UNIVERSIDAD AUTÓNOMA DE MADRID
ESCUELA POLITÉCNICA SUPERIOR

Doble grado en Ingeniería Informática y Matemáticas

TRABAJO FIN DE GRADO

Aprendizaje automático basado en las extensiones de la integral de Sugeno.

Autor: María Sarnago Laplaza
Tutores: Humberto Bustince, José Antonio Sanz
Ponente: Jose Dorronsoro

Junio 2021

María Sarnago Laplaza
Aprendizaje automático basado en las extensiones de la integral de Sugeno.

María Sarnago Laplaza
C\ Francisco Tomás y Valiente Nº 11

IMPRESO EN ESPAÑA – PRINTED IN SPAIN

Resumen

Uno de los problemas más importantes en el ámbito del aprendizaje automático es el *problema de la clasificación*. Entre los métodos más utilizados para resolverlos se encuentras las redes neuronales, redes bayesianas o árboles de decisión. Sin embargo, en este trabajo nos centramos en un tipo de clasificadores que todavía no han recibido mucha atención en esta rama de la Inteligencia Artificial, los clasificadores de Sugeno.

El *clasificador de Sugeno* fue propuesto por el científico alemán Eyke Hülermeier en [1] como método para la clasificación binaria, en el que la integral de Sugeno se utiliza como función de agregación que combina varias evaluaciones locales de una instancia en una única evaluación global. Gracias a los resultados del estudio experimental que realizó, comprobamos la gran capacidad de predicción que posee el clasificador de Sugeno. Posteriormente se han publicado varias extensiones de esta integral y, por este motivo, nos preguntamos cómo de precisos serán los clasificadores basados en dichas extensiones. Así, nuestro objetivo es proponer cinco posibles nuevos métodos de clasificación binarios basados en extensiones de la integral de Sugeno.

En este trabajo estudiamos tanto la integral de Sugeno como el algoritmo de clasificación del clasificador de Sugeno junto con los procesos de aprendizaje correspondientes, y proponemos nuevos métodos de clasificación binaria. Para estudiar el comportamiento de estas propuestas, utilizamos ocho conjuntos de datos reales y realizamos un análisis estadístico para apoyar las conclusiones obtenidas. Con el objetivo de comprobar la calidad de los métodos de clasificación propuestos, llevamos a cabo una comparativa del clasificador de Sugeno con cada una de nuestras propuestas. Los resultados experimentales muestran la competitividad de los métodos de clasificación desarrollados y la utilidad de las extensiones de la integral de Sugeno como funciones de agregación en este contexto.

Palabras clave

Aprendizaje automático, clasificación binaria, medida difusa, funciones de agregación, integral de Sugeno

Abstract

One of the most important problems in the context of machine learning is the *classification problem*. Among the most commonly used methods to solve them are neural networks, Bayesian networks or decision trees. However, in this work we focus on a type of classifiers that have not yet received much attention in this branch of Artificial Intelligence, the Sugeno classifiers.

The *Sugeno classifier* was proposed by the German scientist Eyke Hülermeier in [1] as a method for binary classification, in which the Sugeno integral is used as an aggregation function that combines multiple local evaluations of an instance into a single global evaluation. Due to the results of the experimental study he carried out, we realised that the Sugeno classifier is competitive in terms of predictive accuracy. Several extensions of this integral have subsequently been published and, for this reason, we wonder how accurate the classifiers based on those extensions will be. Therefore, our objective is to propose five new binary classification methods based on extensions of the Sugeno integral.

In this paper we study the Sugeno integral and the classification algorithm of the Sugeno classifier along with the corresponding learning processes. Furthermore, we propose new methods of binary classification. To study the behavior of these proposals, we make use of eight real datasets and performed a statistical analysis to support the conclusions. In order to check the precision of the proposed classification methods, we compare each of our methods with the Sugeno classifier. The experimental results show the competitiveness of the new classification methods and the usefulness of the extensions of the Sugeno integral as aggregation functions in this context.

Keywords

Machine learning, Binary classification, fuzzy measure, aggregation function, Sugeno integral

ÍNDICE

Listas

Lista de algoritmos

Lista de ecuaciones

Lista de figuras

Lista de tablas

1

INTRODUCCIÓN

Actualmente, el *Aprendizaje Automático* o *Machine Learning* tiene una gran importancia en el mundo debido a la enorme cantidad de datos que generamos. Estos datos son una extensa fuente de información muy valiosa que, sin ayuda de las máquinas, sería imposible de analizar. El Aprendizaje Automático es una rama de la *Inteligencia Artificial* que proporciona a las máquinas la capacidad de aprender analizando conjuntos de datos. Uno de los problemas más importantes de este ámbito es el *problema de clasificación*, problema que consiste en aprender a asignar etiquetas o clases a datos a partir de un conjunto de datos inicial conocido como *conjunto de entrenamiento*. En función del número de clases que se consideren en el problema, distinguimos dos tipos de problemas de clasificación: problemas de clasificación binarios (dos clases) y problemas de clasificación multi-clase (más de dos clases).

La importancia de combinar modelos matemáticos de decisión y funciones de agregación con métodos de Aprendizaje Automático ha ido creciendo a lo largo de los años proporcionando estudios realmente interesantes. Hasta ahora, en la literatura se han desarrollado varios métodos de clasificación basados en la *integral de Choquet* debido a sus numerosas propiedades como [2] y [3]. Sin embargo, por su parte, la integral de Sugeno no ha recibido tanta atención en el Aprendizaje Automático. Esto es debido a que los métodos basados en la integral de Choquet se pueden interpretar como generalizaciones del aprendizaje de modelos lineales, mientras que los basados en la integral de Sugeno están más relacionados con los árboles de decisión y los modelos basados en reglas. Así, la predicción de un método basado en la integral de Sugeno es algo más complejo.

La integral de Sugeno ha sido utilizada en la literatura para problemas de Aprendizaje Automático. Más concretamente, el científico alemán *Eyke Hülermeier* propuso un método para la clasificación binaria, en el que la integral de Sugeno se utiliza como función de agregación que combina varias evaluaciones locales de una instancia, pertenecientes a diferentes atributos, en una única evaluación global. Los resultados del estudio experimental que llevó a cabo fueron realmente satisfactorios y prometedores, mostrando así la elevada capacidad de predicción del clasificador de Sugeno.

Las extensiones de la integral de Sugeno han sido propuestas en [4] recientemente con el fin de utilizarlas en *algoritmos de binarización de umbral adaptativo*. Los resultados de esta propuesta han sido bastante prometedores en este ámbito, por lo que es interesenta estudiar estas extensiones en otro tipo de algoritmos.

El propósito principal del TFG es estudiar la integral de Sugeno y el clasificador de Sugeno, así como estudiar las extensiones de dicha integral para desarrollar métodos de clasificación binaria basados en las extensiones como medio para agregar información. Además, comparamos experimentalmente los nuevos métodos de clasificación binaria con el basado en la integral de Sugeno y analizamos los resultados obtenidos. Así, los objetivos perseguidos en esta memoria son:

1. Estudiar tanto la integral de Sugeno como el clasificador de Sugeno, y analizar los procesos de aprendizaje del mismo.
2. Estudiar las posibles extensiones de la integral de Sugeno y proponer métodos de clasificación binaria basados en ellas.
3. Implementar los métodos de clasificación binaria propuestos en el lenguaje de programación Python.
4. Realizar un estudio experimental para evaluar el rendimiento de los métodos de clasificación, y comparar el clasificador de Sugeno con las nuevas propuestas.

Aparte de esta introducción, el trabajo se estructura en siete capítulos. En el Capítulo 2 nos situamos en el entorno en el que se desarrolla este proyecto introduciendo brevemente el aprendizaje automático y definiendo los problemas de clasificación y las máquinas de vector soporte. Además, presentamos ciertas definiciones y resultados matemáticos necesarios para comprender la integral de Sugeno. En el Capítulo 3 definimos la integral de Sugeno y desarrollamos tanto el algoritmo de clasificación de Sugeno como su proceso de aprendizaje. En el Capítulo 4 presentamos las extensiones de la integral de Sugeno más interesantes, y proponemos nuevos métodos de clasificación binaria basados en ellas. En el Capítulo 5 describimos los experimentos que vamos a realizar con el objetivo de comparar los clasificadores y llevamos a cabo las distintas pruebas. Por último, en el Capítulo 6 concluimos el trabajo y presentamos posibles trabajos futuros sobre la investigación realizada en este proyecto.

2

Marco Teórico

En este capítulo presentamos el marco teórico en el que vamos a desarrollar el trabajo, el método de clasificación binaria, así como un método de aprendizaje necesario para comprender el diseño del clasificador de Sugeno como son las máquinas de vector soporte. Además, definimos ciertos conceptos y resultados matemáticos necesarios para poder comprender la integral de Sugeno.

2.1. Clasificación binaria

El Aprendizaje Automático es la rama de la Inteligencia Artificial que proporciona a las máquinas la capacidad de aprender identificando patrones en los datos o instancias. Según el proceso de aprendizaje llevado a cabo, existen diversos tipos como el aprendizaje supervisado, el aprendizaje no supervisado o el aprendizaje por refuerzo.

1. El *aprendizaje supervisado* es un método de aprendizaje basado en el conocimiento a priori. Consiste en, a partir de un conjunto de datos y los valores a predecir, establecer la mejor relación entre ambos. De esta forma, el algoritmo busca tendencias entre los datos y los valores a predecir con el fin de ser capaz de asignar un valor conocido a un nuevo dato. Este tipo de aprendizaje se utiliza para realizar un diagnostico médico basado en los síntomas de un paciente y para detectar del rostro en los móviles, entre otras muchas aplicaciones.

2. El aprendizaje no supervisado, a diferencia del anterior, no está basado en un conocimiento previo ya que dispone exclusivamente del conjunto de datos. El objetivo de este proceso de aprendizaje es analizar los datos y encontrar patrones en estos para poder organizarlos por clases. Así, resulta útil para detectar fraudes financieros o segmentar conjuntos de datos por características comunes.

3. El aprendizaje por refuerzo es un método de aprendizaje que aprende de la propia experiencia a través de un proceso de prueba-error en el que las acciones correctas son recompensadas. El objetivo de este algoritmo es maximizar las

recompensas obtenidas repitiendo las decisiones correctas y evitando el resto. Algunas aplicaciones reales de este tipo de aprendizaje las encontramos en sistemas de navegación o en regímenes de tratamiento dinámico en enfermedades crónicas.

Centrándonos en el aprendizaje supervisado, distinguimos dos tipos de problemas: los *problemas de regresión* y *los problemas de clasificación*. En los problemas de regresión, la salida es cuantitativa, es decir, un valor numérico; mientras que en los problemas de clasificación, la salida es cualitativa, es decir, una clase o categoría. Resolver un problema de clasificación consiste en aprender una regla de decisión capaz de determinar la clase de un nuevo dato de entre las clases existentes. A la regla de decisión le denominamos *clasificador*. Por tanto, un clasificador es un sistema capaz de asignar clases a los datos que le proporcionan. No obstante, clasificar datos no es tan sencillo como parece, ya que es necesario que el clasificador adquiera un conocimiento previo. Esto se denomina *proceso de aprendizaje*, y permite al clasificador determinar las características propias de cada clase. Además, dependiendo del número de clases que componen la salida del problema, estos problemas están divididos en dos tipos: binarios y multi-clase. Como ya introdujimos, en el presente trabajo nos centramos en el problema de clasificación binario, uno de los problema más estudiados en el aprendizaje automático.

Un *clasificador binario* es una regla de decisión o función de mapeo que asigna a cada instancia la clase 0 o 1 según los criterios establecidos en el algoritmo y, cuyo proceso de aprendizaje consiste en analizar todas las hipótesis de la clase del clasificador con el objetivo de determinar la que mejor se adapta al conjunto de entrenamiento. Denotamos $\mathcal{H}$ al espacio que contiene todas las posibles funciones $h : \mathcal{X} \to \{0, 1\}$, conocido por *espacio de hipótesis* o *clase del clasificador*, y $\mathcal{X}$ al *espacio de instancias*, las cuales están descritas en términos de atributos, es decir, $\mathcal{X} = \mathcal{X}_1 \times \ldots \times \mathcal{X}_n$, donde $\mathcal{X}_i$ es el i-ésimo atributo de la instancia $\mathcal{X}$. Al conjunto de entrenamiento le denotamos $\mathcal{D}$ y está compuesto por el conjunto de datos ya clasificados que se le proporciona al algoritmo de clasificación. Para determinar qué regla de decisión se adapta mejor a estos datos, definimos el *riesgo del clasificador* como la función $R(\cdot)$ definida por:

$$\mathcal{R}(h) = \iint_{\mathcal{X}\times\{0,1\}} \mathcal{L}(y, h(x))dP(x, y), \tag{2.1}$$

donde $\mathcal{L}$ es una función de pérdida, función que determina el error entre el valor estimado y el valor real. Analizando la función del riesgo del clasificador observamos que el riesgo del clasificador se obtiene calculando la esperanza matemática de la función de pérdida, observación que tendremos en cuenta en el siguiente capítulo. Así, el proceso de aprendizaje

de un clasificador consiste esencialmente en elegir la función h del espacio $\mathcal{H}$ que minimice el riesgo sobre el conjunto de entrenamiento $\mathcal{D}$.

Por otro lado, teniendo en cuenta que cada instancia x está descrita por n valores, uno por atributo, y que generalmente dichos valores no están expresados en el mismo universo de discurso, es necesario unificarlos para poder asignar una clase a la instancia. Para ello, definimos la *función de evaluación local* f_i que se encarga de transformar el atributo i-ésimo de la instancia x en su criterio local u_i. De esta forma, obtenemos los *criterios locales* $u_1, \ldots, u_n$ de $x_1, \ldots, x_n$, respectivamente, como sigue:

$$(u_1, \ldots, u_n) = (f_1(x_1), \ldots, f_n(x_n)) \,. \tag{2.2}$$

Una vez obtenidos los criterios locales, procedemos a realizar el proceso de clasificación según el método de clasificación utilizado.

2.1.1. Máquinas de vector soporte

Las *máquinas de vector soporte* (SVM) son modelos de aprendizaje supervisado utilizados para resolver problemas de clasificación y regresión. Este conjunto de algoritmos de aprendizaje fue desarrollado inicialmente por Vladimir Vapnik y su equipo de trabajo en los laboratorios de AT&T en [5] para resolver problemas de clasificación y, posteriormente, se desarrolló una nueva versión de la SVM para problemas de regresión en [6]. En un problema de clasificación, la SVM interpreta los datos como vectores de dimensión n e intenta buscar un hiperplano de dimensión $n-1$ que separe linealmente las clases del problema. Si encuentra diversos hiperplanos que las separen, escoge aquel que maximiza la distancia de este a los puntos más cercanos de cada clase. A dicho hiperplano le denominamos *hiperplano óptimo*. Además, denominamos *vector soporte asociado a una clase* al vector formado por los puntos de dicha clase más cercanos al hiperplano óptimo.

En el caso de problemas de clasificación binaria, el modelo busca el hiperplano óptimo de dimensión 1 que maximiza la distancia a los vectores soporte de las clases positiva y negativa y separa linealmente ambas clases. Más formalmente, la SVM busca el hiperplano $\mathcal{H}(w, b)$, donde $w \in \mathbb{R}^m$ y $b \in \mathbb{R}$ tal que cumple que, para $i = 1, \ldots, m$ y $\left(x^{(i)}, y^{(i)}\right) \in \mathcal{D}$:

$$\begin{cases} w \cdot x^{(i)} \geqslant b, & \text{si } \; y^{(i)} = 1, \\ w \cdot x^{(i)} \leqslant b, & \text{si } \; y^{(i)} = 0, \end{cases} \tag{2.3}$$

y que, además, maximiza la distancia entre las dos clases.

Sin embargo, puede ocurrir que el conjunto de entrenamiento $\mathcal{D}$ no sea linealmente separable por ningún hiperplano $\mathcal{H}(w,b)$. En este caso, el hiperplano óptimo es aquel que minimice el número de errores. Para determinar los errores, definimos el *margen de error* de una instancia concreta como $\rho_i \geqslant 0 \in \mathbb{R}$, donde $i = 1,\dots,m$, y, así, el objetivo de la SVM es encontrar el hiperplano que minimice $\sum_{i=1}^{m} \epsilon_i$ sujeto a:

$$\begin{cases} wx^{(i)} + \epsilon_i \geqslant b, & \text{si } y^{(i)} = 1, \\ wx^{(i)} + \epsilon_i \leqslant b, & \text{si } y^{(i)} = 0. \end{cases} \tag{2.4}$$

Destacamos que este concepto es necesario para desarrollar el proceso de aprendizaje tanto del método de clasificación de Sugeno como de los métodos de clasificación que se proponen.

2.2. Nociones Matemáticas

En esta segunda sección, introducimos los términos y resultados matemáticos imprescindibles para comprender la integral de Sugeno. En primer lugar, denotamos el conjunto vacío por $\emptyset$ y el conjunto de partes de $[n] = \{1,\dots,n\}$ como $2^{[n]}$ compuesto por todos los subconjuntos de $[n]$.

Definición 2.2.1 (Medida difusa)**.** Sea $n \in \mathbb{N},\ [n] = \{1,\dots,n\}$. Una *medida difusa* es una función $\mu : 2^{[n]} \to [0,1]$ que satisface:

$$\begin{cases} \mu(\emptyset) = 0, \\ \mu([n]) = 1, \\ \mu(A) \leqslant \mu(B), \text{ si } A \subseteq B. \end{cases} \tag{2.5}$$

(Véase el Artículo [7]).

Además, una medida difusa μ es *simétrica* si para cualquier par de conjuntos $A, B \subseteq [n]$ tales que $|A| = |B|$, se cumple que $\mu(A) = \mu(B)$. Una propiedad de las medidas que también nos va a ser útil de cara a optimizar el clasificador es la k-maxitividad.

Definición 2.2.2 (Medida difusa k-maxitiva)**.** Una medida difusa μ es *k-maxitiva* si para cualquier $A \subseteq [n],$ con $|A| > k$, existe un subconjunto propio $B \subset A$ tal que $\mu(A) = \mu(B)$ o, equivalentemente,

$$\mu(A) = \bigvee_{B \subset A} \begin{cases} \mu(B), & \text{si } |A| > k. \end{cases} \tag{2.6}$$

Por simplicidad, dado que todas las medidas que se van a usar son medidas difusas

simétricas, se les denotará como medidas difusas o simplemente como medidas a lo largo del trabajo basado en dicha integral.

Dada la relación entre la integral de Sugeno y el estadístico de la mediana que veremos, definimos el estadístico.

Definición 2.2.3 (Mediana)**.** La función *mediana* está definida para cualquier vector ordenado $x \in \mathbb{R}^n$, donde $x = (x_1, \ldots, x_n)$ y $x_1 \leqslant \ldots \leqslant x_n$, de la siguiente manera:

$$\begin{cases} mediana(x) = x_{\frac{n+1}{2}}, & \text{si } n \text{ es impar,} \\ mediana(x) = \dfrac{x_{\frac{n}{2}} + x_{\frac{n}{2}+1}}{2}, & \text{si } n \text{ es par.} \end{cases} \tag{2.7}$$

La mediana también se puede expresar en términos de la intersección y conjunción como:

$$mediana(x) = \bigvee_{I \subseteq [2n+1],\ |I|=n+1} \bigwedge_{i \in I} x_i, \quad \forall n \geqslant 1. \tag{2.8}$$

(Véase el Artículo [8]).

Otra definición imprescindible es la de función de agregación, puesto que la integral de Sugeno es una función de este tipo.

Definición 2.2.4 (Función de agregación)**.** Una función $\mathcal{A} : [0, \infty)^n \to [0, \infty)$ es una *función de agregación* si es no decreciente y se cumple que

$$\inf_{x \in [0,\infty)^n} \mathcal{A}(x) = 0 \quad \text{y} \quad \sup_{x \in [0,\infty)^n} \mathcal{A}(x) = \infty, \tag{2.9}$$

donde inf representa el ínfimo y sup el supremo de la función $\mathcal{A}$ sobre $[0, \infty)^n$.

En el ámbito de las funciones de agregación, las medidas difusas se utilizan para determinar la relación existente entre los elementos que se van agregar.

A continuación, dado que la integral de Sugeno se puede ver como una función del retículo, definimos este tipo de función y establecemos un resultado que nos será útil para definir la integral como una función con estas características.

Definición 2.2.5 (Retículo)**.** Un *retículo* es una estructura algebraica $(\mathcal{L}, \wedge, \vee)$, donde $\mathcal{L}$ es un conjunto no vacío y $\wedge$ y $\vee$ son dos operaciones binarias tales que satisfacen la propiedad conmutativa, asociativa y de absorción para cualquier par de elementos en $\mathcal{L}$.

Definición 2.2.6 (Función del retículo)**.** Una *función del retículo*, $(\mathcal{L}, \wedge, \vee)$ es una función $f : \mathcal{L}^n \to \mathcal{L}$ tal que preserva las operaciones del mismo es decir, para todo $x, y \in \mathcal{L}^n$ cumple que:

$$\begin{cases} f(x \wedge y) &= f(x) \wedge f(y), \\ f(x \vee y) &= f(x) \vee f(y). \end{cases} \tag{2.10}$$

Proposición 2.2.1. *Sea $f : \mathcal{L}^n \to \mathcal{L}$ una función, entonces las siguientes afirmaciones son equivalentes:*

(i) f es una función del retículo $\mathcal{L}$.

(ii) Existe una función $\alpha : 2^{[n]} \to \mathcal{L}$ tal que $f(x) = \bigvee_{I \subseteq [n]} \left(\alpha(I) \wedge \bigwedge_{i \in I} x_i \right)$.

(iii) Existe una función $\beta : 2^{[n]} \to \mathcal{L}$ tal que $f(x) = \bigwedge_{I \subseteq [n]} \left(\beta(I) \vee \bigvee_{i \in I} x_i \right)$.

Se dice que la expresión *(ii)* de la función f está escrita en su *forma normal disyuntiva* (FND), y la expresión *(iii)*, en su forma normal conjuntiva (CND). (Véase en el Artículo [8]).

Finalmente, introducimos la función cuantil ya que el proceso de aprendizaje de la función de evaluación del método de clasificación está basado en una aproximación por cuantiles.

Definición 2.2.7 (Función cuantil)**.** La *función cuantil* $Q(\cdot)$ es la inversa de la función de distribución de una variable aleatoria.

$$Q(p) = F^{-1}(p) = \text{ínf}\{x : F(x) \geqslant p\}, \tag{2.11}$$

donde $F(\cdot)$ es la función de distribución de la variable aleatoria y p una probabilidad. En otras palabras, la función cuantil indica el valor mínimo que tiene que tomar la variable aleatoria para el cual se cumpla que:

$$F(x) = P(X \leqslant x) = p. \tag{2.12}$$

(Véase su uso en el Artículo [9]).

3

CLASIFICADOR DE SUGENO

En este capítulo estudiamos la integral de Sugeno y diseñamos el clasificador de Sugeno, así como los procesos de aprendizaje del método de clasificación. Destacamos que para comprender la integral de Sugeno, necesitamos tener en cuenta las nociones matemáticas definidas en el capítulo anterior.

3.1. Integral de Sugeno

La *integral de Sugeno* o integral de Sugeno clásica es una función de agregación basada en operaciones disyuntivas y conjuntivas, y definida respecto a una medida difusa μ sobre $[n]$ de la siguiente manera:

$$Sg_\mu(u) = S_\mu(u_1, \ldots, u_n) = \bigvee_{i=1}^{n} \left(u_{\sigma(i)} \wedge \mu\left(A_{\sigma(i)}\right)\right), \quad u \in [0,1]^n, \tag{3.1}$$

donde $\sigma(\cdot)$ es la permutación sobre $[n]$ tal que $u_{\sigma(1)} \leqslant u_{\sigma(2)} \leqslant \ldots \leqslant u_{\sigma(n)}$ y $A_{\sigma(i)} = \{\sigma(i), \sigma(i+1), \ldots, \sigma(n)\}$, con $i = 1, \ldots, n$.

Fácilmente comprobamos que, además de ser una función de agregación, la integral de Sugeno es una función del retículo $([0,1], \wedge, \vee)$. Así, por la Proposición 2.2.1, podemos escribir la expresión (3.1) en su forma normal disyuntiva (FND), donde la función α coincide con la medida μ de la integral:

$$Sg_\mu(u) = S_\mu(u_1, \ldots, u_n) = \bigvee_{A \subseteq [n]} \mu(A) \wedge \bigwedge_{i \in A} u_i, \tag{3.2}$$

Una característica destacable de la integral de Sugeno clásica es su relación con la mediana (véase Definición 2.2.3), ya que la integral se puede expresar en términos de la mediana como:

$$Sg_\mu(u) = mediana\left(u_1, \ldots, u_n, \mu\left(A_{\sigma(2)}\right), \ldots, \mu\left(A_{\sigma(n)}\right)\right), \quad u \in [0,1]^n. \tag{3.3}$$

De esta forma, definimos la integral de Sugeno como una mediana ponderada, ordenando los $2n-1$ valores y tomando el valor que ocupa la posición n. Esta relación será útil para reducir la complejidad del algoritmo de clasificación y sus procesos de aprendizaje.

3.2. Algoritmo de Clasificación de Sugeno

El *clasificador de Sugeno* es un algoritmo de clasificación binario que utiliza la integral de Sugeno como función de agregación que combina los criterios locales de los atributos de cada instancia en una única evaluación global representativa. Este método de clasificación asigna la *clase positiva* o clase 1 a aquellas instancias cuyo *criterio global* U supera el valor del umbral, y la *clase negativa* o clase 0, al resto de instancias. Para ello, es necesario que el espacio de hipótesis $\mathcal{H}$ esté compuesto por hipótesis expresadas en términos de la integral de Sugeno. Formalmente, $\mathcal{H}$ está formado por las funciones $h : \mathcal{X} \subset \mathbb{R}^n \to [0,1]$ tales que:

$$h(x) = \mathbb{I}\left(Sg_\mu(f(x)) \geqslant \beta\right), \tag{3.4}$$

donde $x = (x_1, \ldots, x_n) \in \mathcal{X}$ representa una instancia con n atributos, $\mathbb{I}(\cdot)$ es la función indicatriz del subconjunto $A = \{x \in \mathcal{X} : Sg_\mu(f(x)) \geqslant \beta\}$ en $\mathcal{X}$:

$$\mathbb{I}(x) : \mathcal{X} \to [0,1], \quad \mathbb{I}(x) = \begin{cases} 1, & \text{si } x \in A, \\ 0, & \text{si } x \notin A, \end{cases} \tag{3.5}$$

y f es la *función de evaluación local* en $\mathbb{R}^n$ cuyas componentes son funciones en $\mathbb{R}$ tales que evalúan localmente los atributos de x:

$$f : \mathbb{R}^n \to [0,1]^n, \quad f(x) = (f_1(x_1), \ldots, f_n(x_n)) = (u_1, \ldots, u_n). \tag{3.6}$$

La componente de evaluación local $f_i(\cdot)$, con $i = 1, \ldots, n$, es la función que evalúa localmente el atributo i-ésimo de la instancia x, x_i, y está definida como:

$$f_i : \mathbb{R} \to [0,1], \quad f_i(x_i) = u_i. \tag{3.7}$$

Así, la tarea fundamental de la función de evaluación local $f(\cdot)$ es estandarizar los valores de cada atributo i de las instancias en el intervalo unidad $[0,1]$ con el objetivo de realizar la clasificación correctamente.

La clasificación de un dato arbitrario $x = (x_1, \ldots, x_n) \in \mathcal{X}$ mediante el clasificador de Sugeno está compuesta por tres etapas:

1. En primer lugar, a partir de las funciones de evaluación local $f_i(\cdot)$ $(i = 1, \ldots, n)$, se transforma cada atributo x_i, respectivamente, obteniendo los correspondientes criterios locales $u = (u_1, \ldots, u_n)$ de dicha instancia:
$$f(x) = u \in [0,1]^n. \tag{3.8}$$

2. En segundo lugar, se obtiene el criterio global U representativo de los criterios locales $u_1, \ldots, u_n$ aplicando la integral de Sugeno:
$$Sg_\mu(u) = U \in [0,1]. \tag{3.9}$$

3. Por último, se le asigna la clase a la instancia x según si el criterio global U obtenido cumple el umbral de decisión β:
$$h(x) = \begin{cases} 1, & \text{si } U \geqslant \beta, \\ 0, & \text{si } U < \beta. \end{cases} \tag{3.10}$$

Analizando las etapas de la clasificación, observamos que el clasificador requiere en su diseño de tres componentes importantes: la función de evaluación local $f(\cdot)$, la medida μ y el umbral de decisión β. No obstante, estos pueden ser inicialmente desconocidos, en cuyo caso serán aprendidos en el proceso de aprendizaje del método de clasificación.

Respecto a la complejidad del algoritmo, teniendo en cuenta la definición de la integral de Sugeno (3.1), necesitamos conocer el valor de la medida $\mu(A)$ para todo $A \subseteq [n]$. Más concretamente, necesitaríamos $\sum_{i=1}^{n} \binom{n}{i} - 1 = 2^n - 2$ valores de la medida y, para valores relativamente grandes de n, la complejidad del algoritmo crece de manera exponencial. Con el fin de evitar este problema, se propone tomar medidas k-maxitivas de tal forma que serían necesarios únicamente $\sum_{i=1}^{k} \binom{n}{i}$ valores. Esto es debido a que a partir de las medidas de los subconjuntos $A \subseteq [n]$ tales que $|A| \leqslant k$, podemos obtener el resto. Además, si tomamos valores moderados para k, el número de valores de la medida necesarios es notablemente inferior a $2^n - 2$. Por tanto, si queremos diseñar un clasificar más óptimo, debemos especificar en su diseño el valor de k de la maxitividad de la medida.

Por otro lado, destacamos que el umbral de decisión β también puede establecerse previamente en el diseño del clasificador, evitando así su aprendizaje. Sin embargo, observamos que, a diferencia del valor k de la maxitividad de la medida, si el valor de β no es introducido en el diseño del clasificador, es necesario aprenderlo. El motivo es sencillo, el clasificador puede ser diseñado y aprender con medidas que no son k-maxitivas, pero no puede clasificar sin el umbral de decisión. Así, definimos el conjunto de *hiperparámetros*

del clasificador, formado por: el parámetro k de la k-maxitividad de la medida y el valor del umbral de decisión β. Además, a dicho conjunto le añadimos el parámetro del margen del umbral denotado ρ, parámetro que se explicará a lo largo de los procesos de aprendizaje.

En cuanto a las técnicas aplicadas para aprender los componentes del clasificador, encontramos la *interpolación lineal* en el aprendizaje de la función de evaluación $f(\cdot)$, y los *problemas de programación lineal* o *problemas lineales* en el aprendizaje de la medida μ y en el del umbral de decisión β.

3.3. Proceso de Aprendizaje

Como hemos visto, la especificación del clasificador de Sugeno está formada por tres componentes: f, μ y β. En consecuencia, el proceso de aprendizaje del clasificador de Sugeno está dividido en tres subprocesos importantes:

- Proceso de aprendizaje de la función de evaluación f.
- Proceso de aprendizaje del umbral de decisión β.
- Proceso de aprendizaje de la medida μ.

El proceso de aprendizaje del umbral de decisión, a diferencia de los otros procesos, es necesario únicamente si su valor no se introduce en el diseño del clasificador, en cuyo caso se omitiría su aprendizaje. Por ello, decimos que β es un hiperparámetro del clasificador. Resaltamos que el orden de los subprocesos de aprendizaje no es arbitrario, ya que en cada uno son necesarios ciertos parámetros aprendidos. El proceso de aprendizaje de f requiere exclusivamente de los atributos de las instancias, por lo que se realiza el primero. En segundo lugar, se aprende el umbral de decisión, puesto que su proceso de aprendizaje necesita los criterios locales de los atributos y, por tanto, la función de evaluación f. Finalmente, el proceso de aprendizaje de la medida μ se realiza al final puesto que requiere tanto de la función f como del valor del umbral de decisión β.

El conjunto de entrenamiento $\mathcal{D}$ considerado tiene la siguiente forma:

$$\mathcal{D} = \left\{ \left(x^{(i)}, y^{(i)} \right) \right\}_{i=1}^{m} \subset (\mathcal{X} \times \{0,1\})^m , \tag{3.11}$$

donde las instancias $x^{(i)}$, $i = 1, \dots m$, son independientes y están idénticamente distribuidas (i.i.d.) respecto a la medida de probabilidad $P : \mathcal{X} \times \{0,1\} \to \{0,1\}$.

Así, como vimos en el capítulo anterior, dado el conjunto $\mathcal{D}$, el objetivo del presente método de clasificación consiste en encontrar la regla de decisión $h \in \mathcal{H}$ que minimice el riesgo sobre $\mathcal{D}$. Esta regla es la denominada *clasificador de Sugeno*. En concreto, este método de clasificación utiliza en la función del riesgo la función de pérdida 0/1 definida por:

$$\begin{cases} \mathcal{L}(y, y') = 0, & \text{si } y = y', \\ \mathcal{L}(y, y') = 1, & \text{en otro caso.} \end{cases} \tag{3.12}$$

Dicha función $\mathcal{L}$ es una de las funciones de coste más conocidas y simplemente toma el valor 0 si se ha clasificado correctamente la instancia y 1 si no.

En las tres subsecciones siguientes desarrollamos los tres procesos de aprendizaje siguiendo el orden en el que se llevan a cabo.

3.3.1. Proceso de aprendizaje de la función de evaluación

La tarea fundamental de la función de evaluación $f = (f_1, \ldots, f_n)$ es unificar los atributos de las instancias en el intervalo unidad, es decir, transformar el atributo i-ésimo (x_i) de una instancia x en un criterio local u_i en el intervalo $[0, 1]$ aplicando la componente i-ésima de la función de evaluación f. Este proceso de aprendizaje no tiene en cuenta la clasificación de las instancias, por lo que se trata de aprendizaje no supervisado. Adicionalmente, sin pérdida de generalidad, asumimos que un atributo es mejor cuanto mayor sea su valor y, en consecuencia, la función f es monótona. En caso contrario, si un atributo fuese mejor cuanto menor valor tuviese, bastaría con invertir el signo de los atributos, es decir, $-x_i$, volviendo así al caso anterior.

La idea principal de este proceso de aprendizaje consiste en reemplazar los valores de cada atributo por su función de distribución, considerando cada atributo como una variable aleatoria que puede tomar cualquier valor. De esta forma, tenemos una función de distribución por cada atributo que denotamos por $f_i(\cdot)$. Luego el objetivo es sustituir $x_i^{(j)}$ por $f_i\left(X_i \leqslant x_i^{(j)}\right)$, donde $i = 1, \ldots, n$, $j = 1, \ldots, m$, X_i es el atributo i de las instancias de $\mathcal{D}$ $\left(x^{(1)}, \ldots, x^{(m)}\right)$ y P la probabilidad subyacente de cada atributo. Sin embargo, dado que la función de probabilidad teórica es desconocida, utilizamos en su lugar la función de distribución empírica, deducida a partir de los datos del conjunto de entrenamiento $\mathcal{D}$. Entonces, vamos a definir la función de evaluación local f_i como una función de distribución y, para ello, distinguimos distintos casos.

En primer lugar, denotamos u_j al valor local representativo del atributo x_j definido de la siguiente manera:

$$u_j = \frac{1}{2}\left(\left\{ x_i^{\beta(k)},\ k = 1, \ldots, m :\ x_i^{\beta(k)} < x_i^{\beta(j)} \right\} \ + \ \left\{ x_i^{\beta(k)},\ k = 1, \ldots, m :\ x_i^{\beta(k)} \leqslant x_i^{\beta(j)} \right\} \right), \tag{3.13}$$

donde $j = 1, \ldots, m$, i indica el atributo de cada instancia evaluado y $\beta(\cdot)$ es una permutación sobre $[m]$ tal que $x_i^{\beta(1)} \leqslant x_i^{\beta(2)} \leqslant \ldots x_i^{\beta(m)}$, es decir, una permutación que ordena los valores del atributo i de las instancias de $\mathcal{D}$. Por la definición de la permutación $\beta(\cdot)$, observamos que el mínimo valor conocido del atributo i es $x_i^{\beta(1)}$ y el máximo $x_i^{\beta(m)}$. De esta forma, la función de distribución f_i toma exactamente los valores $u_1, \ldots, u_m$ en cada valor $x_i^{(1)}, \ldots, x_i^{(m)}$, respectivamente.

Ahora, para que la función f_i sea una función de distribución bien definida, queda por definir su imagen en el resto de posibles valores. Para ello, distinguimos tres casos. Si $y < x_i^{\beta(1)}$, $f(y) = 0$; si $y > x_i^{\beta(m)}$, $f(y) = 1$; y si $x_i^{\beta(1)} < y < x_i^{\beta(m)}$, calculamos $f(y)$ mediante la interpolación lineal de los dos valores conocidos del atributo i más cercanos al valor y.

En definitiva, tenemos que la función de evaluación local f es la función cuyas componentes f_i son la función de distribución del atributo i de las instancias y que, combinando los casos anteriores, está definida por:

$$\begin{cases} f_i(y) = u_i, & \text{si } y = x_i^{\beta(m')}, \quad \text{para algún } m' = 1, \ldots, m, \\ f_i(y) = 0, & \text{si } y < x_i^{\beta(1)}, \\ f_i(y) = 1, & \text{si } y > x_i^{\beta(m)}, \\ f_i(y) = interp\left(x_i^{\beta(j)}, x_i^{\beta(k)}\right) & \text{si } x_i^{\beta(1)} < y < x_i^{\beta(m)}, \end{cases} \tag{3.14}$$

donde $x_i^{\beta(j)}$ es el mayor valor tal que $x_i^{\beta(j)} < y$ y $x_i^{\beta(k)}$ es el menor valor tal que $y < x_i^{\beta(k)}$.

En Algoritmo 3.1 presentamos el pseudocódigo del proceso de aprendizaje de la función de evaluación local que acabamos de desarrollar.

input : Conjunto de entrenamiento $x^{(1)}, \ldots, x^{(m)}$

output: Función de evaluación $f = (f_1, \ldots, f_n)$

result = lista con las componentes de la función de evaluación function
for $x_j^{(i)}$ *en* $x_j^{(1)}, \ldots, x_j^{(m)}$ **do**
  $[u_1, \ldots, u_m]$ = transformamos $[x_j^{(1)}, \ldots, x_j^{(m)}]$ según la definición u_j
  $[u_1, \ldots, u_m]$ sin repeticiones
  Definimos $f_j(z)$ como:
  if $z = x_j^{(i)}$ *para algún* $i = 1, \ldots, m$ **then**
    $f_j(z) = u_j$
  else if $z < x_j^{(i)}$ *para todo* $i = 1, \ldots, m$ **then**
    $f_j(z) = 0$
  else if $z > x_j^{(i)}$ *para todo* $i = 1, \ldots, m$ **then**
    $f_j(z) = 1$
  else
    $f_j(z) =$ `interp` $\left(x_j^{(i1)}, x_j^{(i2)}\right)$ (donde $x_j^{(i1)}, x_j^{(i2)}$ son los valores inferior y superior más cercanos a z
  end
  Añadimos a result la definición de la componente f_j
end

Algoritmo 3.1: Pseudocógido del proceso de aprendizaje de la función de evaluación.

3.3.2. Proceso de aprendizaje del umbral de decisión

El clasificador de Sugeno clasifica las instancias en función de si cumplen o no el umbral de decisión. Como hemos visto, el umbral de decisión puede ser impuesto en el diseño del clasificador si se especifica en los parámetros iniciales del mismo, en cuyo caso no haría falta realizar el presente aprendizaje.

El proceso de aprendizaje del umbral de decisión β consiste en optimizar un problema lineal (PL) para el que existen numerosos métodos de programación lineal capaces de resolverlo de forma óptima. En concreto, hacemos uso de una de las librerías más conocidas, la librería *PuLP* de Python, utilizada para modelar y resolver problemas de optimización, como problemas de maximización o minimización, mediante programación lineal. El tipo de problema de optimización se indica especificando el parámetro `sense` al inicializar el problema: `LpMinimize` si es de minimización y `LpMaximize` si es de maximización. Además, esta librería incluye soporte para los elementos básicos de un problema de optimización: variables, restricciones y función objetivo. Destacamos que, dado que se trata de un problema lineal, tanto las restricciones como la función objetivo tienen que ser combinación lineal de las variables definidas en el problema, es decir, expresiones lineales. Para detectar los errores de clasificación, utilizamos el concepto de *margen de error* definido en las máquinas de vector soporte (véase 2.1.1). Así, el objetivo del PL es calcular el valor del umbral β tal que minimice los errores de clasificación de las instancias del conjunto de entrenamiento $\mathcal{D}$, denotados ζ_i, con $i = 1, \ldots, m$.

Teniendo en cuenta la relación de la integral de Sugeno y la mediana, la idea principal de este proceso está basada en asumir que los valores de la medida no afectan al cálculo de la integral, luego no afectan al cálculo de la mediana de los valores. Formalmente, suponemos que los valores de la medida cumplen:

$$\begin{aligned} Sg_\mu(f(x)) &= mediana\left(f_1(x_1), \ldots, f_n(x_n), \mu\left(A_{\sigma(2)}\right), \ldots, \mu\left(A_{\sigma(n)}\right)\right) \\ &= mediana\left(f_1(x_1), \ldots, f_n(x_n)\right), \end{aligned} \tag{3.15}$$

donde $f(\cdot)$ es la función de evaluación local aprendida. De esta forma, cualquier instancia $\left(x^{(i)}, y^{(i)}\right) \in \mathcal{D}$ tiene que satisfacer:

$$\begin{cases} Sg_\mu(f(x^{(i)})) = mediana\left(f_1\left(x_1^{(i)}\right), \ldots, f_n\left(x_n^{(i)}\right)\right) + \zeta_i \geqslant \beta, & \text{si } y^{(i)} = 1, \\ Sg_\mu(f(x^{(i)})) = mediana\left(f_1\left(x_1^{(i)}\right), \ldots, f_n\left(x_n^{(i)}\right)\right) - \zeta_i < \beta, & \text{si } y^{(i)} = 0. \end{cases} \tag{3.16}$$

Además, si estudiamos los posibles valores que toma esta mediana, vemos que los valores pertenecen al intervalo unidad, luego el valor aprendido de β se encuentra en el intervalo [0,1]. Por tanto, el PL que ajusta el umbral de decisión del clasificador a los datos de entrenamiento es:

$$minimizar \sum_{i=1}^{m} \zeta_i \quad \text{sujeto a} \quad \begin{cases} mediana\left(f_1\left(x_1^{(i)}\right), \ldots, f_n\left(x_n^{(i)}\right)\right) + \zeta_i \geqslant \beta, \text{ si } y^{(i)} = 1, \\ mediana\left(f_1\left(x_1^{(i)}\right), \ldots, f_n\left(x_n^{(i)}\right)\right) - \zeta_i < \beta, \text{ si } y^{(i)} = 0, \end{cases} \tag{3.17}$$

para todo $\left(x^{(i)}, y^{(i)}\right) \in \mathcal{D}$.

En Algoritmo 3.2 presentamos el pseudocódigo del proceso de aprendizaje del umbral de decisión que desarrollamos en esta sección.

input : Conjunto de entrenamiento $\left(x^{(1)}, y^{(1)}\right), \ldots, \left(x^{(m)}, y^{(m)}\right)$ y función de evaluación f,

output: Valor del umbral obtenido β

```
# Inicialización del PL
 linprob = problema de minimización PL
 beta = variable del umbral de decisión
 slack_variables = lista de los errores de clasificación ζ_1, ..., ζ_m

# Restricciones a las que está sujeto el PL
for (x^(i), y^(i)) ∈ D do
    # Clase positiva
    if y^(i) == 1 then
        Añadimos a linprob la restricción:
        mediana(f_1(x_1^(i)), ..., f_n(x_n^(i))) + ζ_i ⩾ β
    end
    # Clase negativa
    else
        Añadimos a linprob la restricción:
        mediana(f_1(x_1^(i)), ..., f_n(x_n^(i))) − ζ_i < β
    end
end

#Función objetivo del PL
 objective_function = Σ_{i=1}^{m} ζ_i
```

Algoritmo 3.2: Pseudocógido del proceso de aprendizaje del umbral de decisión.

3.3.3. Proceso de aprendizaje de la medida

La integral de Sugeno, base del clasificador de Sugeno, está definida respecto a una medida difusa y, por tanto, su proceso de aprendizaje es indispensable para el correcto funcionamiento del método de clasificación. Comenzamos recordando que el clasificador es la función h de la clase del clasificador que minimiza el riesgo empírico $\mathcal{R}(h)$. Sin embargo, la probabilidad P de la función de riesgo es desconocida y, al igual que en la sección anterior, lo sustituimos por el riesgo empírico. Teniendo en cuenta que el riesgo de un clasificador es la esperanza de la función de pérdida, en primer lugar planteamos el problema que minimice dicha función. El problema de esta primera aproximación es la dificultad elevada que presenta este PL y, en consecuencia, podemos no encontrar una solución óptima del problema. Por este motivo, plantemos un PL análogo al del proceso de aprendizaje anterior.

En primer lugar, antes de definir el PL, introducimos el parámetro *margen del umbral* ρ, parámetro útil si queremos un umbral de decisión más ajustado a cada clase. El valor de ρ se lo añadimos al umbral de decisión en la clase positiva y restamos al umbral de decisión en la clase negativa. Así, el clasificador asignaría la clase a cada instancias según los siguientes umbrales de decisión:

$$\begin{aligned} \beta^+ &= \beta + \rho \quad \text{de la clase positiva,} \\ \beta^- &= \beta - \rho \quad \text{de la clase negativa.} \end{aligned} \tag{3.18}$$

Luego el margen ρ es un parámetro que introducimos al diseñar el clasificador y, al igual que el hiperparámetro $\mathtt{k}$ de la k-maxitividad, si no se introduce, no se aprende y no se tiene en cuenta. Además, asumimos de antemano que el conjunto de entrenamiento no se puede separar linealmente y hacemos uso de nuevo del concepto de margen de error.

En segundo lugar, definimos el PL. Para ello, declaramos tantas variables como valores de la medida sean necesarios, así como un margen de error ξ_i por cada variable anterior. Una vez declaradas las variables, imponemos las restricciones que deben de cumplir los valores de la medida. Las primeras restricciones establecidas están relacionadas con la monotonicidad de la medida:

$$\forall A, B \subseteq [n],\ a \in [n] \setminus A \ \text{ tales que } \ B = A \cup \{a\},\ \mu(A) \leqslant \mu(B). \tag{3.19}$$

Las segundas restricciones las deducimos de la relación entre la integral de Sugeno y la mediana (3.3) junto con el margen de error. Como ya explicamos en la sección anterior, la integral de Sugeno la podemos obtener ordenando los $2n-1$ argumentos de la mediana, es decir, ordenando los n atributos de x junto con los $n-1$ valores $\mu\left(A_{\sigma(2)}\right), \ldots, \mu\left(A_{\sigma(n)}\right)$,

y tomando el que ocupa la posición n. De esta forma, analizando los atributos podemos reducir la complejidad del PL disminuyendo el número de valores de la medida que se tienen que aprender.

Por el diseño del clasificador de Sugeno y teniendo en cuenta el margen de error ξ_i, si una instancia $x^{(i)}$ del conjunto de entrenamiento $\mathcal{D}$ es de clase positiva ($y^{(i)} = 1$), entonces se tiene que cumplir que $Sg_\mu\left(x^{(i)}\right) + \xi_i \geqslant \beta^+$, o equivalentemente, que al menos n de los $2n-1$ valores $x_1, \ldots, x_n, \mu\left(A_{\sigma(2)}\right), \ldots, \mu\left(A_{\sigma(n)}\right)$ sean mayores o iguales que β. Denotamos l al número de atributos cuyo valor supera o iguala el umbral, es decir, $l_i = \left|\left\{x_j^{(i)} : x_j^{(i)} \geqslant \beta^+\right\}\right|$, donde $|\cdot|$ indica el cardinal del conjunto, y estudiamos los distintos casos posibles en función del valor de l_i:

- Si $l_i = 0$, ningún atributo de $x^{(i)}$ cumple el umbral de decisión β^+ y, por tanto, no se añade ninguna restricción ya que, independientemente de los $n-1$ valores de la medida, es imposible que la mediana satisfaga el umbral. En este caso, lo más probable es que obtengamos un valor ligeramente elevado en el error de clasificación ξ_i.
- Si $l_i < n$, es necesario que al menos $n - l$ de los valores de las medidas superen o igualen a β^+ para que se cumpla que $mediana\left(x_1^{(i)}, \ldots, x_n^{(i)}, \mu\left(A_{\sigma(2)}\right), \ldots, \mu\left(A_{\sigma(n)}\right)\right) \geqslant \beta^+$. Por definición de medida sabemos que $\mu\left(A_{\sigma(n)}\right) \leqslant \ldots \leqslant \mu\left(A_{\sigma(2)}\right)$, dado que $A_{\sigma(n)} \subseteq \ldots \subseteq A_{\sigma(2)}$. Por tanto, basta añadir la restricción:
$$\mu\left(A_{\sigma(n-l_i+1)}\right) + \xi_i \geqslant \beta^+. \tag{3.20}$$
- Si $l_i = n$, todos los atributos de la instancia son mayores o iguales que β^+ y la mediana siempre va a ser superior o igual que β^+, por lo que no es necesaria ninguna restricción.

Por el contrario, si la instancia es de clase negativa ($y^{(i)} = 0$), se tiene que cumplir que $Sg_\mu\left(x^{(i)}\right) - \xi_i < \beta^-$, es decir, que al menos n de los $2n-1$ valores sean menores que β. Análogamente, suponemos $y^{(i)} = 0$, denotamos $l_i = \left|\left\{x_j^{(i)} : x_j^{(i)} < \beta^-\right\}\right|$, y distinguimos los posibles casos en función de l_i:

- Si $l_i = 0$, no se añade ninguna restricción y utilizamos el mismo argumento que en el caso anterior.
- Si $l_i < n$, basta añadir la restricción:
$$\mu\left(A_{\sigma(l_i+1)}\right) - \xi_i \geqslant \beta^-. \tag{3.21}$$

- Si $l_i = n$, no se añade ninguna restricción puesto que automáticamente se satisface la desigualdad.

Por último, declaramos la función objetivo del PL definida como el sumatorio de los márgenes de error:

$$\sum_{i=1}^{m} \xi_i. \tag{3.22}$$

En definitiva, el PL considerado está definido por:

$$minimizar \sum_{i=1}^{m} \xi_i \quad \text{sujeto a} \begin{cases} \mu(A) \leqslant \mu(A \cup \{a\}), & \text{para todo } A \subseteq [n],\ a \in [n] \setminus A, \\ \mu\left(A_{\sigma(n-l_i+1)}\right) + \xi_i \geqslant \beta^+, & \text{si } y^{(i)} = 1, \\ \mu\left(A_{\sigma(l_i+1)}\right) - \xi_i < \beta^-, & \text{si } y^{(i)} = 0, \end{cases} \tag{3.23}$$

para todo $\left(x^{(i)}, y^{(i)}\right) \in \mathcal{D}$.

Por otro lado, con el objetivo de optimizar el método de clasificación, podemos imponer que las medidas sean k-maxitivas introduciendo el valor de k al diseñar el clasificador de Sugeno. En primer lugar, recordamos que una medida es k-maxitiva si para todo $A \subseteq [n]$ tal que $|A| > k$, $\mu(A) = \bigvee_{B \subset A} \mu(B)$, o equivalentemente, $\mu(A) = \max\{\mu(B) : B \subset A\}$. Por este motivo, observamos que en este caso serían necesarios menos valores de la medida y, por tanto, las segundas restricciones son ligeramente distintas a las del PL anterior (3.23).

Suponemos que $\left(x^{(i)}, y^{(i)}\right) \in \mathcal{D}$ es de clase positiva y analizamos la restricción anterior (3.20) ya que es la única que se ve afectada. Ahora, para garantizar $Sg_\mu\left(x^{(i)}\right) \geqslant \beta^+$, distinguimos dos casos en función del cardinal del conjunto $A_{\sigma(n-l_i+1)}$:

1. Si $|A_{\sigma(n-l_i+1)}| \leqslant k$, basta con añadir la restricción anterior:
$$\mu\left(A_{\sigma(n-l_i+1)}\right) \geqslant \beta^+. \tag{3.24}$$
2. Si $|A_{\sigma(n-l_i+1)}| > k$, necesitamos que la medida de al menos un subconjunto propio del conjunto $A_{\sigma(n-l_i+1)}$ sea mayor o igual que β^+, es decir, que $\mu(A) \geqslant \beta^+$, para algún $A \subset A_{\sigma(n-l_i+1)}$. Sin embargo, esta restricción es difícil de linealizar debido a que presenta una disyunción de condiciones, por lo que no podemos añadirla al PL directamente. En su lugar, dado que se trata de disyunciones, tomamos arbitrariamente un subconjunto propio $A \subset A_{\sigma(n-l_i+1)}$ tal que $|A| = k$ y añadimos al PL la siguiente restricción:
$$\mu(A) + \xi_i \geqslant \beta^+. \tag{3.25}$$

Por el contrario, suponemos que $\left(x^{(i)}, y^{(i)}\right) \in \mathcal{D}$ es de clase negativa. Análogamente, teniendo en cuenta la restricción anterior (3.21) para asegurar que $Sg_\mu\left(x^{(i)}\right) < \beta^-$ necesitamos añadir las siguientes restricciones:

$$\mu(A) - \xi_i < \beta^-, \quad \forall A \subset A_{\sigma(l_i+1)} : |A| = k. \tag{3.26}$$

Así, el PL planteado en el proceso de aprendizaje de la medida para el caso de medidas k-maxitivas está definido como:

$$minimizar \sum_{i=1}^{m} \xi_i \quad \text{sujeto a} \quad \begin{cases} \mu(A) \leqslant \mu(A \cup \{a\}), & \forall A \subseteq [n],\ a \in [n] \setminus A, \\ \mu(A) + \xi_i \geqslant \beta^+, & A \subset A_{\sigma(l_i+1)} : |A| = k, \text{ si } y^{(i)} = 1, \\ \mu(A) - \xi_i < \beta^-, & \forall A \subset A_{\sigma(l_i+1)} : |A| = k, \text{ si } y^{(i)} = 0, \end{cases} \tag{3.27}$$

En Algoritmo 3.3 mostramos el pseudocódigo del proceso de aprendizaje desarrollado en esta sección que, debido al grado de abstracción, es válido también para el caso de medidas k-maxitivas. En este, denotamos por c_variables a los distintos valores que toma la medida en los distintos subconjuntos, es decir, c_variables $= [\mu(A), \mu(B), \ldots]$, donde $A, B \subset [n]$. Recalcamos que aunque no se observa explícitamente el uso del parámetro k en el pseudocódigo, dicho valor lo tenemos en cuenta en las segundas restricciones al calcular la lista de subconjuntos cuya medida queremos deducir, es decir, la lista `feature_subsets`. En el caso de medidas k-maxitivas, la lista de `feature_subsets` está formada generalmente por menos subconjuntos, luego como cabía esperar, se añaden menos restricciones al PL.

input : Conjunto de entrenamiento $\left(x^{(1)}, y^{(1)}\right), \ldots, \left(x^{(m)}, y^{(m)}\right)$, función de evaluación f, umbral de decisión β, margen ρ y valor k de la maxitividad

output: Valores de la medida c_variables

#Inicialización del PL

linprob = problema de minimización PL
number_of_c_variables = numero de subconjuntos en $[n]$
c_variables = lista de las variables $c\_variables_1, \ldots$
slack_variables = lista de los errores de clasificación ξ_i

Restricciones 1 a las que está sujeto el PL. Monotonicidad de la medida

for $A, B : A = B \cup a,\ A, B \subset [n],\ a \in [n]$ **do**
  Añadimos restricción: c_variables$(A) \leqslant$ c_variables(B)
end

Restricciones 2 a las que está sujeto el PL. Mediana

for $\left(x^{(i)}, y^{(i)}\right) \in \mathcal{D}$ **do**
  Comprobamos que: | feature_subsets | $\leqslant k$
  feature_subsets = lista de subconjuntos cuya medida es necesario calcular
  slack_item = 1
  for $A \in$ feature_subsets **do**
    s_variable = variable de error de clasificación ξ_i
    # Clase positiva
    if $y^{(i)} == 1$ **then**
      Añadimos restricción: c_variables$(A) + \xi_i \geqslant \beta +\ \rho$
    end
    # Clase negativa
    else
      Añadimos restricción: c_variables$(A) - \xi_i \leqslant \beta -\ \rho$
    end
    slack_variables += s_variable
    slack_item += 1
  end
end

#Función objetivo del PL

objective_function = $\sum_{i=1}^{m} \xi_i$

Algoritmo 3.3: Pseudocógido del proceso de aprendizaje de la medida.

4

CLASIFICADORES PROPUESTOS

En este trabajo proponemos desarrollar clasificadores generalizados de Sugeno, es decir, clasificadores binarios basados en una extensión de la integral de Sugeno o integral de Sugeno generalizada. En concreto, hacemos uso de ciertas funciones que han sido utilizadas en la literatura anteriormente. No obstante, antes de comenzar a diseñar los algoritmos, es necesario estudiar las composición de las extensiones de la integral de Sugeno.

Este capítulo está dividido en dos secciones. En la primera, mostramos las definiciones de las extensiones de la integral de Sugeno y, en la segunda, diseñamos los nuevos métodos de clasificación binaria.

4.1. Extensiones de la Integral de Sugeno

Como vimos en la Sección 3.1, debido a la naturaleza de la integral de Sugeno, podemos interpretarla de distintas maneras, obteniendo así diversas expresiones para la misma. Además de las estudiadas, otra interpretación importante consiste en interpretarla como un funcional, es decir, como una función $\mathcal{A} : [0,1]^n \to [0,1]$ definida por:

$$A(u) = A(u_1, \ldots, u_n) = G\left(F\left(u_{\sigma(1)}, \mu\left(A_{\sigma(1)}\right)\right), \ldots, F\left(u_{\sigma(n)}, \mu\left(A_{\sigma(n)}\right)\right)\right), \tag{4.1}$$

donde $F : [0,1] \times [0,1] \to [0,1]$, $G : [0,1]^n \to [0,1]$ y μ es una medida difusa simétrica. De hecho, tomando $G(x_1, \ldots, x_n) = \bigvee_{i=1}^{n} x_i$ y $F(x,y) = x \wedge y$, obtenemos la definición de la integral de Sugeno clásica (3.1):

$$A(u) = \bigvee_{i=1}^{n} \left(u_{\sigma(i)} \wedge \mu\left(A_{\sigma(i)}\right)\right), \tag{4.2}$$

De la misma forma, tomando distintas funciones de agregación como $F(\cdot)$ y $G(\cdot)$, se definen las extensiones de la integral de Sugeno. En concreto, vamos a utilizar como $G(\cdot)$ la función suma n-aria y como $F(\cdot)$, diversas funciones que han sido estudiadas en la literatura en el caso de la integral de Choquet en [10] y [11], así como otras que se han estudiado

en [4] en el caso de la integral de Sugeno. En la Tabla 4.1 mostramos las funciones binarias $F : [0,1]^2 \to [0,1]$ con las que vamos a trabajar.

Nombre	**Definición**
Producto algebraico	$F_P(x,y) = xy$
t-norma de Hamacher	$F_H(x,y) = \dfrac{xy}{x+y-xy}$
Lukasiewicz	$F_l(x,y) = \max\{0, x+y-1\}$
Mínimo	$F_M(x,y) = \min\{x,y\}$
Mínimo Nilpotente	$F_{MN}(x,y) = \begin{cases} \min\{x,y\}, & \text{si } x+y>1 \\ 0, & \text{otro caso} \end{cases}$

Tabla 4.1: T-Normas utilizadas

Observamos también que las funciones $F(\cdot)$ de la Tabla 4.1 son t-normas y, para comprobarlo, basta ver que satisfacen las propiedades propias de una t-norma:

- Conmutatividad: $F(x,y) = F(y,x)$, para todo $x, y \in [0,1]^2$,
- Monotonía: $F(x,y) \leqslant F(w,z)$, para todo $x, y, w, z \in [0,1]^2$ tales que $x \leqslant w$ y $y \leqslant z$,
- Asociatividad: $F(x, F(y,z)) = F(F(x,y), z)$, para todo $x, y, z \in [0, 1^2]$.

Así, escogiendo las distintas t-normas como $F(\cdot)$ en el funcional (4.1), obtenemos las extensiones de la integral de Sugeno. Destacamos que la primera extensión, la extensión del producto algebraico, ha obtenido muy buenos resultados en el ámbito de *Deep Learning*, por lo que ya posee un gran interés en el campo de la Inteligencia Artificial.

4.2. Clasificadores generalizados de Sugeno

El objetivo de este trabajo consiste en diseñar métodos de clasificación binarios basados en las extensiones de la integral de Sugeno, y comparar su rendimiento con el del clasificador de Sugeno (CS) desarrollado en el capítulo anterior. Denominamos *clasificador generalizado de Sugeno* a la mejor regla de decisión obtenida por un método de clasificación binario basado en una integral de Sugeno generalizada como función de agregación.

La implementación de los métodos de clasificación propuestos es análoga a la del clasificador de Sugeno y, respecto al conjunto de entrenamiento, consideramos el mismo que anteriormente (3.11). No obstante, omitimos cierta optimización puesto que la complejidad que conlleva excede los objetivos del trabajo. Así, el conjunto de hiperparámetros de los nuevos métodos está formado por: el margen del umbral ρ y el umbral de decisión β, cuya función es la misma que en el capítulo anterior. Asimismo, omitimos la implementación de los dos últimos métodos de clasificación debido a que la linealización de las t-normas respectivas posee una dificultad que excede los objetivos de este proyecto y que dejamos para una investigación futura.

En las siguientes subsecciones analizamos cada posible algoritmo de clasificación junto con su proceso de aprendizaje, dividido en tres subprocesos como en el clasificador de Sugeno. Respecto al proceso de aprendizaje de la función de evaluación, dado que en el algoritmo de clasificación de Sugeno es independiente de la función de agregación utilizada (véase Subsección 3.3.1), repetiremos dicho proceso en los nuevos clasificadores y, por ello, omitimos su explicación. Respecto a la implementación de los dos procesos de aprendizaje restantes, esta es análoga a la explicada en las Subsecciones 3.3.2 y 3.3.3, respectivamente, luego reducimos su explicación a lo necesario. Recalcamos que, basándonos en la expresión de las extensiones de la integral de Sugeno y en la definición de los clasificadores que se proponen ahora, el umbral de decisión se encuentra en el intervalo [0, #atributos], donde #atributos representa el número de atributos que contiene cada conjunto de datos. Esto es debido a que la integral de Sugeno está formada por máximos y mínimo, por lo que su valor máximo es 1; mientras que sus extensiones están formadas por la suma de t-normas, por lo que su valor máximo es el número de atributos del conjunto de datos (#atributos), es decir, la suma del valor máximo de los criterios locales que es 1.

Cabe destacar que, para resolver los problemas lineales de los procesos de aprendizaje, hacemos uso de la librería DOcplex de Python debido a la eficiencia de esta en la búsqueda de la solución óptima del PL. Esto se encuentra explicado con más detalle en el marco experimental del trabajo, Capítulo 5.

4.2.1. Clasificador generalizado de Sugeno basado en el producto algebraico

El primer método de clasificación binario que proponemos se basa en la primera extensión de la integral de Sugeno, el producto algebraico. Por ello, todas las posibles reglas de

decisión $h : \mathcal{X} \subset \mathbb{R}^n \to [0,1]$ están definidas como:

$$h(x) = \mathbb{I}\left(Sg_\mu^P(f(x)) \geqslant \beta\right) = \mathbb{I}\left(\sum_{i=1}^{n}\left(u_{\sigma(i)} \cdot \mu\left(A_{\sigma(i)}\right)\right) \geqslant \beta\right) \tag{4.3}$$

donde $\mathbb{I}(\cdot)$ es la función indicatriz del subconjunto $\{x \in \mathcal{X} : Sg_\mu^P(f(x)) \geqslant \beta\} \subset \mathcal{X}$ y $u \in [0,1]^n$ es la evaluación local de la instancia obtenida aplicando $f(x)$, función de evaluación definida de la misma manera que en CS (3.14).

Si definimos la extensión de la integral respecto a la medida constante e igual a $1/n$, donde n es la dimensión del espacio de las instancias $\mathcal{X}$, destacamos que se obtiene exactamente la media de los n criterios locales $u_1, \ldots, u_n$. De esta forma, encontramos una relación entre esta extensión y el estadístico de la media.

$$Sg_\mu^P(u) = \sum_{i=1}^{n}\left(u_{\sigma(i)} \cdot \frac{1}{n}\right) = \frac{1}{n}\sum_{i=1}^{n} u_{\sigma(i)} = media(u_1, \ldots, u_n). \tag{4.4}$$

Sin embargo, a diferencia de la relación entre la integral de Sugeno y la mediana, esta no es independiente de la medida y, por tanto, no es posible tenerla en cuenta para optimizar el diseño del clasificador.

En cuanto a los procesos de aprendizaje de este método, comenzamos describiendo el proceso del umbral de decisión. Como sabemos, este aprendizaje tiene lugar el segundo pero dado que el proceso de aprendizaje de la función de evaluación local es el mismo, lo omitimos. Por el mismo argumento que en el CS, suponemos que el valor de la medida es el valor neutro de la operación producto. En consecuencia, el PL del proceso de aprendizaje es el siguiente:

$$minimizar \sum_{i=1}^{m} \zeta_i \quad \text{sujeto a} \quad \begin{cases} \sum_{j=1}^{n} x_{\sigma(j)}^{(i)} + \zeta_i \geqslant \beta, & \text{si } y^{(i)} = 1, \\ \sum_{j=1}^{n} x_{\sigma(j)}^{(i)} - \zeta_i < \beta, & \text{si } y^{(i)} = 0, \end{cases} \tag{4.5}$$

donde $i = 1, \ldots, m$. Por otro lado, el proceso de aprendizaje de la medida μ requiere de un PL que minimiza la suma de los márgenes de error sujeto a las restricciones de monotonicidad de la medida y a un segundo tipo de restricciones que, en este caso, vienen impuestas por la definición de la extensión de la integral utilizada. Teniendo en cuenta los márgenes de error ξ_i, con $i = 1, \ldots, m$, donde m es el número de instancias que hay en el conjunto de entrenamiento, imponemos que la definición de la extensión satisfaga el umbral si la instancia es de clase positiva y, que no lo satisfaga, en caso contrario. Entonces, el PL del proceso de aprendizaje de la medida es el siguiente:

$$minimizar \sum_{i=1}^{m} \xi_i \quad \text{sujeto a} \begin{cases} \mu(A) \leqslant \mu(A \cup \{a\}), \text{ para todo } A \subseteq [n],\ a \in [n] \setminus A, \\ \sum_{j=1}^{n} x_{\sigma(j)}^{(i)} * \mu\left(A_{\sigma(j)}\right) + \xi_i \geqslant \beta^+, \quad \text{si } y^{(i)} = 1, \\ \sum_{j=1}^{n} x_{\sigma(j)}^{(i)} * \mu\left(A_{\sigma(j)}\right) - \xi_i < \beta^-, \quad \text{si } y^{(i)} = 0, \end{cases} \tag{4.6}$$

donde $\left(x^{(i)}, y^{(i)}\right) \in \mathcal{D}$. Fácilmente vemos que estas restricciones están directamente relacionadas con la definición de la presente integral de Sugeno generalizada, debido a que no hemos encontrado una relación con algún estadístico independiente a la medida.

4.2.2. Clasificador generalizado de Sugeno basado en Lukasiewicz

El segundo método de clasificación está basado en la t-norma de Lukasiewicz. Así, su espacio de hipótesis $\mathcal{H}$ está compuesto por las reglas de la siguiente forma:

$$h(x) = \mathbb{I}\left(Sg_{\mu}^{L}(u) \geqslant \beta\right) = \mathbb{I}\left(\sum_{i=1}^{n} \max\left\{0, u_{\sigma(i)} + \mu(A_{\sigma(i)}) - 1\right\} \geqslant \beta\right) \tag{4.7}$$

donde $\mathbb{I}(\cdot)$ es la función indicatriz del subconjunto $\{x \in \mathcal{X} : Sg_{\mu}^{L}(f(x)) \geqslant \beta\} \subset \mathcal{X}$ y $u = f(x) \in [0,1]^n$, función de evaluación definida de la misma manera que la del CS.

En cuanto a los procesos de aprendizaje, describimos en primer lugar el aprendizaje del umbral de decisión β. Para ajustar el mejor valor de β al conjunto de entrenamiento, planteamos un PL que minimize el número de errores de clasificación. De la misma manera que en CS, suponemos que la medida es el elemento neutro de la operación suma. Sin embargo, en este caso, la extensión devolvería siempre el elemento nulo 0, ya que los criterios locales u_i pertenecen al intervalo $[0,1]$ y, por tanto, la extensión devolvería el máximo entre 0 y $u_{\sigma(i)} - 1$, un valor negativo o nulo. Sin pérdida de generalidad y con el objetivo de evitar este inconveniente, invertimos el signo del segundo argumento de la función máximo de tal forma que siempre obtenemos un número positivo. Así, para el proceso de aprendizaje del umbral de decisión proponemos utilizar el siguiente clasificador subyacente:

$$h(x) = \mathbb{I}\left(\sum_{i=1}^{n} \left(1 - u_{\sigma(i)}\right) \geqslant \beta\right) \tag{4.8}$$

Luego el PL cuya resolución devuelve el valor óptimo de β es:

$$minimizar \sum_{i=1}^{m} \zeta_i \quad \text{sujeto a} \begin{cases} \sum_{j=1}^{n} \left(1 - x_j^{(i)}\right) + \zeta_i \geqslant \beta, \quad \text{si } y^{(i)} = 1, \\ \sum_{j=1}^{n} \left(1 - x_j^{(i)}\right) - \zeta_i \geqslant \beta, \quad \text{si } y^{(i)} = 0, \end{cases} \tag{4.9}$$

donde $\left(x^{(i)}, y^{(i)}\right) \in \mathcal{D}$ y ζ_i es el error de clasificación de la instancia $x^{(i)}$, con $i = 1, \ldots, m$.

Por otro lado, para aprender la medida consideramos otro PL que utiliza la definición de la t-norma. No obstante, observamos que la definición depende de la función no lineal $\text{máx}(\cdot)$ y, por tanto, es necesario linealizarla. Para ello, definimos las variables m_{ij} que denotan el máximo entre 0 y $x^{(i)}_{\sigma(j)} + \mu\left(A_{\sigma(j)}\right) - 1$ y $m_{ij} \in [0, 1]$, y las variables binarias y_{ij}, cuyo valor es 1 si $x^{(i)}_{\sigma(j)} + \mu\left(A_{\sigma(j)}\right) - 1 < 0$ y 0 en caso contrario.

Así, las restricciones que deben de satisfacer las nuevas variables para cumplir su definición son:

$$\begin{aligned} & x^{(i)}_{\sigma(j)} + \mu\left(A_{\sigma(j)}\right) - 1 \leqslant 1 - y_{ij}, \\ & m_{ij} \geqslant x^{(i)}_{\sigma(j)} + \mu\left(A_{\sigma(j)}\right) - 1, \\ & m_{ij} \leqslant 0 + (1 - y_{ij}), \\ & m_{ij} \leqslant x^{(i)}_{\sigma(j)} + \mu\left(A_{\sigma(j)}\right) - 1 + y_{ij}. \end{aligned} \tag{4.10}$$

Finalmente, el PL considerado es:

$$minimizar \sum_{i=1}^{m} \xi_i \quad \text{sujeto a} \begin{cases} \text{Restricciones (4.10)} \\ \sum_{j=1}^{n} m_{ij} + \zeta_i \geqslant \beta, \quad \text{si } y^{(i)} = 1, \\ \sum_{j=1}^{n} m_{ij} - \zeta_i \geqslant \beta, \quad \text{si } y^{(i)} = 0. \end{cases} \tag{4.11}$$

4.2.3. Clasificador generalizado de Sugeno basado en el mínimo

El tercer y último método de clasificación binario propuesto junto con su implementación, está basado en el mínimo. De esta forma, las posibles reglas de decisión $h : \mathcal{X} \subset \mathbb{R} \to [0, 1]$ están definidas por:

$$h(x) = \mathbb{I}\left(Sg^M_\mu(f(x)) \geqslant \beta\right) = \mathbb{I}\left(\sum_{i=1}^{n} \text{mín}\left\{u_{\sigma(i)}, \mu\left(A_{\sigma(i)}\right)\right\} \geqslant \beta\right) \tag{4.12}$$

donde $\mathbb{I}(\cdot)$ es la función indicatriz del subconjunto $\{x \in \mathcal{X} : Sg^M_\mu(f(x)) \geqslant \beta\} \subset \mathcal{X}$ y $u = f(x)$, función de evaluación definida como en el CS.

Respecto al proceso de aprendizaje del umbral de decisión, asumimos que la medida toma el elemento neutro de la función $\text{mín}(\cdot)$ en todo subconjunto. En consecuencia, el PL planteado tiene la siguiente forma:

$$minimizar \sum_{i=1}^{m} \zeta_i \quad \text{sujeto a} \begin{cases} \sum_{j=1}^{n} x_j^{(i)} + \zeta_i \geqslant \beta, & \text{si } y^{(i)} = 1, \\ \sum_{j=1}^{n} x_j^{(i)} - \zeta_i \geqslant \beta, & \text{si } y^{(i)} = 0, \end{cases} \tag{4.13}$$

donde $\left(x^{(i)}, y^{(i)}\right) \in \mathcal{D}$ y ζ_i es el error de clasificación de la instancia $x^{(i)}$, con $i = 1, \ldots, m$.

En cuanto al proceso de aprendizaje de la medida, por el mismo argumento que en el método anterior, necesitamos linealizar la función $\min(\cdot)$. Análogamente, definimos las variables $m_{ij} = \min\left(x_{\sigma(j)}^{(i)}, \mu\left(A_{\sigma(j)}\right)\right)$ tales que pertenecen al intervalo $[0, 1]$, y las variables binarias y_{ij}, cuyo valor es 0 si $x_{\sigma(j)}^{(i)} > \mu\left(A_{\sigma(j)}\right)$ y 1 en caso contrario. Luego el PL planteado es:

$$minimizar \sum_{i=1}^{m} \zeta_i \quad \text{sujeto a} \begin{cases} \mu\left(A_{\sigma(j)}\right) - u_{\sigma(j)}^{(i)} \leqslant y_{ij}, \\ m_{ij} \leqslant u_{\sigma(j)}^{(i)}, \\ m_{ij} \leqslant \mu\left(A_{\sigma(j)}\right), \\ m_{ij} \geqslant x_{\sigma(j)}^{(i)} - (1 - y_{ij}), \\ m_{ij} \geqslant \mu\left(A_{\sigma(j)}\right) - y_{ij}, \\ \sum_{j=1}^{n} m_{ij} + \zeta_i \geqslant \beta, & \text{si } y^{(i)} = 1, \\ \sum_{j=1}^{n} m_{ij} - \zeta_i \geqslant \beta, & \text{si } y^{(i)} = 0, \end{cases} \tag{4.14}$$

donde las 5 primeras restricciones definen las variables m_{ij} y y_{ij} definidas para linealizar la función del mínimo, y las 2 últimas imponen que se cumpla la restricción de la extensión de la integral de Sugeno al igual que en el resto de métodos propuestos.

4.2.4. Clasificador Generalizado de Sugeno basado en la t-norma de Hamacher

El cuarto método de clasificación propuesto hace uso la t-norma de Hamacher como función de agregación y, por tanto, las reglas de decisión $h : \mathcal{X} \subset \mathbb{R}^n \to [0, 1]$ están expresadas en términos de dicha t-norma, es decir, tienen la siguiente forma:

$$h(x) = \mathbb{I}\left(Sg_{\mu}^{H}(u) \geqslant \beta\right) = \mathbb{I}\left(\sum_{i=1}^{n} \frac{u_{\sigma(i)} \cdot \mu\left(A_{\sigma(i)}\right)}{u_{\sigma(i)} + \mu\left(A_{\sigma(i)}\right) - u_{\sigma(i)} \cdot \mu\left(A_{\sigma(i)}\right)} \geqslant \beta\right) \tag{4.15}$$

donde $\mathbb{I}(\cdot)$ es la función indicatriz del subconjunto $\{x \in \mathcal{X} : Sg_{\mu}^{H}(f(x)) \geqslant \beta\} \subset \mathcal{X}$ y $u = f(x)$, función de evaluación definida como en CS.

A pesar de que puede ser un clasificador interesante, este método de clasificación no ha sido implementado debido a que la linealización de la t-norma posee una dificultad que se encuentra fuera de los objetivos de este proyecto, y que dejamos para una investigación futura.

4.2.5. Clasificador generalizado de Sugeno basado en el mínimo nilpotente

El último método de clasificación propuesto utiliza el mínimo nilpotente como función de agregación, por lo que las posibles hipótesis del clasificador están expresadas en términos de este y tiene las siguiente forma:

$$h(x) = \mathbb{I}\left(Sg_{\mu}^{MN}(f(x)) \geqslant \beta\right), \tag{4.16}$$

donde $\mathbb{I}(\cdot)$ es la función indicatriz del subconjunto $\{x \in \mathcal{X} : Sg_{\mu}^{N}M(f(x)) \geqslant \beta\} \subset \mathcal{X}$, f la función de evaluación definida como en CS (3.14) y la función $Sg_{\mu}^{M}N(\cdot)$ viene dada por:

$$Sg_{\mu}^{MN}(u) = \begin{cases} \text{mín}\{u_{\sigma(i)}, \mu(A_{\sigma(i)})\}, & \text{si } u_{\sigma(i)} + \mu(A_{\sigma(i)}) > 1 \\ 0, & \text{otro caso} \end{cases}, \quad u \in [0,1]^n. \tag{4.17}$$

De la misma forma que en el método de clasificación anterior, la implementación de este clasificador la dejamos para investigación futuras debido a que su complejidad excede los objetivos del trabajo.

5

Estudio Experimental

En este capítulo presentamos el marco experimental en el que desarrollamos los experimentos junto con el análisis de los resultados de los mismos. En primer lugar, describimos los conjuntos de datos o datasets seleccionados para el estudio experimental, así como la metodología de evaluación llevada a cabo. En segundo lugar, realizamos el estudio experimental en 3 etapas siguiendo la metodología explicada.

5.1. Marco experimental

5.1.1. Conjuntos de datos

Con el fin de analizar el rendimiento de los algoritmos de clasificación propuestos, consideramos 8 conjuntos de datos reales, de los cuales 7 pertenecen al repositorio *UCI Machine Learning* y el restante lo he obtenido de [12]. El repositorio *UCI Machine Learning* contiene una amplia colección de bases de datos y datasets que son utilizados para el análisis empírico de algoritmos de aprendizaje automático. Más concretamente, consideramos conjuntos de datos en los que se puede suponer la monotonicidad en los atributos y que, además, han sido utilizados previamente en estudios de clasificación similares. La Tabla 5.1 resume las características de los conjuntos de datos seleccionados, indicando por cada uno la abreviación con la que los vamos a identificar (Id.), el nombre del conjunto de datos (Conjunto de datos), el número de muestras que contiene (# Instancias), el número de atributos de cada dato (# Atributos) y la fuente de donde ha sido obtenido (Fuente).

Referimos al lector al Artículo [2] para una descripción detallada de los conjuntos de datos: DBS, MMG, BC, BCW y HAB. Para el resto de datasets, referimos al propio repositorio: UCI BCC, UCI TRA y UCI CAE.

Id.	Conjunto de datos	# Instancias	# Atributos	Fuente
DBS	Den Bosch	119	8	[12]
MMG	Mammographic	961	5	UCI
BC	Breast Cancer	278	7	UCI
BCW	Breast Cancer Wisconsin	699	9	UCI
BCC	Breast Cancer Coimbra	116	9	UCI
HAB	Haberman's Survival	306	3	UCI
TRA	Blood Transfusion Service	748	4	UCI
CAE	Caesarian Section	80	5	UCI

Tabla 5.1: Resumen de los conjuntos de datos

5.1.2. Metodología de evaluación

Para realizar el estudio experimental utilizamos la conocida técnica de *validación cruzada de k particiones*. Este tipo de validación consiste en dividir el conjunto de datos en k subconjuntos y utilizar un subconjunto como conjunto de validación y los otros $k-1$ como conjunto de entrenamiento. Se realizan k iteraciones del proceso utilizando en cada una un subconjunto de validación distinto. Finalmente, se realiza la media aritmética de los resultados de cada iteración para obtener un único resultado. Este método es muy preciso puesto que evalúa el clasificador con distintas combinaciones de datos tanto de entrenamiento como de validación. En nuestro experimento, consideramos un modelo de validación cruzada de 5 particiones, es decir, particiones formadas por un 20 % de los datos, y empleamos una combinación de cuatro de ellas para entrenar el modelo. De esta forma, el conjunto de entrenamiento está formada por un 80 % de los datos y el de validación por un 20 %. El resultado de cada conjunto de datos se obtiene calculando la media aritmética de los resultados de las 5 iteraciones.

Adicionalmente, con el objetivo de comprobar el rendimiento de los diferentes métodos, empleamos la métrica más conocida: el *grado de precisión* del clasificador (*accuracy rate*). Esta métrica está definida como el porcentaje de muestras correctamente clasificadas en relación con el número total de muestras. Con el objetivo de otorgar soporte estadístico al análisis de los resultados, hacemos uso de la *prueba de rangos con signo de Wilcoxon* desarrollada en [13], una prueba no paramétrica que compara el rendimiento general del modelo contando el número de casos en los que el algoritmo clasifica los datos correctamente. En particular, utilizamos esta prueba para realizar las comparaciones por pares del clasificador de Sugeno con los clasificadores propuestos en este trabajo.

Por otro lado, en el estudio experimental del clasificador de Sugeno se emplea una validación cruzada interna de 10 particiones con distintos valores de k para determinar el mejor hiperparámetro k de la maxitividad. Sin embargo, en este trabajo hacemos uso de la clase *GridSearchCV* disponible en *scikit–learn*, ya que permite evaluar y seleccionar de forma sistemática los mejores parámetros de un modelo. Para ello, introducimos los posibles valores que puede tomar k que son exactamente $\{1, \ldots, \texttt{nfeatures} - 1\}$, donde `nfeatures` contiene el número de atributos del correspondiente dataset, en el atributo *params* de *GridSearchCV*. No obstante, añadimos un estudio en el que probamos con distintos valores posibles de k y, así, corroboramos el valor de k obtenido mediante *GridSearchCV*.

Finalmente, respecto a la implementación de los clasificadores, destacamos la utilización de la librería DOcplex de Python en el proceso de aprendizaje de la medida. DOcplex o *IBM Decision Optimization CPLEX Modeling for Python* es una librería desarrollado por IBM que permite resolver problemas de optimización desarrollado por IBM. A diferencia del proceso de aprendizaje la medida del clasificador de Sugeno, el proceso de aprendizaje de los algoritmos de clasificación basados tanto en el mínimo como en Lukasiewicz requieren de una cantidad elevada de restricciones. En estos casos, la libría DOcplex proporciona la solución óptima del problema de forma más eficiente y rápida. Por este motivo y para evitar discrepancias entre los clasificadores, hemos implementado el proceso de aprendizaje de la medida de los cuatro clasificadores que comparamos con DOcplex, incluyendo el de Sugeno. Recalcamos que hemos comprobado que el rendimiento del clasificador de Sugeno no varía al intercambiar la librería PuLP por la librería DOcplex, como cabía esperar.

5.2. Análisis de los resultados

En esta segunda sección desarrollamos el estudio experimental dividido en 3 etapas. En la primera etapa, analizamos la precisión del clasificador de Sugeno según el valor k de la maxitividad de la medida. En la segunda etapa, observamos el grado de precisión de los clasificadores en los diversos conjuntos de datos, y comparamos estadísticamente el rendimiento del clasificador de Sugeno con cada uno de los clasificadores generalizados mediante la prueba de rangos con signo de Wilcoxon. Finalmente, analizamos el efecto del valor del umbral de decisión en el rendimiento de los métodos de clasificación.

5.2.1. Hiperparámetro k en el clasificador de Sugeno

Como explicamos en el diseño del clasificador de Sugeno, el valor k de la maxitividad de la medida es un hiperparámetro del clasificador. Asimismo, recordamos que es el único hiperparámetro que, si no se especifica en el diseño del CS, se omite y que, dado que omitimos cierta optimización en los clasificadores propuestos, estos no disponen de este hiperparámetro. Por este motivo, resulta interesante estudiar el comportamiento del clasificador de Sugeno en función de dicho valor. La Tabla 5.2 muestra el porcentaje de aciertos del clasificador de Sugeno en función del valor de k en los distintos conjuntos de datos, y aparece resaltado el porcentaje más elevado de cada conjunto con el menor k posible. Cada columna de la tabla contiene tanto el identificador del conjunto como el número de atributos de sus instancias entre paréntesis. De esta manera, es sencillo ver qué valores de k puede tomar el hiperparámetro en cada caso ($k = 1, \ldots, \#\text{atributos} - 1$). Además, la tabla contiene en la última file el porcentaje de aciertos del clasificador sin tener en cuenta medidas k-maxitivas.

	DBS (8)	MMG (5)	BC (7)	BCW (9)	HAB (3)	BCC (9)	TRA (4)	CAE (5)
k = 1	52.8986	55.9036	33.8117	78.4929	**46.0338**	55.1812	**42.4966**	**57.5000**
k = 2	66.3406	**56.9880**	39.5909	91.0820	46.0338	56.8841	42.4966	57.5000
k = 3	70.5797	56.9880	**39.9481**	95.6151	46.0338	62.8986	42.4966	55.0000
k = 4	**71.4493**	56.9880	39.9481	**96.0509**	-	62.0652	42.4966	55.0000
k = 5	71.4493	56.9880	39.9481	95.7600	-	62.9348	-	55.0000
k = 6	71.4493	-	39.9481	95.7611	-	**64.6739**	-	-
–	71.4493	56.9880	39.9481	95.6129	46.0338	64.6739	42.4966	57.5000

Tabla 5.2: Precisión del clasificador de Sugeno en función del hiperparámetro k.

Analizando los resultados vemos que en casi todos los casos el porcentaje de aciertos del clasificador con el mejor valor de k coincide exactamente con el obtenido sin tener en cuenta medidas k-maxitivas (última fila de 5.2), excepto en BCW donde vemos una ligera variación. Por otro lado, destacamos que el valor óptimo de k observado en cada conjunto de datos coincide con el obtenido mediante la clase *GridSearchCV*. Por tanto, dado que tener en cuenta medidas k-maxitivas optimiza este método de clasificación además de mejorar los resultados, aprendemos el valor de k mediante validación cruzada en las siguientes pruebas.

5.2.2. Evaluación del rendimiento de las propuestas

Con el fin de comparar el rendimiento de los clasificadores propuestos junto con el clasificador de Sugeno, evaluamos cada clasificador mediante la técnica de validación cruzada de 5 particiones. La Tabla 5.3 muestra el grado de precisión obtenido por los clasificadores en los distintos conjuntos de datos y aparece resaltado el porcentaje de acierto más elevado de cada dataset. Las dos últimas filas muestran la media de los porcentajes de acierto de cada clasificador y el número de datasets en los que obtienen el mejor resultado, respectivamente. Denotamos *CS* al clasificador de Sugeno, *CSProd* al clasificador generalizado de Sugeno basado en el producto algebraico, *CSLuka* al basado en la norma de Lukasiewicz y *CSMin* al basado en el mínimo.

Dataset	**CS**	**CSProd**	**CSLuka**	**CSMin**
DBS	71.4493	46.8841	61.3043	**81.5580**
MMG	56.9880	71.8072	56.8675	**79.0361**
BC	39.9481	**68.6883**	68.3377	68.6688
BCW	**96.0509**	93.2739	93.5616	94.4440
HAB	46.0338	71.5653	**72.5489**	70.9096
BCC	64.6739	47.3913	62.2101	**69.8188**
TRA	42.4966	46.6702	**74.6031**	69.3808
CAE	57.5000	42.5000	57.5000	**61.2500**
Media(%)	59.3926	61.0975	68.3667	**74.3833**
#Datasets	1	1	2	4

Tabla 5.3: Precisión obtenida en el conjunto de prueba por cada clasificador.

Como podemos ver, en cada conjunto de datos obtiene el mejor resultado un clasificador distinto. Analizando estos resultados, observamos que el clasificador de Sugeno obtiene el mayor porcentaje de aciertos respecto al resto de clasificadores generalizados únicamente en el conjunto de datos BCW con una precisión de 96 %. El clasificador basado en el producto algebraico, obtiene el porcentaje más elevado en BC exclusivamente, mientras que el basado en Lukasiewicz, tanto en HAB como en TRA. Por su parte, el clasificador que mejores resultados logra en general es el clasificador generalizado basado en el mínimo. Esto lo apreciamos con claridad en la fila de la media, donde vemos que obtiene la media más elevada entorno al 74.38 % y, además, si nos fijamos en la última fila de la tabla vemos que obtiene los mejores resultados en 4 de los 8 conjuntos de datos evaluados. De forma intuitiva, afirmamos que el método de clasificación binario más competitivo es claramente el método basado en el mínimo como función de agregación.

Para confirmar este descubrimiento, llevamos a cabo varias comparaciones por pares mediante la *prueba de rangos con signo de Wilcoxon*. La Tabla 5.4 muestra los resultados de estas comparaciones, donde la columna $\boldsymbol{R}^+$ contiene los rangos a favor del CS y $\boldsymbol{R}^-$ a favor del clasificador generalizado comparado (*CSProd*, *CSLuka* y *CSMin*, respectivamente); la columna ***Hipótesis*** indica si se rechaza o no la hipótesis nula, es decir, si existen diferencias estadísticas entre los clasificadores, y ***p-valor*** indica el valor de la diferencia estadística a favor del clasificador con rango más alto. Además, el *p-valor* se encuentra resaltado cuando existen diferencias estadísticas significativas, es decir, p-valores inferiores a 0.05.

	$\mathbf{R}^+$	$\mathbf{R}^-$	**Hipótesis**	**p-valor**
CSProd	16	20	No Rechazada	0.8438
CSLuka	14.5	21.5	No Rechazada	0.5781
CSMin	1	35	Rechazada	**0.0156**

Tabla 5.4: Prueba de rangos con signo de Wilcoxon que compara el rendimiento de CS con CSProc, CSLuka y CSMin.

De acuerdo con estos resultados, el rendimiento proporcionado por los dos primeros clasificadores, *CSProd* y *CSLuka*, no presenta diferencias estadísticas con el rendimiento de CS. Por su parte, la tercera comparación obtiene un p-valor inferior a 0.05, por lo que afirmamos que CSMin es mejor que CS con más de un 95 % de confianza. De esta forma, apoyamos la afirmación intuitiva anterior.

A continuación, resulta interesante comparar los clasificadores propuestos entre ellos y, para ello, realizamos la prueba de rangos con signo de Wilcoxon comparándolos por pares, cuyso resultados se muestran en la Tabla 5.5.

	$\mathbf{R}^+$	$\mathbf{R}^-$	**Hipótesis**	**p-valor**
CSProd vs. CSLuka	8	28	No rechazada	0.1953
CSProd vs. CSMin	3	33	Rechazada	**0.0391**
CSLuka vs. CSMin	8	28	No rechazada	0.1953

Tabla 5.5: Prueba de rangos con signo de Wilcoxon que compara el rendimiento de los clasificadores propuestos.

Como cabía esperar, estas comparaciones confirman que CSMin es mejor que CSProd con un 95 % de confianza y, además, por al p-valor obtenido en las 2 comparaciones restantes, apreciamos que hay una tendencia a que CSMin sea mejor que CSLuka y a que CSLuka sea mejor que CSProd. Resultados que corroboran los resultados mostrados en 5.3.

En consecuencia, con estas dos pruebas de rangos apoyamos estadísticamente la afirmación intuitiva anterior deduciendo que el clasificador generalizado de Sugeno basado en el mínimo es el que obtiene los resultados más prometedores, seguido del basado en Lukasiewicz. La razón principal de estos resultados está directamente relacionada con la función de agregación utilizada en los métodos de clasificación. Recordamos que los métodos propuestos utilizan el sumatorio de t-normas, mientras que el método de clasificación basado en la integral de Sugeno utiliza el máximo de mínimos. Analizando los experimentos realizados hasta este momento, deducimos que el sumatorio como función $G(\cdot)$ del funcional proporciona mejores resultados y que, en cuanto a la función $F(\cdot)$, el mínimo es el que mejores resultados obtiene.

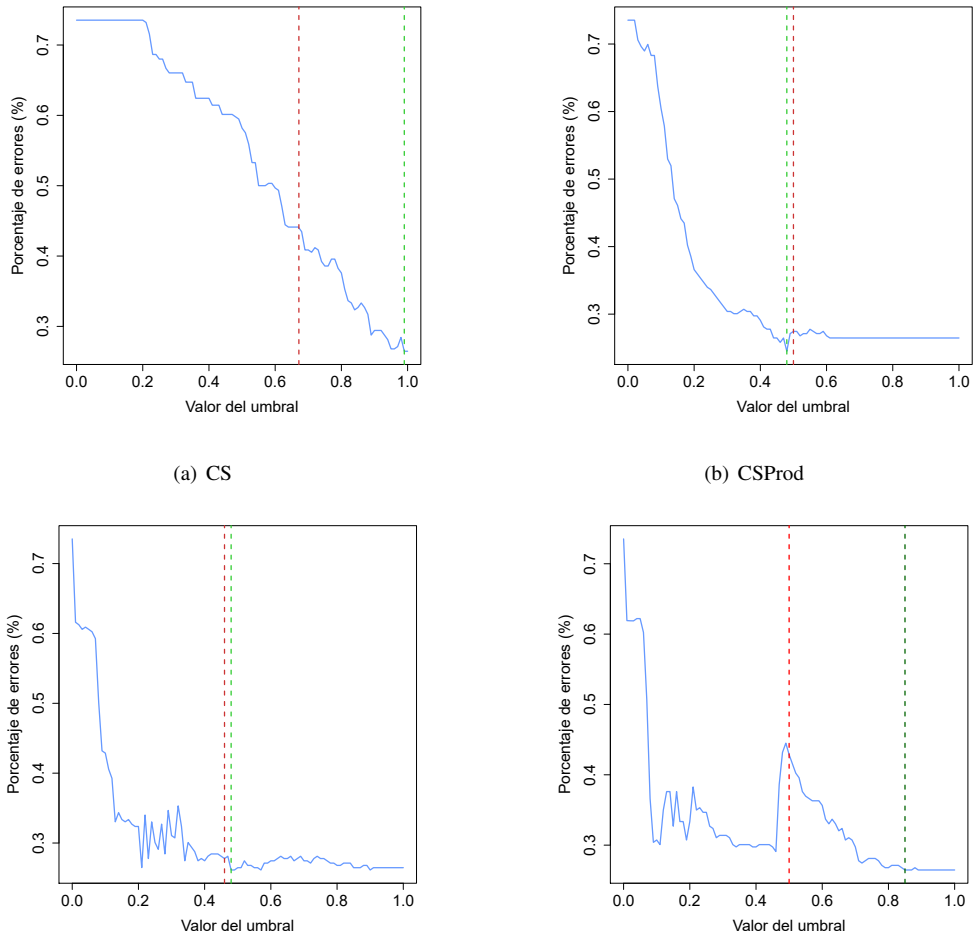
5.2.3. Efecto del umbral

Finalmente, el proceso de aprendizaje de la medida requiere conocer el valor del umbral de decisión, por lo que si el umbral es desconocido, es necesario realizar el proceso de aprendizaje del umbral primero. Esto nos lleva a preguntarnos cuál es el efecto real del umbral de decisión sobre la clasificación de los datos.

Con el fin de observar el efecto del umbral de decisión, realizamos las diversas evaluaciones con distintos valores en el umbral β y sobre el conjunto de datos *Haberman* (HAB), tanto del método de clasificación basado en la integral de Sugeno como de los métodos propuestos en este trabajo. En concreto, especificamos en el clasificador de Sugeno distintos valores comenzando en 0 y aumentándolo en 0.01 hasta 1, es decir, valores pertenecientes al intervalo (0, 0.01, 0.02,. . ., 1), mientras que en el caso de CSProd, CSMin y CSLuka utilizamos valores del umbral pertenecientes al intervalo (0, 0.03, 0.09,. . .,3). Esto es debido a que, como describimos en el Capítulo 4, el umbral de decisión de los métodos propuestos puede tomar cualquier valor entre 0 y el número de atributos del dataset, entre 0 y 3 en el conjunto de datos HAB. Además, escogemos pasos de 0.03 en 0.03 dentro de dicho intervalo para tomar el mismo número de valores que en la prueba del CS y mostrar todos los valores correctamente adaptados en el intervalo unidad. De esta forma, interpretamos mejor gráficamente los resultados.

Para una mayor precisión, utilizamos de nuevo la validación cruzada de 5 particiones tomando la media en cada valor del umbral. La Figura 5.1 representa la evolución del porcentaje de error de cada clasificador en función del valor del umbral en el intervalo unidad, así como en color rojo el valor del umbral aprendido con el proceso de aprendizaje del respectivo método de clasificación y, en color verde, el mejor valor de umbral deducido a

partir de los resultados obtenidos.



(a) CS

(b) CSProd

(c) CSLuka

(d) CSMin

Figura 5.1: Porcentaje de error en función del umbral de decisión en la clasificación sobre HAB junto con el mejor valor de β (línea verde) y el valor de β aprendido (línea roja).

En CS observamos que el porcentaje de errores del clasificador desciende prácticamente de forma constante a partir del 0.2 hasta alcanzar el mínimo porcentaje de error en $\beta = 1$. Tanto en CSProd como en CSLuka observamos una bajada notable del porcentaje de errores en cuanto comienza a crecer el valor del umbral, y una parte más estable a partir del umbral 0.5 en ambas gráficas. En el caso de CSProd destaca además la parte totalmente constante entre el 0.6 y el 0.8 entorno al 26.46 %. Por último, en CSMin observamos la estabilidad tras alcanzar el porcentaje de errores sobre el 0.85. Respecto a los umbrales aprendidos, tanto en CSProd como en CSLuka se encuentran bastante cerca, mientras que en CS y CSMin aparecen algo más distanciados.

Por tanto, vemos claramente que el umbral tiene un notable efecto en la clasificación y que, tomando el valor adecuado del umbral de decisión, podemos reducir el número de errores considerablemente.

6 Conclusiones y trabajo futuro

En este trabajo hemos ampliado el uso de la integral de Sugeno como función de agregación en los problemas de clasificación. Con el objetivo de mejorar los resultados obtenidos por el clasificador de Sugeno, hemos diseñado varios clasificadores basándonos en extensiones de la integral de Sugeno como función de agregación, obteniendo unos resultados muy prometedores. De hecho, haciendo uso de la prueba de rangos con signo de Wilcoxon, hemos determinado que el clasificador generalizado basado en el mínimo supera considerablemente la precisión del clasificador de Sugeno, con más de un 95 % de confianza. Para ser más exactos, analizando los resultados obtenidos a lo largo del estudio experimental en el Capítulo 5, concluimos que de los clasificadores propuestos implementados, el clasificador basado en el mínimo posee una precisión superior. No obstante, hemos podido apreciar que en los datasets utilizados, los 3 métodos propuestos obtenían un porcentaje de aciertos superior al de método basado en la integral de Sugeno. Estos resultados nos han permitido mostrar la importancia de la función de agregación del clasificador. Asimismo, hemos observado el efecto del parámetro k de la maxitividad de la medida en el caso del clasificador de Sugeno y el efecto del umbral de decisión en los 4 clasificadores evaluados.

En el futuro, y con intención de seguir ampliando esta línea de clasificadores basados en extensiones de la integral de Sugeno, deberían de implementarse las extensiones presentadas sin implementación, además de nuevas funciones como las que han sido utilizadas en el caso de la integral de Choquet en [10] y [11], diseñando así nuevos clasificadores que pueden resultar interesantes. Para ello, sería necesario abordar este tipo de problemas de optimización con librerías específicas para expresiones no lineales o realizar previamente el complejo proceso de linealización. De esta forma, se podrían realizar numerosos experimentos que nos ofrecerían más información acerca de cómo trabajan las extensiones de la integral de Sugeno como funciones de agregación. Además, incluso se podrían diseñar de tal forma que tengan en cuenta medidas k-maxitivas, optimizando así su proceso de aprendizaje. Por otro lado, se podría investigar con más detalle la existencia de relaciones entre las extensiones de la integral de Sugeno y estadísticos utilizados en el ámbito de la inteligencia artificial, como la que existe en el caso de la integral de Sugeno. Así, los métodos de clasificación propuestos llevarían a cabo un proceso de aprendizaje más óptimo.

BIBLIOGRAFÍA

[1] S. Abbaszadeh and E. Hüllermeier, "Machine learning with the sugeno integral: The case of binary classification," *IEEE Transactions on Fuzzy Systems*, vol. 1, pp. 1063–6706, July 2020.

[2] A. F. Tehrani, W. Cheng, K. Dembczynski, and E. Hüllermeier, "Learning monotone nonlinear models using the choquet integral," *Machine Learning*, p. 183– 211, October 2012.

[3] M. Grabisch and C. Labreuche, "A decade of application of the choquet and sugeno integrals in multi-criteria decision aid," *Annals of Operations Research*, vol. 175, pp. 1–44, April 2008.

[4] F. Bardozzo, B. D. L. Osa, L. Horanská, J. F. Idocin, M. delli Priscoli, L. Troiano, R. Tagliaferri, J. Fernandez, and H. Bustince, "Adaptive binarization based on fuzzy integrals," *arXiv*, March 2020.

[5] V. Vapnik and C. Cortes, "Support-vector networks," *Machine Leaming, 20*, pp. 273–297, 1995.

[6] B. Schölkopf and A. Smola, "Support vector machines and kernel algorithms," *Machine Leaming, 20*, pp. 273–297, March 2002.

[7] L. A. Zadeh, "Fuzzy sets," *Information and Control*, vol. 8, pp. 338–353, June 1965.

[8] M. Couceiro and J. L. Marichal, "Characterizations of discrete sugeno integrals as polynomial functions over distributive lattices," *Fuzzy Sets and Systems*, vol. 161, pp. 694–707, March 2010.

[9] R. J. Hyndman and Y. Fan, "Sample quantiles in statistical packages," *The American Statistician*, vol. 50, pp. 361–365, November 1996.

[10] G. Lucca, J. A. Sanz, G. P. Dimuro, B. Bedregal, H. Bustince, and R. Mesiar, "Cf-integrals: A new family of pre-aggregation functions with application to fuzzy rule-based classification system," *Information Sciences*, vol. 435, pp. 94–110, April 2018.

[11] G. Lucca, J. A. Sanz, G. P. Dimuro, B. Bedregal, H. Bustince, and R. Mesiar, "Pre-aggregation functions: construction and an application," *IEEE Transactions on Fuzzy Systems*, vol. 24, pp. 1063–6706, January 2015.

[12] H. Daniels and B. Kamp, "Application of mlp networks to bond rating and house pricing," *Neural Computing & Applications*, p. 226–234, March 1999.

[13] F. Wilcoxon, "Individual comparisons by ranking methods," *Biometrics*, vol. 1, p. 80–83, December 1945.