibshirani, R. Additive Logistic Regression: A Statistical View of Boosting. *Machine Learning..*

[FI93] Fayyad, U. M. y Irani, K. B. Multi-Interval Discretization of Continuous-Valued Attributes for Classification Learning. *Proc. 13th Int'l Joint Conf Artificial Intelligence, pp. 1022-1029.*

[Fis36] Fisher, R. A. The use of multiple measurements in taxonomic problems. *Annual Eugenics 7(Part II): 179-188.*

[FJ00] Fajfer, M. y Janikow, C.Z. Bottom-Up Fuzzy Partitioning in Fuzzy Decision Trees. *Proc. 19th Int'l Conf. North Am. Fuzzy Information Processing Soc., pp. 326-330.*

[GH97] Gonzalez, A. y Herrera, F. Multi-stage genetic fuzzy systems based on the iterative rule learning approach. *Mathware and Soft Computing 4(3): 233 249.*

[Gil88] Giles, R. The concept of grade of membership. *Fuzzy Sets and Systems 25: 297–323.*

[GN96] Giordana, A y Neri, F. Search intensive concept induction. *Evolutionary Computation 3, 375-416.*

[Gol89] Goldberg, D. E. Genetic algorithms in search optimization ad machine learning. *Addison-Wesley Reading, MA.*

[GP98] Gonzalez, A. y Perez, R. Completeness and consistency conditions for learning fuzzy rules. *Fuzzy Sets and Systems 96: 37-51.*

[GS93] Giordana, A. y Saitta, L. REGAL: an integrated system for learning relations using genetic algorithm. *In Proc. Second International Workshop on Multistrategy Learning, Harpers Ferry, VA, USA, pp. 234-249.*

[GSm92] Greene, D. P. y Smith, S. F. COGIN: symbolic induction with genetic algorithms. *In Proc. Tenth National Conference on Artificial Intelligence, San Mateo, CA, USA, pp. 111-116. Morgan Kaufmann.*

[GSm93] Greene, D. P. y Smith, S. F. Competition-based induction of decision models from examples. *Machine Learning 13, 229-257.*

[GSm94] Greene, D. P. y Smith, S. F. Using coverage and a model building constraint in learning classifier systems. *Evolutionary Computation 2, 67-91.*

[Hay99] Haykin, S. Neural Networks. *Prentice Hall.*

[HC76] Hersh, H. y Carmazza, A. A fuzzy set approach to modifiers and vagueness in natural language, *Journal of Experimental Psychology: General 105(3): 254–276.*

[Hek96] Hekanaho, J. Background knowledge in GA-based concept learning. *In Proc. Thirteenth International Conference on Machine Learning, pp. 234-242.*

[Hek97] Hekanaho, J. GA-based rule enhancement concept learning. *In Proc. Third International Conference on Knowledge Discovery and Data Mining, Newport Beach, CA, USA, pp.183-186.*

[His85] Hisdal, E. Reconciliation of the Yes No versus grade of membership dualism in human thinking, *in M. M. Gupta, A. Kandel,W. Bandler & J. B. Kiszka (eds), Approximate Reasoning in Expert Systems, NorthHolland, Amsterdam, pp. 33–46. Also in Dubois, D., H. Prade, and R. Yager eds., Readings in Fuzzy Sets for Intelligent Systems, Morgan Kaufmann, pp. 854860.*

[His88] Hisdal, E. Are grades of membership probabilities?, *Fuzzy Sets and Systems 25: 325–348.*

[HKC01] Hong, T. P., Kuo, C. S. y Chi, S. C. Trade-off Between computation Time and Number for Fuzzy Mining from Quantitative Data. *International Journal of Uncertainty Fuzziness and Knowledge- Base Systems, 9(5), 586-604.*

[HLPR89] Hilliard, M.R., Liepins, M. Palmer, M., Rangarajen, G. The computer as a partner in algorithmic design: Automated discovery of parameters for a multiobjective scheduling heuristic. *In R. Sharda, B.L. Golden, E. Wasil, O. Balci and W. Stewart (eds.), Impact of Recent Computer Advances on Operations Research. North-Holland Publishing Company.*

[Hol75] Holland, J. R. Adaptation en natural and artificial systems. *University of Michigan Press, Ann Arbor.*

[HLV95] Herrera, F., Lozano, M. y Verdegay, J. L. Tuning fuzzy controllers by genetic algorithms. *International Journal of Approximate Reasoning 12: 299-315.*

[HLV98] Herrera, F., Lozano, M. y Verdegay, J. L. Tackling real-coded genetic algorithms: operators and tools for the behaviour analysis. *Artificial Intelligence Review 12: 265-319.*

[HR78] Holland, J. H. y Reitman, J. S. Cognitive systems based on adaptative algorithms. *In D. A. Waterman and F. Hayes-Toth (Eds.), Pattern-Directed Inference Systems. Academic Press*

[INM95] Ishibuchi, H., Nakashima, T y Murata, T. A fuzzy classifier system that generated fuzzy if-then rule for pattern classification problems. *In Proc. of 2th IEEE International conference of evolutionary computation, 759-764.*

[INM99] Ishibuchi, H., Nakashima, T y Murata, T. Voting in fuzzy rule-based systems for pattern classification problems. *Fuzzy Set and Systems 103(2) 223-238.*

[INN04] Ishibuchi, H., Nakashima, T. y Nii, M. Classification and Modeling with Linguistic Information Granules: Advanced Approaches to Linguistic Data Mining, *pp.1-308, Springer, Berlin, November.*

[INT92] Ishibuchi, H., Nozaki, K., y Tanaka, H. Distributed representation of fuzzy rules and its application to pattern classification. *Fuzzy Sets and Systems 52: 21-32.*

[INYT94] Ishibuchi, H., Nozaki, K., Yamamoto, N. y Tanaka, H. Construction a fuzzy classification system with rectangular fuzzy rules using genetic algorithms. *Fuzzy set and systems,* 65(2/3), *237-253.*

[INYT95] Ishibuchi, H., Nozaki, K., Yamamoto, N. y Tanaka, H. Selecting fuzzy If-then rules for classification problems using genetic algorithms. *IEEE Trans. on fuzzy systems*, *3(3), 260-270.*

[IYN01] Ishibuchi, H., Yamamoto, T., Nakashima, T. Fuzzy Data Mining: Effect of Fuzzy Discretization. *In Proc. Of 1st IEEE Internacional Conference on Data Mining*, *242-248.*

[Jan93] Jang, J.-S. R. ANFIS: Adaptive-Network-Based Fuzzy Inference System. *IEEE Trans. Systems, Man, and Cybernetics, vol. 23, no. 3, pp. 665-685.*

[Jes99] Jesús, M.J. del, Aprendizaje evolutivo de sistemas de clasificación basados en reglas difusas. *Tesis Doctoral.*

[JF99] Janikow, C. Z. y Fajfer M. Fuzzy Partitioning with FID3.1. *Proc. 18thInt'l Conf. North Am. Fuzzy Information Processing Soc., pp. 467-471.*

[JHJS04] Jesus, M. J. del, Hoffmann F., Junco L., Sánchez L. Induction of Fuzzy Rule Based Classifiers with Evolutionary Boosting Algorithms. *IEEE Transactions on Fuzzy Sets and Systems 12(3): 296-308.*

[JS00] Junco L., Sanchez L. Using the Adaboost algorithm to induce fuzzy rules in classification problems, *Proc. ESTYLF 2000, Sevilla, pp 297-301.*

[Kam75] Kamp, J. A. W. Two theories about adjectives, *in E. L. Keenan (ed.), Formal Semantics of Natural Language, Cambridge University Press, London, pp. 123-155.*

[Kar91] Karr, C. L. Design of an Adaptive Fuzzy Logic Controller Using a Genetic Algorithm. *Proc. Fourth Int'l Conf. Genetic Algorithms, pp. 450-457.*

[KB74a] Kochen, M. y Brade, A. N. On the precision of the adjectives which denote fuzzy sets, *Journal of Cybernetics 4: 49-59.*

[KB74b] Kochen, M. y Badre, A. N. On the precision of adjectives which denote fuzzy sets, *Journal of Cybernetics 4: 49–59.*

[Kos91] Kosko, B. Neural Networks and Fuzzy Systems, *PrenticeHall, Englewood Cliffs, NJ.*

[KM87] Kruse, R. y Meyer, K. D. Statistics with vague data. *D. Reidel Publishing Company.*

[Kun00] Kuncheva, L. I. Fuzzy Classifier Design. *Springer-Verlag, NY.*

[Kun96] Kuncheva, L.I. On the equivalence between fuzzy and statistical classifiers. *International Journal of Uncertainty, Fuzzyiness and Knowledge-Based Systems 4(3): 245-253.*

[KY95] Klir, G. y Yuan B. Fuzzy Sets and Fuzzy Logic. *New Jersey: Prentice-Hall International .*

[Lak87] Lakoff, G. Women, Fire, and Dangerous Things: What categories reveal about the mind, *The University of Chicago Press, Chicago*

.

[Lim05] Limbourg, P. Multi-objective optimization of problems with epistemic uncertainty, *in EMO, 2005, pp. 413-427.*

[LWW04] Liu, L., Wong, A.K. C. y Wang, Y. A Global Optimal Algorithm for Class-Dependent Discretization of Continuous Data. *Intelligent Data Analysis, vol. 8, no. 2, pp. 151-170.*

[Men95] Mendel, J. M. Fuzzy logic systems for engineering: A tutorial. *Proceedings of the IEEE, vol. 83 no.3, pp. 345-377.*

[Mic96] Michalewicz Z. Genetic algorithms + data structures = evolution programs. *Springer-Verlag.*

[MMP92] Mandal, D.P., Murthy, C. A., y Pal, S. K. Formulation of a multivaluated recognition system. *IEEE Transactions on Systems, Man and Cybernetics 22(4): 607-620.*

[NG95] Neri, F. y Giordana, A. A parallel genetic algorithm for concept learning. In *L. Eshelman (Ed.), Proc. Sixth international Conference on Genetic Algorithms (ICGA'95), pp. 436-443. Morgan Kaufmann.*

[NIT96] Nozaki, K., Ishibuchi, H, y Tanaka, H. Adaptive fuzzy rule based classification systems. *IEEE transactions on Fuzzy Systems 4: 238-250.*

[NK97] Nauck, D. y Kruse, R. A neuro-fuzzy method to learn fuzzy classification rules from data. *Fuzzy Sets and Systems 89: 277-288.*

[NT82a] Norwich, A. M. & Türksen, I. B. The construction of membership functions, *in R. R. Yager (ed.), Fuzzy Sets and Possibility Theory: Recent Developments, Pergamon Press, New York, pp. 61–67.*

[NT82b] Norwich, A. M. & Türksen, I. B. The fundamental measurement of fuzziness, *in R. R. Yager (ed.), Fuzzy Sets and Possibility Theory: Recent Developments, Pergamon Press, New York, pp. 49–60.*

[NT84] Norwich, A. M. y Türksen, I. B. A model for the measurement of membership and the consequences of its empirical implementation, *Fuzzy Sets and Systems 12: 1–25.*

[NTS93] Nakanishi, H., Turksen, I. B. y Ugeno, M. A review and comparison of six reasoning methods, *Fuzzy Sets and Systems 57: 257–294.*

[OS06] Otero, J., Sánchez, L. Induction of descriptive fuzzy classifiers with the Logitboost algorithm. *Publication in Soft Computing.*

[PM79] Procyk, T. J. y Mamdani, E. H. A linguistic self-organizing process controller. *Automática 15(1), 15-30.*

[PM92] Pal, S. K. y Mandal, D. P. Linguistic recognition system base on approximate reasoning. *Information Sciences 61: 135-161.*

[RM75] Rosch, E. y Mervis, C. B. Family resemblances: Studies in the internal structure of categories, *Cognitive Pschology 7: 573–605.*

[RN95] Reyero, R. y Nicolás, C. Sistemas de control basados en Lógica borrosa: fuzzy control. *Madrid: Omron Electronic S.A. Centro de Investigaciones Tecnológicas IKERLAN.*

[Roc82] Rocha, A. F.Basic properties of neural circuits, *Fuzzy Sets and Systems 7: 109–121.*

[Rus69] Ruspini, E.H. A new approach to clustering. *Information and Control 15:22-32.*

[Saa86] Saaty, T. L. Scaling themembership function, *European Journal of Operacional Research 25: 320–329.*

[SD94] Srinivas, N., Deb, K. Multiobjective Optimization using Nondominated Sorting in Genetic Algorithm. *Evolutionary Computation, 2, pp.221-248.*

[SO03] Sánchez L., Otero J. Tuning fuzzy partitions or assigning weights to fuzzy rules: which is better? Accuracy Improvements in Linguistic Fuzzy Modeling. *J. Casillas, O. Cordón, F. Herrera, L. Magdalena (Eds.). Physica-Verlag. pp 266-386.*

[SO07] Sánchez, L., Otero, J. Boosting fuzzy rules in classification problems under single-winner inference. *International Journal of Intelligent Systems In press.*

[SOV06] Sánchez, L., Otero, J., Villar, J.R. Boosting of fuzzy models for high-dimensional imprecise datasets. *In 11th International Conference on Information Processing and Management (IPMU2006). Paris (France).*

[Smi80] Smith, S. F. A learning system based on genetic algorithms. *PhD dissertation, University of Pittsburg*, Pittssburg.

[Sni92] Sniedovich, M. Dynamic Programming. *New York: Marcel Dekker, Inc.*

[SSC05] Sanchez, L., Suárez, M. R., Couso, I. A fuzzy definition of Mutual Information with application to the Genetic Fuzzy Classifiers. *International Conference on Machine Intelligence, Tozeur – Tunisia, November 5-7.*

[Str00] Straszecka, E. Cap. Defining Membership functions, Fuzzy systems in medicine. *P. Szczepaniak, G. L. P.J y J. Kacprzyk, Ed. New York: Physica-Verlag, 2000, pp. 32-47.*

[SW49] Shannon, C. E. y Weaver, W. The Mathematical Theory of Comunication. *Urbana, IL: University of Illinois Press.*

[Tak90] Takagi, H. Fusion technology of fuzzy theory and neural network: Survey and future directions, *Proc. of International Conference on Fuzzy Logic and Neural Networks, Iizuka, Japan, pp. 13–26.*

[TH91] Takagi, H. & Hayashi, I. Non-driven fuzzy reasoning, *International Journal of Approximate Reasoning 5(3): 191–213. (Special Issue of IIZUKA'88).*

[Tür88] Türksen, I. B. Stochasic fuzzy sets: a survey, Combining Fuzzy Imprecision with Probabilistic Uncertainty in Decision Making, *Springer-Verlag, New York, pp. 168–183.*

[Tür91] Turksen, I. B. Measurement of membership functions and their acquisition. *Fuzzy Sets and Systems, 40:5–38.*

[Ven93] Venturini, G. SIA: A supervised inductive algorithm with genetic search for learning attributes based concepts. *Lecture Notes in Artificial Intelligence: Machine Learning (ECML'93), páginas 280-296.*

[Wal91] Walley, P., Statistical Reasoning with Imprecise Probabilities. *Chap-man and Hall, London.*

[WC87] Wong, A.K.C. y Chiu, D.K.Y. Synthesizing Statistical Knowledge from Incomplete Mixed-Mode Data. *IEEE Trans. Pattern Analysis and Machine Intelligence, vol. 9, no. 6, pp. 796-805.*

[WM99] Walley, P. y Moral, S., Upper probabilities based only on the likelihood function, *J. Roy. Statist. Soc. Ser. B, vol. 61, pp. 831-847.*

[Yag79] Yager, R. R. A measurement–informational discussion of fuzzy union and intersection, *International Journal of Man–Machine Studies 11: 189–200.*

[Zad75] Zadeh, L. A. Fuzzy logic and approximate reasoning. *Synthese 30: 407–428.*

[Zys81] Zysno, P. Modeling membership functions, *in B. B. Rieger (ed.), Empirical Semantics I, Vol. 1 o Quantitative Semantics, Vol. 12, Studienverlag Brockmeyer, Bochum, pp. 350-375.*

[ZZ94] Zeng, X. y Zingh, M. Approximation theory of fuzzy systems-SISO case. *IEEE Trans. Fuzzy Systems, vol. 2, no.2, pp. 162-176.*