Información Tecnológica
Vol. 31(1), 273-286 (2020)
http://dx.doi.org/10.4067/S0718-07642020000100273

Estudio comparativo entre máquinas de soporte vectorial multiclase, redes neuronales y sistema de inferencia neuro-difuso auto organizado para problemas de clasificación

Eiber A. Galindo, Jairo A. Perdomo y Juan C. Figueroa-García
Facultad de Ingeniería, Laboratorio de Automática, Microelectrónica e Inteligencia Computacional LAMIC, Universidad Distrital Francisco José de Caldas, Colombia. (correo-e: eagalindoa@correo.udistrital.edu.co; jcfigueroac@udistrital.edu.co; japerdomot@correo.udistrital.edu.co).

Recibido Jun. 26, 2019; Aceptado Ago. 22, 2019; Versión final Sep. 25, 2019, Publicado Feb. 2020

Resumen

En este trabajo se contextualiza un sistema neuro-difuso autoorganizado (SONFIS), su estructura y funcionamiento son explicados en detalle. Se usa el algoritmo SONFIS en tres problemas de clasificación (Fisher iris, Cáncer de Seno y Actividades Humanas) para posteriormente comparar sus resultados frente a clasificadores universales de buen desempeño en problemas de clasificación como las redes neuronales artificiales (ANN) y máquinas de soporte vectorial multiclase (SVM). Se hace una breve descripción de cada uno de estos métodos. Los resultados del estudio muestran que SONFIS tiene un desempeño similar y en algunos casos mejor que ANN y SVM en problemas de clasificación, con la ventaja que genera una base de reglas que puede usarse para entender el problema estructuralmente.

Palabras clave: lógica difusa; algoritmos inteligentes; máquinas de soporte vectorial; redes neuronales

Comparative study between multiclass support vector machines, neural networks and self-organized neuro-fuzzy inference system for classification problems

Abstract

In this paper an explanation of the structure and how a self-organized neuro-fuzzy inference system (SONFIS) works, is given with detail. The study uses three classification problems (Fisher iris, Breast Cancer and Human Activities) to then compare the results with well-known universal classifiers such as artificial neural networks (ANN) and multiclass support vector machines (SVM). A brief description of each of these methods is presented. The results show that SONFIS has a similar, and sometimes better, performance than ANN and SVM with the advantage of generating a rule basis that helps understanding the inner structure of the problem.

Keywords: fuzzy logic; intelligent algorithms; support vector machines; neural networks.

INTRODUCCIÓN

La clasificación automática de datos puede definirse como la tarea realizada por un autómata de asignar clases a grupos de datos multivariable, de manera autónoma y basado en datos aprendidos a partir de bases de datos previamente ordenadas (Cárdenas et.al, 2014). Sin embargo, la capacidad de generalización de un sistema dependerá tanto de la calidad de los datos usados en su entrenamiento, como de la destreza del analista para ajustar los parámetros del modelo de tal manera que el comportamiento del mismo sea el mejor para cada problema.

El aprendizaje automático busca generar un clasificador con buena capacidad de generalización, es decir, que aprenda a clasificar los ejemplos usados para entrenar el modelo, y que sea capaz de clasificar bien nuevos ejemplos desconocidos (Cárdenas et.al, 2014; De la Hoz et.al, 2019). El desempeño de un clasificador automático dependerá de qué tan similar sea esta clasificación respecto a la humana, lo que se evalúa con algunas métricas como precisión, exactitud, sensibilidad, especificidad, entre otras, que se obtienen a partir de la matriz de confusión (De la Hoz, 2019). Diversas técnicas como K-medias, K-medias difuso, Máquinas de Soporte Vectorial, Bosques Aleatorios, Arboles de decisión, Máxima verosimilitud, Redes Neuronales Artificiales y algoritmos de Aprendizaje Profundo son utilizadas para la clasificación automática de datos (Toosi et.al 2019; Castillo et.al., 2016; De la Hoz et.al., 2019). Aunque la mayoría son técnicas que tienen un buen desempeño sobre problemas complejos y no lineales, son modelos de caja negra o que impiden abstraer conocimiento de los problemas abordados (Brito et.al, 2019, Icarte, 2016).

De esta manera, surge la necesidad de diseñar clasificadores que permitan encontrar relaciones entre las variables que caracterizan un problema y así, abstraer conocimiento del mismo. La propuesta que se presenta es un sistema neuro-difuso autoorganizado que utiliza la capacidad de generalización de las redes neuronales para entrenar un sistema de inferencia difuso (SID), generando únicamente las reglas necesarias para caracterizar el problema y hacer la mejor clasificación a su posible alcance (Wang y Mendel, 1992). Aplicaciones exitosas en control (Icarte, 2016) y pronostico o inferencia de datos (Figueroa et.al (2015); Rodas et.al., 2019) han sido reportadas. Para comparar el desempeño del algoritmo SONFIS utilizamos las ANNs, y las SVMs ya que han mostrado ser buenos clasificadores de datos (Peña y Orellana, 2018; Santana et.al, 2014) y son ampliamente utilizadas en la literatura.

ALGORITMOS UTILIZADOS

Se describen las estructuras de los algoritmos utilizados en este estudio comparativo, haciendo mayor énfasis en la estructura y funcionamiento del algoritmo SONFIS.

Redes Neuronales Artificiales (ANN)

Una red neuronal artificial es un modelo computacional que tiene por objetivo emular el comportamiento del cerebro al procesar conjuntos de datos. Una red neuronal está compuesta por una colección de neuronas artificiales. Una neurona artificial k mapea n señales de entradas x a un único valor de salida y_k. primero calcula la suma ponderada entre las entradas y los costos de transferir información entre una neurona y otra (w_k), y agrega un término umbral, b_k. entonces una operación matemática es desarrollada sobre la suma ponderada, la función de activación Ω (De Smet, 2019; Gutierrez et,al, 2018).

$$y_k = \Omega * \overrightarrow{(x} * \vec{w}_k + b_k) \quad (1)$$

Muchas variaciones de redes neuronales existen en la literatura, pero todas tienen una estructura particular: una capa de entradas, una capa de salida, y varias capas intermedias (capas ocultas), a este tipo de redes neuronales se les llama Perceptrones Multicapa (MLP) (Garcia et.al., 2018; Castillo, 2019). Otro dipo de red neuronal bastante usada en la literatura son las redes neuronales de base radial (RBF), las cuales utilizan funciones de activación en los nodos ocultos radialmente simétricas. Se dice que una función es radialmente simétrica (o es una Función de Base Radial) si su salida depende de la distancia entre un vector que almacena los datos de entrada y un vector de pesos sinápticos, que recibe el nombre de centro o centroide. Las redes RBF tienen una estructura de tres capas de conexión hacia adelante (es decir, que no presentan conexiones laterales ni hacia neuronas de capas anteriores sino solamente con neuronas de la siguiente capa): la capa de entrada, la capa oculta o intermedia y la capa de salida. Las neuronas de la capa de entrada tienen la función de enviar la información a la capa intermedia. Las neuronas de la capa oculta se activan en función de la distancia que separa cada patrón de entrada con respecto al centroide que cada neurona oculta almacena, a la que se le aplica una función radial con forma gaussiana (Hemati et,al, 2018). Las neuronas de la capa de salida son lineales, y simplemente calculan la suma ponderada de las salidas que proporciona la capa oculta. La red neuronal trata de relacionar un conjunto de entrada con unas salidas esperadas, y para

este fin, utiliza métodos de aprendizaje como el Levenberg-Marquardt, retro propagación (incorporado en SONFIS), entre otros, los cuales calculan el error a través de la red y ajusta los costos w_k y el umbral en cada iteración con el fin de acercar más la salida calculada por la red con la salida real. La Fig. 1. Muestra la estructura general que presenta una red neuronal (9).

Para este caso de estudio, se utilizará una red neuronal de 5 capas, y se usará el perceptrón multicapa (MLP) y redes neuronales de base radial (RBF) para ver el desempeño de las redes neuronales desde dos puntos de vista, algoritmo de aprendizaje Levenberg - Mardquart y como criterio de parada: 1000 iteraciones, utilizando el software MATLAB.

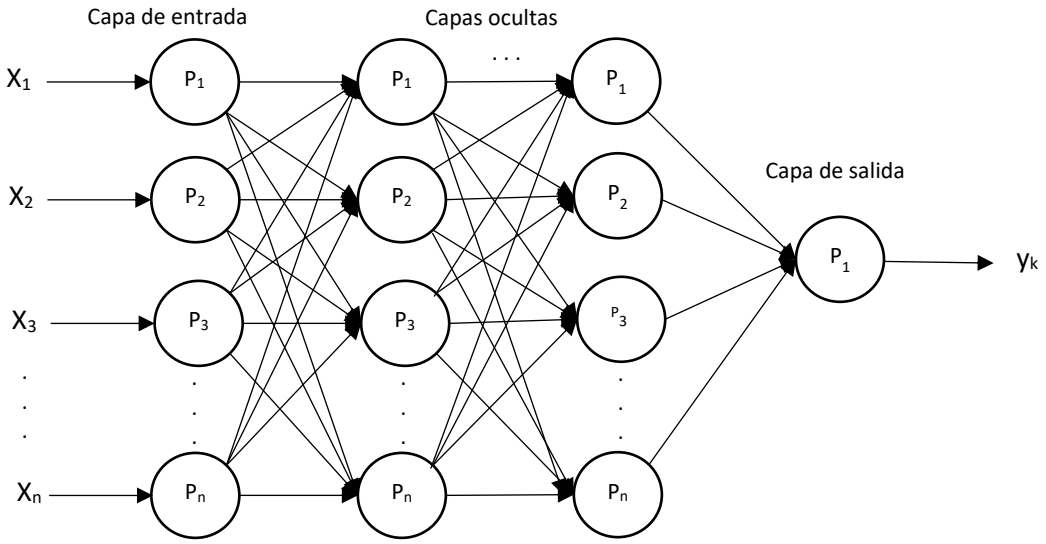

Fig. 1: Estructura general de una red neuronal

Máquinas de Soporte Vectorial (SVM)

Las máquinas de soporte vectorial son un ejemplo de modelos de aprendizaje supervisado. La idea de una máquina de soporte vectorial es generar un hiperplano que permita separar una clase de otra maximizando la distancia entre los puntos de diferentes clases y una función separadora. Un SVM no lineal usa varias funciones kernel para estimar el margen (que es la distancia desde el punto más próximo de una clase a la función separadora). El principal objetivo de estas funciones kernel (i.e. lineal, polinomial, de base radial y sigmoidal) es maximizar el margen entre hiperplanos. Utilizaremos los kernels radial, lineal y polinomial para evaluar el desempeño de SVM sobre cada base de datos. (Ahmad et.al. 2018; Zendehboudi et.al, 2018; García, 2019; Castillo et.al., 2016). Una SVM permite aproximar y_t a través de la función:

$$y_t=b+\sum_{d=i}^{D} w_d * k(x_T,x_d) \quad (2)$$

Donde x_t son los regresores o las variables que representan el problema, y D, el número de ejemplos representativos con los que cuenta la base de datos, b es una constante y w_d son los factores de ponderación de la función de núcleo $k(\cdot,\cdot)$. De esta forma, una SVM es la combinación lineal de la imagen de x_t en un espacio de características no lineal definido por los puntos x_d y la función de transformación no lineal $k(\cdot,\cdot)$. Para efectos de este estudio se utilizarán SVM's con funciones kernel lineal, de base radial (RBF) y polinomial de 4 orden, 1000 iteraciones como criterio de parada utilizando el software MATLAB.

INTRODUCCIÓN A SISTEMAS DIFUSOS

Zadeh (1965) propone los conjuntos difusos para manejar la incertidumbre asociada con el lenguaje humano. Diferentes aplicaciones y experimentos con conjuntos difusos, se han hecho en el campo de la inteligencia computacional para problemas de clasificación (De la Hoz, 2019; Rodas et.al., 2019). Un conjunto difuso *A* en un universo de discurso *X* (conjunto de valores que puede tomar la variable), es caracterizado, por una función de pertenencia $\mu(A): X \rightarrow [0,1]$ que indica el "*grado de pertenencia*" del elemento *x* al conjunto *A*. Por lo tanto, un conjunto difuso es una generalización de un conjunto clásico al permitir que la función de pertenencia tome cualquier valor en el intervalo *[0,1],* en otras palabras, la función de pertenencia de un conjunto clásico solo puede tomar dos valores *{0,1}*, mientras que la función de pertenencia de un conjunto

difuso es una función continua con rango *[0,1].* Un conjunto difuso A en X puede ser representado como un conjunto de pares ordenados de un elemento genérico *x,* y su valor de pertenencia, esto es:

$$A=\{ (x,\mu_A(x)) \mid x \epsilon X\} \quad (3)$$

Un elemento *x* puede ser representado por $j \epsilon R$ conjuntos difusos, cada uno asociado a una función de pertenencia $\mu_A(x)$, $\mu_B(x)$, … $\mu_R(x)$. Entonces, *x* puede pertenecer a diferentes conjuntos, con diferente grado de pertenencia. Un sistema de lógica difusa utiliza un número M de reglas SI-ENTONCES que relacionan un conjunto de entradas a un conjunto de salidas. Cada regla R^i es representada así (Figueroa et.al, 2015):

$$\text{sí } x_1 \text{ es } A_1^i \text{ OP… OP si } x_n \text{ es } A_n^i, \text{ entonces } \hat{y} \text{ es } G^i; i=1,\ldots,M \quad (4)$$

donde G^i representa la salida del sistema difuso para la *i-esima* regla, A_j^i representa el conjunto difuso de la variable x_j para la regla i-*esima*. En este documento, *OP* son los operadores and (∧) – or (∨) utilizando *t*-normas y *s*-normas para representar estos operadores. La salida del sistema se asume como un singleton, de la siguiente manera:

$$\mu_G(x)=\begin{cases} 1 \text{ para } x, \\ 0 \text{ para } x \notin X. \end{cases} \quad (5)$$

El paso final en los sistemas difusos es la defuzzificación, o la conversión de la salida difusa a un número real i.e. $\hat{y}: f^i \rightarrow R$. Se usará el método del centroide como método de defuzzificación, el cual es el promedio de las salidas, y se calcula como se muestra a continuación:

$$\hat{y}=\frac{\sum_{i=1}^{M} f^i w_i}{\sum_{i=1}^{M} w_i}=\sum_{i=1}^{M} G^i * w_i \quad (6)$$

Donde G^i es la normalización de las funciones de activación. El sistema difuso propuesto tiene tantas salidas como reglas, y cada regla conduce a una salida singleton, por lo que el problema central es seleccionar un número de reglas adecuado.

SISTEMA SONFIS

El método SONFIS, Sistema de Inferencia Neuro-Difuso Auto Organizado (Figueroa et.al, 2015) busca generar reglas para un sistema difuso usando redes neuronales. Su estructura general se muestra en la Fig. 2.

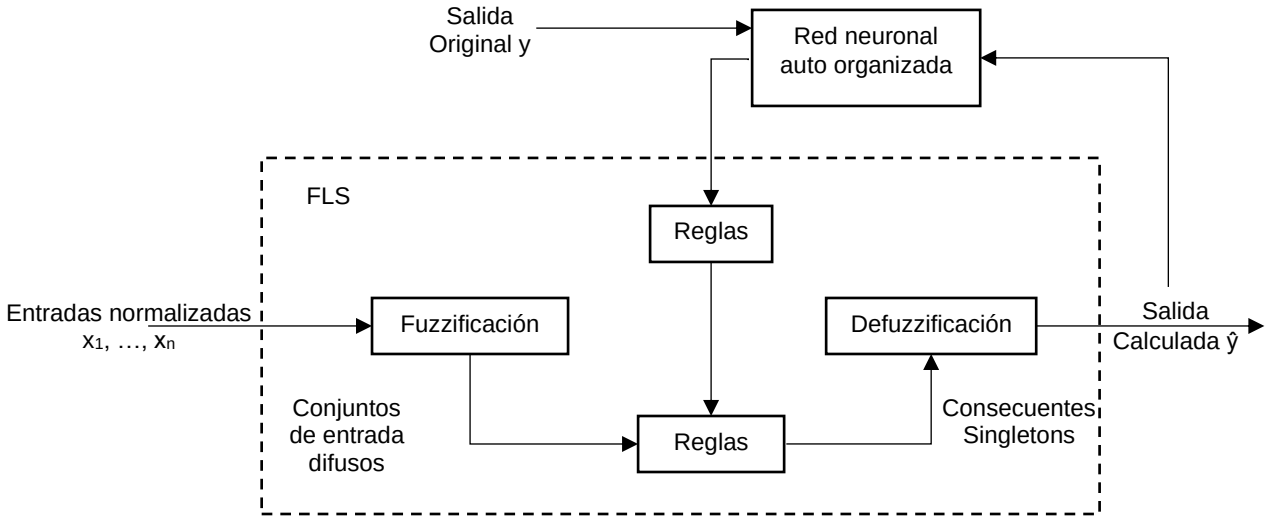

Fig. 2: Diseño de la metodología del sistema neuro-difuso considerado.

El método propuesto para entrenar el sistema difuso es una ANN de 5 capas, en donde los datos de entrada son *n* vectores $\{X_1,\ldots, X_n\}$ de tamaño *k*, y la salida del sistema o respuesta deseada se denota como $\hat{y}$, donde $\{ X_j, \hat{y}\} \epsilon R$. Las capas de la estructura propuesta son: *Normalización, Fuzzificación, Intersección, Agregación, y Defuzzificación.* La estructura del SONFIS se muestra en la Fig. 3.

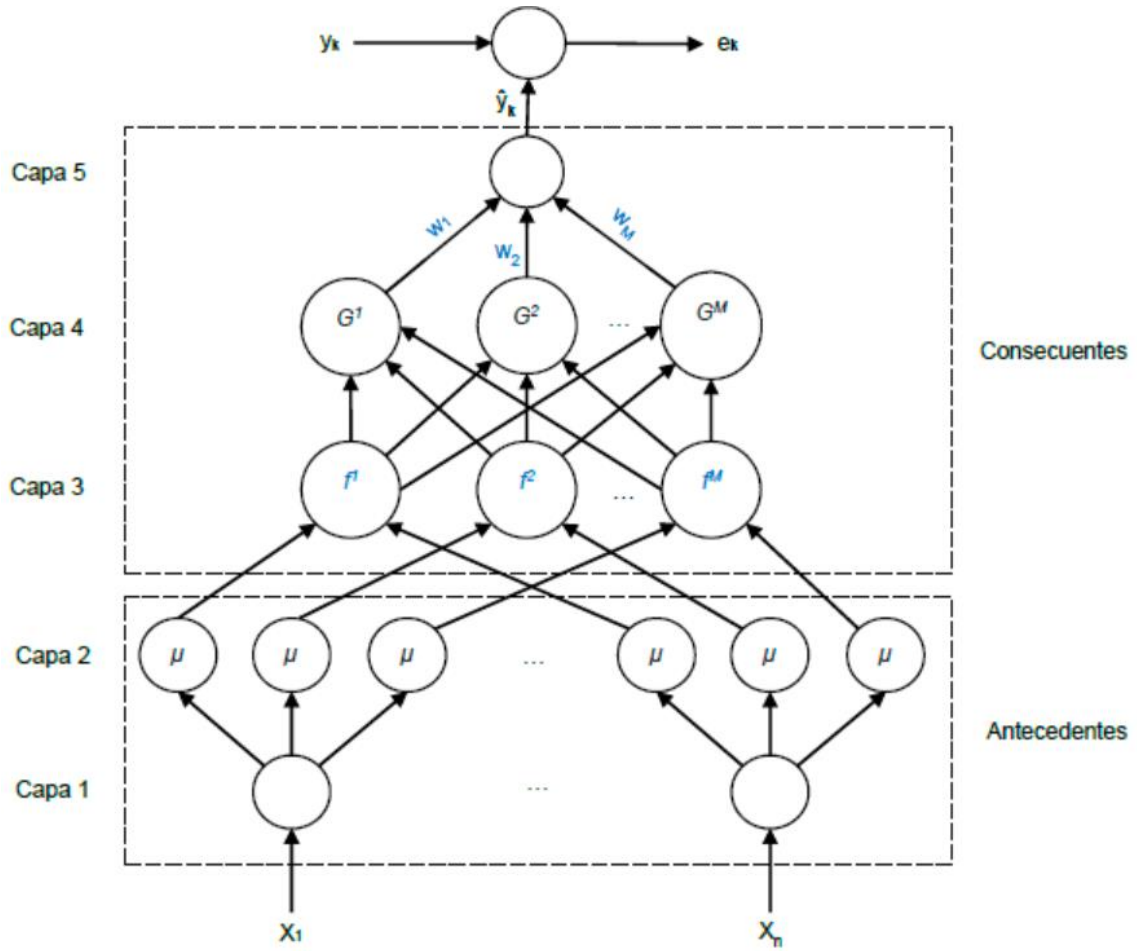

Fig. 3: Red neuronal auto organizada.

Capa 1 – Normalización:

En esta capa, los datos de entrada $\{X_1,\ldots, X_n\}$ deben ser normalizados a dos opciones: ya sea el intervalo $[-1,1]$ o $[0,1]$.

Capa 2 – Fuzzificación:

Esta capa define un conjunto difuso A_j^i para la *j-esima* entrada, llamado μ_{ji}. El conjunto A_j^i debe ser derivable, usualmente se define como una función de pertenencia gaussiana.

Capa 3 – Intersección:

Cada nodo en esta capa es una regla del sistema difuso que usa la *t*-norma del producto para realizar la operación de intersección. La salida de un nodo en esta capa es definida como su *nivel de activación,* f^i, el cual es calculado de la siguiente manera:

$$f^i = \prod_{j=1}^{n} \mu_{ji}\,, \forall \quad i \in M \tag{7}$$

donde *M,* representa el número de Reglas del SID

Capa 4 – Agregación

El proceso realizado en esta capa consiste en normalizar cada f^i usando la suma de todos los niveles de activación provenientes de la capa 3. El número de nodos en esta capa es el mismo que el número de nodos en la capa 3. El proceso de normalización se calcula así:

$$G^i = \frac{f^i}{\sum_i f^i} \tag{8}$$

Capa 5 – Defuzzificación:

Es la última capa de la estructura del sistema neuro-difuso. Esta capa calcula la salida del sistema $\hat{y}$ y realiza el proceso de conversión a salida clásica (defuzzificación). Se calcula usando el promedio entre todos los valores agregados en la capa 4 y los centros de los conjuntos singletons w_i:

$$\hat{y}= \sum_{i=1}^{M} G^i w_i \tag{9}$$

Donde *M* es el número de reglas generadas.

Generación de reglas y parámetros

Figueroa et.al (2015) propuso la siguiente metodología para la generación de reglas: 1) para la primera entrada *x*, generar una nueva regla con centros $m_j = x_j$ y δ_j igual a la desviación estándar de cada X_j, 2) para un nuevo dato de entrada *x*, calcular:

$$f^i(\check{x})= \arg \max_{1\le i\le M(t)} f^i(x) \tag{10}$$

Donde *M(t)* es el número de reglas existentes en el tiempo *t*. a) Si $f^i(x) \le \Phi$, generar una nueva regla:

$$M(t+1)=M(t)+1 \tag{11}$$

Donde Φ es un parámetro predefinido. 3) definir un nuevo conjunto difuso $\mu_{ji=M(t)+1}$ para cada variable de entrada $j=1, \ldots, n$, y un nuevo nodo en la capa 3.

El algoritmo utiliza un umbral Φ que opera como un nivel de activación mínimo que deben tener los datos entrantes x en al menos una de las reglas que componen el sistema de inferencia difuso (SID). Si Φ es bajo, el algoritmo creará pocas reglas centradas en cada una de las variables de entrada $\{X_1, \ldots, X_n\}$ en la primera iteración. Por otro lado, si Φ es alto, entonces el algoritmo creará muchas reglas. Se aclara que dato nuevo *x* para el cual $f^i(x) \ge \Phi$ estará contenido dentro de una regla ya existente.

Parámetros del Algoritmo

Es importante recordar que las funciones de pertenencia deben ser funciones derivables en el intervalo *X*. La función de pertenencia derivable más usada es la gaussiana:

$$\mu_{ji}(x_j)=\exp\left\{-0{,}5\left(\frac{x_j-m_j^i}{\delta_j^i}\right)^2\right\} \tag{12}$$

Donde m_j^i y δ_j^i representan el centro y la desviación de μ_{ji} para la regla i. Las salidas obtenidas del sistema difuso son singletons *w.* Los parámetros para la generación de una nueva regla (*M+1*) proveniente de un nuevo dato x, que no alcanzo el nivel de activación Φ en alguna de las reglas del SID, es creado usando las siguientes ecuaciones recursivas:

$$m_j^{(M+1)}= x_j \tag{13}$$

$$\delta_j^{(M+1)}= -\beta\cdot \ln(f^i) \tag{14}$$

$$w^{(M+1)}= \hat{y}\cdot K \; ; \; k\in[0,1] \tag{15}$$

Donde $\beta>0$ es un parámetro aleatorio que define la amplitud inicial de la regla. El parámetro $k\in[0,1]$ es un número aleatorio que busca ampliar el espacio de búsqueda (Figueroa et.al, 2015).

Algoritmo de aprendizaje

El algoritmo de aprendizaje seleccionado es el algoritmo de retro propagación difuso (Figueroa et.al, 2015) donde ϵ_t es el error de $\hat{y}$ con respecto a la salida real *y*.

$$\epsilon_t = \frac{(y_t - \hat{y}_t)^2}{2} \tag{16}$$

Las ecuaciones de actualización de los parámetros de los conjuntos difusos están basadas en el filtro Kalman y el algoritmo del descenso del gradiente y se presentan de manera escalar, como sigue (Figueroa et.al, 2015):

$$m_j^i(t+1) = m_j^i(t) - \eta \cdot \frac{\hat{y} - y}{\sum_{i=1}^{M} f^i} \cdot (w^i(t) - \hat{y}) \cdot f^i \cdot \frac{2(x_j - m_j^i(t))}{\delta_j^i(t)^2} \tag{17}$$

$$\delta_j^i(t+1) = \delta_j^i(t) - \eta \cdot \frac{\hat{y} - y}{\sum_{i=1}^{M} f^i} \cdot (w^i(t) - \hat{y}) \cdot f^i \cdot \frac{2(x_j - m_j^i(t))^2}{\delta_j^i(t)^3} \tag{18}$$

$$w^i(t+1) = w^i(t) - \eta \cdot \frac{\hat{y} - y}{\sum_{i=1}^{M} f^i}. \tag{19}$$

donde $\eta \in [0,1]$ es la tasa de aprendizaje. El valor asignado a η tiene fuertes implicaciones en el desempeño del algoritmo ya que valores pequeños de η conducen a un costo computacional elevado y puede causar que el algoritmo tenga sobre-entrenamiento. Por otro lado, valores altos de η pueden derivar en una rápida convergencia del algoritmo junto a posibles grandes desviaciones de salida obtenida con respecto a la salida objetivo. En la Figura 4, se muestra el pseudocódigo del algoritmo.

> Inicio
> Calcular parámetros iniciales (m_j^1, δ_j^1, w_1)
> Calcular niveles de activación ($\mu_{ji}(X_j)$, f^1)
> Calcular salida difusa ($\hat{y}$)
> Ajustar parámetros de actualización (t, m_j^t, δ_j^t, w_t)
> para t=1:t
> para (X, Y)
> calcular niveles de activación ($\mu_{ji}(X_j)$, f^t)
> calcular salida difusa ($\hat{y}_t$)
> si ($f^t \leq \Phi$)
> $M = M+1$
> $m_j^M = X$
> $\delta_j^M = \beta \cdot Ln(f^t)$
> $w^M = \hat{y}_t \cdot K$
> para i=1:M
> actualizar parámetros para M-esima regla
> en otro caso
> para i=1:M
> actualizar parámetros para M-esima regla
> fin para
> fin si
> fin para
> fin para
> fin

Fig. 4: Pseudo código de SONFIS

Metodología de clasificación y sintonización de parámetros

La salida del algoritmo será calculada al entero menor, si la parte decimal de $\hat{y}$ es menor que 0,5, y al entero mayor, si la parte decimal de la salida $\hat{y}$ es mayor o igual a 0,5. Es decir, para un ejemplo que entra a la red y que obtiene una salida de 2,4, por ejemplo, la clase asignada al ejemplo entrante es la clase 2. Los parámetros iniciales del SONFIS son: η= 0,05, para que la búsqueda de optimalidad sea más controlada, β= 0,5 que asegura que la amplitud de una nueva regla este focalizada en el dato sobre el cual se generó y Φ=0,5 ya que es un punto central en el que se es ni pesimista ni optimista. Mientras η se mantiene constante los valores de β y Φ se van ajustando a través de una búsqueda exhaustiva que mantenga un balance entre el

desempeño sobre el conjunto de validación y el número de reglas creado por el SID, la finalidad es que se genere un número de reglas del cual se pueda inferir comportamientos y relaciones entre las variables.

Evaluación, desempeño y comparación del algoritmo.

Las métricas a usar para comparar el desempeño de los algoritmos serán la precisión, la exactitud, la sensibilidad y la especificidad, calculando la matriz de confusión de cada algoritmo. Se realizaron 50 experimentos de Montecarlo por cada algoritmo con conjuntos de entrenamiento y validación aleatorios (la metodología para seleccionar las bases de datos es presentada en la sección bases de datos), promediando el desempeño en cada experimento.

$$\text{Precisión=PR=}\frac{VP}{VP+FP}*100\% \tag{20}$$

$$\text{Exactitud=EX=}\frac{VP}{VP+VN+FP+FN}*100\% \tag{21}$$

$$\text{Sensibilidad= SB=}\frac{VP}{VP+FN}*100\% \tag{22}$$

$$\text{Especificidad=ES=}\frac{VN}{VN+FP}*100\% \tag{23}$$

Donde VP = verdaderos positivos = Casos positivos correctamente clasificados; VN = verdaderos negativos = Casos negativos correctamente clasificados, FP = falsos positivos = Casos negativos incorrectamente clasificados, FN = falsos negativos = Casos positivos incorrectamente clasificados. y Sensibilidad (Fracción positiva verdadera) es la probabilidad de que una prueba de diagnóstico sea positiva, dado que la persona tiene la afección. La especificidad (fracción negativa verdadera) es la probabilidad de que una prueba de diagnóstico sea negativa, dado que la persona no tiene la enfermedad. La precisión es la probabilidad de que una prueba de diagnóstico se realice correctamente (Mosquera et.al., 2018, Tobón y Cortés, 2018).

RESULTADOS Y DISCUSIÓN

Se usaron tres bases de datos de problemas de clasificación de amplio uso en la literatura, para de esta manera establecer un punto de comparación entre los diferentes algoritmos y sus desempeños basados en las métricas descritas anteriormente. La descripción de las bases de datos, las metodologías usadas y los resultados obtenidos son mostrados a continuación:

Bases de datos

Las bases de datos son seleccionadas con el fin de evaluar los modelos en cada una de las métricas de manera objetiva. El primer conjunto de datos es el conjunto de datos de Fisher, o *Fisher iris,* está compuesto por la descripción de cuatro características morfológicas de tres diferentes especies de flor iris, disponible en UCI repository, las variables morfológicas que clasifican a cada especie de flor son: 1) Largo del sépalo, 2) Ancho del sépalo, 3) Largo del pétalo y 4) Ancho del pétalo, y las posibles clases en las que se pueden agrupar los datos son 1) Iris setosa, 2) Iris versicolor y 3) Iris virginica. Numero de ejemplos: 150. Esta base de datos se seleccionó con el fin de evaluar el comportamiento de los modelos sobre la precisión y sensibilidad pues el objetivo es que logre diferenciar una clase de otra y así logre la mayor cantidad de aciertos tanto en entrenamiento como en validación

El segundo conjunto usado es la base de datos de actividades físicas humanas basadas en la captura de datos de sensores de smartphones. Las clases de actividades en las cuales se clasifican los datos son: 1) Sentado, 2) Parado, 3) Caminando, 4) Corriendo y 5) Bailando. Cada observación tiene 60 variables extraídas de datos de aceleración tomados por sensores de teléfonos inteligentes. Este conjunto está disponible en las bases de datos de ejemplo de MATLAB. Numero de ejemplos: 24.075. Esta base de datos se selecciona con el fin de evaluar los modelos sobre la métrica de exactitud, pues la clasificación de las posiciones del cuerpo es una tarea de gran interés y un reto para los dispositivos tecnológicos como smartphones, dada la característica cambiante y la naturaleza temporal de estas actividades. La tercera base de datos describe el diagnostico de tumores de mama tomados con aguja fina (FNA) para 699 pacientes, y clasificados como 1) Maligno o 2) Benigno. Se usan 9 variables de entrada, con esta base de datos se pretende evaluar el desempeño de los modelos sobre la especificidad y precisión, por la implicación que tiene un falso negativo en estos tipos de problemas.

Para construir los conjuntos de entrenamiento y validación de las bases de datos, en cada experimento se realiza una simulación de Montecarlo donde si un número aleatorio es menor o igual a 0,8, el ejemplo será usado para entrenamiento, de lo contrario, será usado en la validación de los modelos, de esta manera evitamos caer en la "falacia de la evidencia incompleta" y aseguramos total aleatoriedad en las bases de datos; también se realiza normalización sobre las bases de datos para evitar problemas con las escalas de las variables de entrada, por lo que se normalizan las nuevas bases de datos en el intervalo [0,1].

Selección de características

Para la base de datos "iris", dado que las características de una flor son discriminantes entre clases, no es necesario reducir la dimensión o el número de variables de entrada pues se perdería información importante sobre las características que distinguen una clase de otra, por lo que se usan todas las variables para entrenar los modelos. Para los demás problemas, dado que no conocemos el impacto de las variables en el problema, se utiliza análisis de componentes principales para reducir la dimensión o el número de características entregadas a los modelos con el fin intrínseco de evitar el sobre entrenamiento de las técnicas debido al exceso de características (Tobón y Cortés, 2018), y como premisa, se considerarán el número de variables de las nuevas matrices para las cuales se asegure una varianza retenida de al menos el 95%. De esta manera, para la base de datos "actividades humanas" se utiliza una matriz reducida de 2 variables de entrada con las cuales se asegura una varianza retenida del 98.460%, y para el problema "cáncer de seno" se utiliza una matriz reducida de 7 variables de entrada con las que se asegura un 96.9094% de varianza retenida.

Evaluación de los métodos

Para evaluar los métodos, se analizan las matrices de confusión para cada problema, en cada método, y en cada experimento. Una matriz de confusión mide el número de clasificaciones que se realizan correctamente y también las clasificaciones que pertenecen a una clase y que se asumen como otra, y de estas pueden extraerse indicadores como la precisión, la exactitud, la sensibilidad y la especificidad, indicadores claros del desempeño de un clasificador automático. Se realizan 50 experimentos con cada algoritmo y se promedian dichos indicadores para obtener un indicador total del desempeño de cada algoritmo sobre cada problema. La tabla 1 muestra los indicadores mencionados anteriormente con de cada método, sobre cada conjunto de datos:

Tabla 1. Comparación entre algoritmos de clasificación.

	Metrica	Iris (entrenamiento)	Iris (validación)	Cáncer (entrenamiento)	Cáncer (validación)	Actividades humanas (entrenamiento)	Actividades humanas (validación)
RNA MLP	PR	97.898 ± 0.031	95.668 ± 0.084	96.8 ± 0.019	95.4 ± 0.03	97.4 ± 0.005	94.7 ± 0.026
	EX	97.797 ± 0.016	95.338 ± 0.044	97.3 ± 0.007	95.8 ± 0.016	96.2 ± 0.014	95.7 ± 0.032
	SB	97.796 ± 0.035	95.501 ± 0.087	97.2 ± 0.012	95.5 ± 0.033	97.2 ± 0.286	90.1 ± 0.002
	ES	98.898 ± 0.017	97.691 ± 0.047	97.2 ± 0.012	95.5 ± 0.033	97.204 ± 0.03	94.3 ± 0.629
RNA RBF	PR	100 ± 0	87.072 ± 0.111	100 ± 0	86.3 ± 0.132	100 ± 0	84.4 ± 0.008
	EX	100 ± 0	86.57 ± 0.052	100 ± 0	80.5 ± 0.119	100 ± 0	91.5 ± 0.027
	SB	100 ± 0	86.306 ± 0.153	100 ± 0	72.1 ± 0.379	100 ± 0	93.6 ± 0.324
	ES	100 ± 0	93.139 ± 0.066	100 ± 0	72.1 ± 0.379	100 ± 0	97.1 ± 0.049
SVM lineal	PR	95.236 ± 0.047	94.594 ± 0.097	96 ± 0.02	95.6 ± 0.029	87.5 ± 0.101	84.8 ± 0.14
	EX	94.952 ± 0.008	94.393 ± 0.044	96.6 ± 0.004	96.2 ± 0.015	88.3 ± 0.001	85.3 ± 0.055
	SB	94.882 ± 0.055	94.449 ± 0.094	96.5 ± 0.007	95.9 ± 0.027	86 ± 0.1	83.6 ± 0.144
	ES	97.458 ± 0.027	97.28 ± 0.048	96.5 ± 0.007	95.9 ± 0.027	97.1 ± 0.025	96.3 ± 0.039

Tabla 1 (continuación)

	Metrica	Iris (entrenamiento)	Iris (validación)	Cáncer (entrenamiento)	Cáncer (validación)	Actividades humanas (entrenamiento)	Actividades humanas (validación)
SVM polinomial (orden 4)	PR	100 ± 0	94.996 ± 0.088	97.5 ± 0.085	91.6 ± 0.073	92 ± 0.104	88.5 ± 0.174
	EX	100 ± 0	95.009 ± 0.039	95.2 ± 0.111	89.6 ± 0.083	94 ± 0.018	90.8 ± 0.079
	SB	100 ± 0	94.899 ± 0.081	93.1 ± 0.236	86.1 ± 0.215	90.9 ± 0.139	88.4 ± 0.189
	ES	100 ± 0	97.551 ± 0.04	93.1 ± 0.236	86.1 ± 0.215	98.6 ± 0.02	97.7 ± 0.048
SVM RBF	PR	98.496 ± 0.023	97.388 ± 0.061	100 ± 0	92.4 ± 0.081	94 ± 0.068	93.1 ± 0.072
	EX	98.403 ± 0.007	97.399 ± 0.026	100 ± 0	93.8 ± 0.023	95.2 ± 0.001	93.8 ± 0.028
	SB	98.395 ± 0.025	97.436 ± 0.058	100 ± 0	95.1 ± 0.046	92.3 ± 0.121	91.1 ± 0.126
	ES	99.196 ± 0.013	98.724 ± 0.029	100 ± 0	95.1 ± 0.046	98.9 ± 0.014	98.5 ± 0.021
SONFIS	PR	95.13±0.019	94.10 ±0.060	93.80 ±0.051	93.16 ±0.047	94.47 ±0.035	93.63 ±0.054
	EX	95.80 ±0.061	95.10 ±0.099	96.17 ±0.054	96.20 ±0.053	95.99 ±0.057	95.62 ±0.076
	SB	95.07 ±0.075	93.65 ±0.155	94.77 ±0.024	93.75 ±0.035	94.92 ±0.049	93.70 ±0.093
	ES	97.60 ±0.037	97.10 ±0.069	93.77 ±0.065	93.73 ±0.070	96.19 ±0.051	95.41 ±0.069

Podemos inferir que la red neuronal RBF presenta un claro sobre entrenamiento dada su alta tasa de acierto sobre la base de datos de entrenamiento y la diferencia entre los indicadores de validación, para las demás metodologías, el comportamiento es consistente pues presentan indicadores similares en ambas etapas. SONFIS, a diferencia de los demás métodos, genera una base de reglas que permiten la generación de conocimiento sobre el problema abordado. La Tabla 2. Muestra el número de reglas en promedio generadas en cada problema analizado:

Tabla 2. Numero de reglas generadas por SONFIS

	Numero de reglas
iris (entrenamiento)	4.26
iris (validación)	0.751
cáncer (entrenamiento)	16.98
cáncer (validación)	13.46
actividades humanas (entrenamiento)	11.28
actividades humanas (validación)	1.565

Además, se tiene en cuenta el desempeño de cada algoritmo en términos de costo computacional. La tabla 3 muestra el costo computacional de cada uno de los modelos sobre cada problema abordado (en segundos): Como puede apreciarse, SONFIS tiene un desempeño similar en términos de tiempo de computo, pues, por ejemplo, en la base de datos “Fisher iris”, es mucho más rápido en generar la clasificación que un Perceptrón Multicapa, y más rápido que un SVM con kernel Polinomial, pero es más costoso en términos de tiempo que, por ejemplo, un SVM lineal o RBF, para la misma base de datos.

BASE DE CONOCIMIENTO OBTENIDA MEDIANTE SONFIS

Gracias a la construcción del sistema de inferencia difuso por medio de la generación de reglas SI-ENTONCES, se puede establecer relaciones entre las variables que describen el problema. Para mostrar esta característica, se utiliza uno de los experimentos sobre la base de datos Fisher iris, donde se generaron 3 reglas. La Figura 5, 6, 7, 8 y 9 muestra los conjuntos difusos generados sobre cada una de las variables del problema (Longitud del Sépalo, Ancho del Sépalo, Longitud del Pétalo y Ancho del Pétalo) y los conjuntos Singleton respectivamente (con valores w_1 =0,05, w_2 =0,755 y w_3 =3,141), que conforman cada una de las reglas.

Tabla 3. Costo computacional de los algoritmos (en segundos)

	Fisher iris	Cáncer	Actividades humanas
RNA MLP	138.289	31.901	5563.530
RNA RBF	3.686	7.976	964.743
SVM lineal	2.996	2.231	1141.043
SVM polinomial (orden 4)	44.361	162.489	45608.082
SVM RBF	2.901	2.737	813.310
SONFIS	22.910	279.600	4358.400

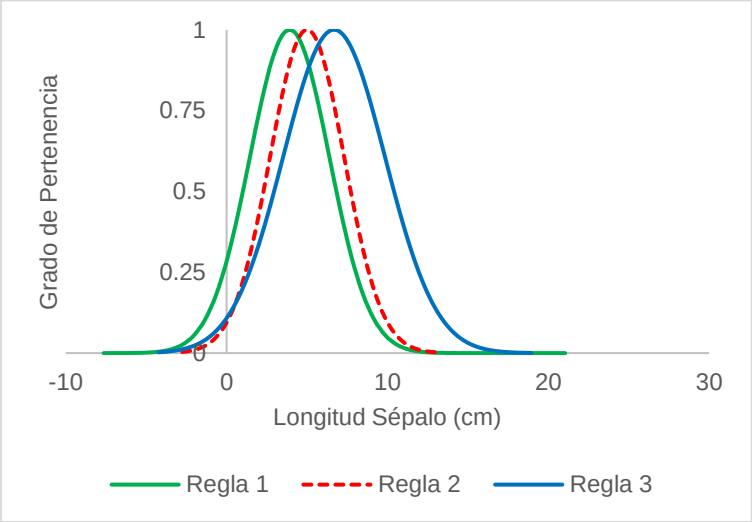

Fig. 5: Conjuntos Difusos generados sobre la variable Longitud de Sépalo

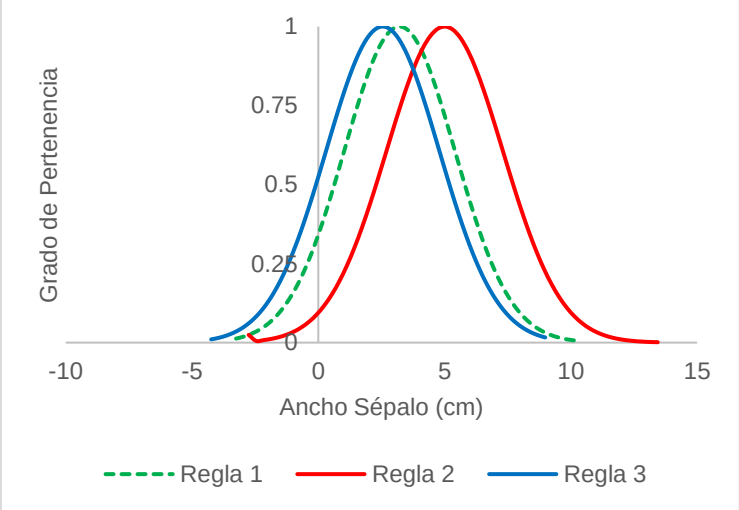

Fig. 6: Conjuntos Difusos generados sobre la variable Ancho de Sépalo

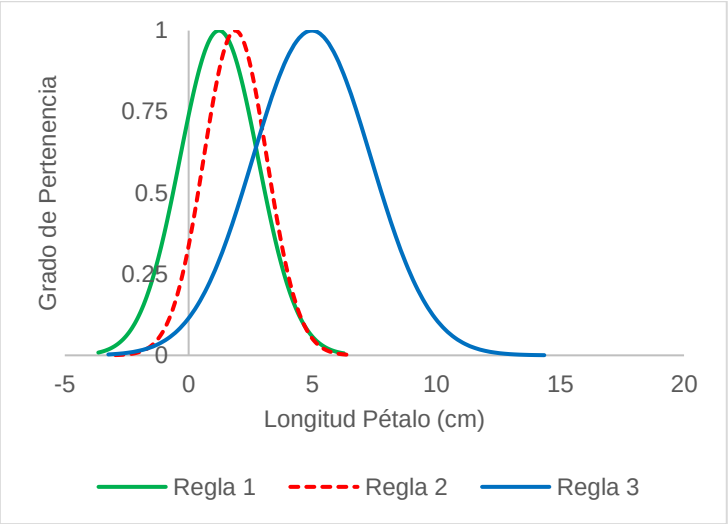

Fig. 7: Conjuntos Difusos generados sobre la variable Longitud de Pétalo

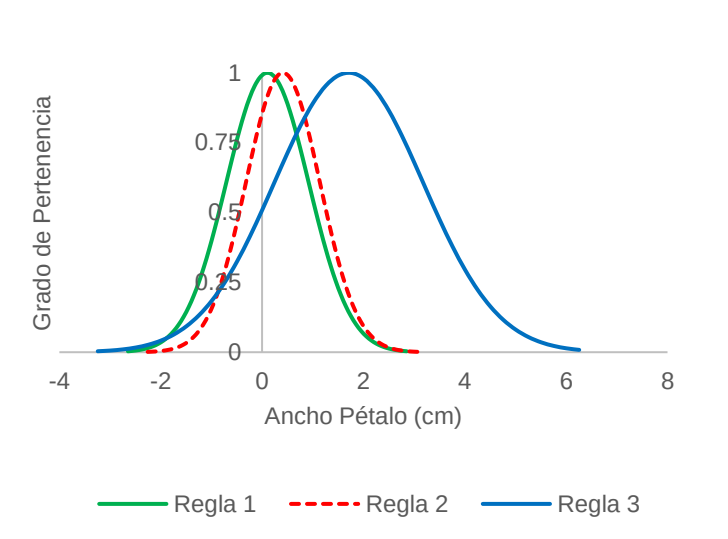

Fig. 8: Conjuntos Difusos generados sobre la variable Ancho de Pétalo

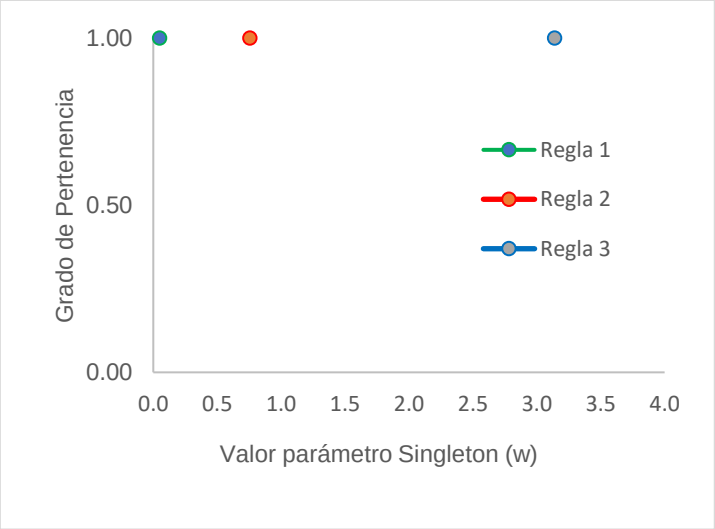

Fig. 9: Conjuntos Difusos Singleton

Se puede entender cada conjunto difuso como una caracterización de cierta región del universo de discurso de las variables, que, de manera autónoma, encuentran diferentes relaciones entre ellas y se materializan en las reglas que caracterizan cierto tipo de Flor iris. La principal ventaja de este enfoque es que a cada conjunto se le puede asociar una etiqueta lingüística, que permitirá establecer las reglas en términos lingüísticos, como se ejemplifica a continuación, utilizando la Regla 1 generada por SONFIS. SI la Longitud del Sépalo es pequeña y el Ancho del Sépalo es mediano y tanto la Longitud y el Ancho del Pétalo es pequeño, ENTONCES, el conjunto Singleton es w=0,05, y aplicando el método de defuzzificación el SID concluye que son características que tienen las Plantas tipo 1, que son las Flores iris setosa.

CONCLUSIONES

SONFIS surge como una metodología alternativa para abordar problemas de clasificación, ya que tiene un desempeño similar e incluso en algunos casos mejor que algunas de las metodologías más efectivas y más usadas para este tipo de problemas con costo operacional similar e incluso mejor en algunos casos y con la ventaja adicional de permitir la abstracción de conocimiento acerca del problema, por medio de las reglas generadas para extraer las relaciones entre las variables que caracterizan el comportamiento del problema.

En cuanto a costo computacional, SONFIS utiliza los mismos recursos (en términos de tiempo de ejecución) que los algoritmos propuestos para su comparación y en algunos casos realiza la clasificación de manera más rápida, y generando resultados similares en términos de los indicadores considerados. Con respecto a la base de datos obtenida, es de gran ayuda poder inferir comportamientos basados en reglas, pues estas pueden aportar en la visualización de un problema de forma más efectiva que simplemente realizando la división de los datos, como se demuestra en el ejemplo presentado.

La evidencia mostrada en los casos de estudio presentados es satisfactoria y muestra el potencial que SONFIS tiene ya que inicialmente fue utilizado en problemas de series de tiempo y control (Figueroa et.al, 2015), el desempeño obtenido en problemas de clasificación es adecuado. Metodologías evolutivas como los algoritmos genéticos, o los métodos de inteligencia de enjambres pueden ser combinados con SONFIS para seleccionar los mejores parámetros que se ajusten a cada problema.

REFERENCIAS

Ahmad, I., Basheri, M., Iqbal, M.J.: Performance Comparison of Support Vector Machine, Random Forest, and Extreme Learning Machine for Intrusion Detection, IEEE Acces,, 6(1), 33789 – 33795 (2018).

Ahmadloua, M., Karimia, M., Alizadehb, S., Shirzadic, A., Parvinnejhadd, D., Shahabie H.: Flood susceptibility assessment using integration of adaptive network-based fuzzy inference system (ANFIS) and biogeography-based optimization (BBO) and BAT algorithms (BA), Doi: https://doi.org/10.1080/10106049.2018.1474276, Geocarto International, 34(11), 1252-1272 (2018).

Brito, F.J., Arantes Filho, L. R., Frutuoso Guimaraes, L.N., Intelligent Classification of Supernovae Using Artificial Neural Networks, Inteligencia Artificial, ISSN: 1137-3601, 22(63), 39 – 60, (2019)

Cárdenas, J.P., Olivares, O. y Alfaro, R., Clasificación automática de textos usando redes de palabras, Revista Signos,Chile, ISSN: 0718-0934, 47(86), 346 – 364, (2014)

Castillo, D. D., Perez, M.R., Pérez, L., Orozco, R., Ginori, J.: Algoritmos de aprendizaje automático para la clasificación de neuronas piramidales afectadas por el envejecimiento, ISSN 1684-1859, Revista Cubana de Informática Médica, 8(3), 559-571 (2016)

Clavero, A., M. Salicrú y D. Turbón, Sex prediction from the femur and hip bone using a sample of CT images from a Spanish population, doi: 10.1007/s00414-014-1069-y, International Journal of Legal Medicine, 129(2), 373-383 (2015)

De la Hoz, E.J., De la Hoz, E.J., Fontalvo, T.J., Metodología de Aprendizaje Automático para la Clasificación y Predicción de Usuarios en Ambientes Virtuales de Educación, DOI: 10.4067/S0718-07642019000100247, Información Tecnológica, 30(1), 247-254 (2019).

De Smet, S., Scheeres, D.J.: Identifying heteroclinic connections using artificial neural networks, DOI: https://doi.org/10.1016/j.actaastro.2019.05.012, Acta Astronautica, 161(1) 192-199 (2019).

Figueroa, J.C., Ochoa Rey, C.M., Avellaneda Gonzalez, J.A., Rule generation of fuzzy logic systems using a self-organized fuzzy neural network, Doi: 10.1016/j.neucom.2014.09.079, Neurocomputing, 151(3), 955 – 962, (2015)

Galili, T., Dendextend: an R package for visualizing, adjusting and comparing trees of hierarchical clustering, doi.org/10.1093/bioinformatics/btv428, Bioinformatics, 31(22), 3718-3720 (2015)

Gutierrez, E.R., Almeida, J.C., Arzola, J.: Modelado por redes neuronales artificiales de los indicadores de desempeño de operación en instalaciones de gasificación termoquímica downdraft, ISSN: 2616-9541, Aporte Santiaguino, 11(2), 211 – 224 (2018).

García, A.G., Rios, A.B., Teelo, E., Barrón, J.H., Diaz, A.: aplicación de una red neuronal artificial para la clasificación automática de tuits en español, ISSN: 2448-847X, Pistas Educativas, 40(130), 508-524 (2019).

García, M.E.: Máquinas de soporte vectorial y árboles de clasificación para la detección de operaciones sospechosas de lavado de activos, DOI: https://doi.org/10.21501/21454086.2904, Lampzakos, 1(21), 26-38 (2019).

Hemati, S., Beiranvand, P., Sharafi, M.: ellipse perimeter estimation using nonparametric regression of rbf neural network based on elliptic integral of the second type, ISSN: 2224- 5405, Revista Investigacion Operacional, 39(4), 639 – 646, (2018).

Icarte, G.A., Aplicaciones de inteligencia artificial en procesos de cadenas de suministros: una revisión sistemática, Doi: 10.4067/S0718-33052016000400011, Ingeniare, 24(4), 663-679, (2016)

Macias Bernal, J.M., Calama Rodríguez, J.M. y Chávez de Diego, M.J., Modelo de predicción de la vida útil de la edificación patrimonial a partir de la lógica difusa, Informes de la Construcción, ISSN-L: 0020-0883, 66(533), e006, (2014)

Mosquera, R., Castrillon, O.D., Parra, L.: Máquinas de Soporte Vectorial, Clasificador Naïve Bayes y Algoritmos Genéticos para la Predicción de Riesgos Psicosociales en Docentes de Colegios Públicos Colombianos, DOI: http://dx.doi.org/10.4067/S0718-07642018000600015, Información Tecnológica Vol. 29(6), 153-162 (2018)

Peña, M., Orellana, J.: Red neuronal para clasificación de riesgo en cooperativas de ahorro y crédito, DOI: http://dx.doi.org/10.24133/cctespe.v13i1.710, revista del Congreso de Ciencia y Tecnología, 13(1), (2018).

Rodas, P., Guamán, R., Colina, E., Peña, M., Singueza, L.: Modelo matemático basado en programación lineal y lógica difusa para predicción de tiempos en industrias de ensamble de bicicletas, DOI: 10.17013/risti.n.pi-pf, Revista Ibérica de Sistemas e Tecnologias de Informacion, 1(19), 581-594 (2019).

Santana, P., Costaguta, R., Missio, D., Aplicación de Algoritmos de Clasificación de Minería de Textos para el Reconocimiento de Habilidades de E-tutores Colaborativos, ISSN: 1137-3601, Inteligencia Artificial, 17(53), 57-67, (2014)

Toosi, N.B., Reza, A., Fakheran, S., Pourmanafi, S., Ginzler, C., Waser, L.T.: Comparing different classification algorithms for monitoring Mangrove cover changes in southern Iran, DOI: 10.1016/j.gecco.2019.e00662, Global Ecology and Conservation, 19(e00662), (2019).

Tobón, I.J., Cortéz J.A.: Identificación de instrumentos musicales de cuerdas pulsadas de la región andina colombiana en solo, mediante técnicas de aprendizaje de máquina, DOI: https://doi.org/10.24050/reia.v15i30.1245, Revista EIA, 15(30), 177-193 (2018).

Zendehboudi, A., Baseer, M.A., Saidur, R.: Application of support vector machine models for forecasting solar and wind energy resources: A review, DOI: 10.1016/j.jclepro.2018.07.164, Journal of Cleaner Production, 199(1), 272 – 285 (2018).