

Optimización de hiperparámetros en algoritmos de aprendizaje automático

Optimization of hyperparameters in machine learning algorithms

July A. Acero-Lozada¹, Kevin S. Rojas-Ramírez²

¹Opalo Research Group, Facultad de ingenierías físicomecánicas, EEIE, Universidad Industrial de Santander, Colombia. Email: july2165603@correo.uis.edu.co

²Opalo Research Group, Facultad de ingenierías físicomecánicas, EEIE, Universidad Industrial de Santander, Colombia. Email: kevin2180230@correo.uis.edu.co

Recibido: 3 junio, 2023. Aceptado: 3 junio, 2023. Versión final: 3 junio, 2023.

Resumen

En la actualidad, *el Automated Machine Learning* (AutoML) ha sido ampliamente aplicado debido al alto potencial benéfico que aporta a los distintos sectores de la industria, particularmente en la mejora de, los flujos de trabajo, el rendimiento de los procesos y la efectividad empresarial. Dentro del AutoML es importante la adecuada elección de los valores de hiperparámetros mediante técnicas de optimización, ya que los algoritmos de ML dependen de los valores de hiperparámetros (HPs) elegidos debido a que estos influyen en el rendimiento de la máquina. Por ello, en esta investigación se diseña metodológicamente el proceso de construcción de los espacios de búsqueda y se propone una técnica para la optimización hiperparámetros de una máquina de Random Forest mediante la adaptación de la metaheurística de enjambre de partículas (PSO). Esta técnica se utiliza para el análisis de conjunto de datos equilibrados y desequilibrados, mediante un proceso de *benchmarking*. Para validar el rendimiento del método, inicialmente se ejecutó el algoritmo de Random Forest sin aplicar la técnica de optimización, en donde se encontró la existencia de sobreajuste. Para reducir este comportamiento se establece una comparación entre PSO y técnicas de mayor usabilidad como *Grid Search* y *Random Search* a fin de analizar el comportamiento del modelo e identificar la técnica más efectiva en términos de funcionalidad y rendimiento. Los resultados demuestran que el modelo de clasificación *Random Forest* junto con la técnica de optimización de hiperparámetros *Particle Swarm Optimization* (PSO) mejoró la eficacia general del modelo, posibilitando obtener valores óptimos de hiperparámetros que mejoraron el rendimiento y el sobreajuste del modelo.

Palabras clave: HPO, AutoML, espacios de búsqueda, hiperparámetros, estrategia de espacios de búsqueda, Ingeniería de características.

Abstract

Nowadays, Automated Machine Learning (AutoML) has been widely applied due to the high beneficial potential it brings to different industry sectors, particularly in the improvement of workflows, process performance and business effectiveness. Within AutoML, the proper choice of hyperparameter values through optimization techniques is important, since ML algorithms depend on the chosen hyperparameter values (HPs) due to the fact that they influence machine performance. Therefore, in this research we methodologically design the process of constructing the search spaces and propose a technique for hyperparameter optimization of a Random Forest machine by adapting the particle swarm metaheuristic (PSO). This technique is used for the analysis of balanced and unbalanced dataset through a benchmarking process. To validate the performance of the method, the Random Forest algorithm was initially run without applying the optimization technique, where the existence of overfitting was found. To reduce this behavior, a comparison between PSO and more usable techniques such as Grid Search and Random Search is established in order to analyze the behavior of the model and identify the most effective technique in terms of functionality and performance. The results show that the Random Forest classification model together with the hyperparameter optimization technique Particle Swarm Optimization (PSO) improved the overall efficiency of the model, making it possible to obtain optimal hyperparameter values that improved the performance and overfitting of the model.

keywords: HPO, AutoML, search spaces, hyperparameters, search space strategy, feature engineering.

1. Introducción

Los distintos sectores económicos a nivel mundial se han visto enfrentados a nuevos retos para analizar, asegurar y cuidar sus datos [1]. Sectores como el educativo, financiero, energético, salud, industrial, tecnológico, manufacturero y de servicios, generan gran cantidad de datos en múltiples formatos. Estos conjuntos de datos conformados por mayor tamaño y complejidad son conocidos como *Big Data*, caracterizados por poseer un alto volumen, velocidad y variedad en su estructura que, al ser utilizados en la toma de decisiones, requieren velocidad y validez. Sin embargo, los *hardware* y *software* convencionales no tienen la capacidad para gestionarlos o procesarlos por ser tan masivos y complejos de examinar [2]. Por lo tanto, con el fin de abordar tales desafíos, han emergido nuevas técnicas de análisis más robustas con la capacidad de soportar las propiedades de los volúmenes de datos [2].

Técnicas innovadoras como el procesamiento del lenguaje natural, el aprendizaje automático y el aprendizaje profundo, han surgido en el campo de la inteligencia artificial (IA) [3]. En los últimos años, el *Machine Learning* (ML) ha sido uno de los métodos con mayor trascendencia en investigación y parte integral del trabajo eficiente. Esto debido a que soluciona los problemas asociados al *Big Data*, beneficiándose de las diversas características de los datos (alta heterogeneidad, la falta estructura, el carácter incompleto, los errores manuales, etc.) a raíz de un eficiente uso de recursos computacionales y humanos. El ML trae consigo ventajas competitivas en las organizaciones mejorando la toma de decisiones en el ámbito estratégico, de una manera efectiva, informada y fiable, al basarse en un análisis holístico de datos propios de la organización. Este proceso se realiza mediante la adquisición, gestión, procesamiento, extracción, interpretación de dichos datos. El uso de modelos y técnicas de ML es más que necesario para cualquier organización actualmente con el propósito de facilitar el proceso de toma de decisiones [3].

El ML es uno de los métodos de análisis más relevantes en la actualidad, debido a que son genéricos y demuestran alto rendimiento en problemas de análisis de datos con grandes volúmenes. Sin embargo, el proceso de construcción dificulta en algunos casos su aplicación, debido a que requiere grandes cantidades de tiempo en la definición de la arquitectura modelo y elección del tipo de algoritmo [4]. Por esta razón, se implementa el *Automated Machine Learning* (AutoML) como método para automatizar las diversas decisiones asociadas a su construcción, aplicando técnicas de Optimización Hiperparamétrica (HPO) a modelos de ML. De esta manera, se permite la configuración óptima de la

arquitectura, reduciendo el esfuerzo e interacción humana necesario y aumentando el rendimiento [5].

Por tanto, el uso del AutoML mejora el rendimiento en los modelos de ML para lograr, los objetivos planteados utilizando menos recursos beneficiando en gran medida a las organizaciones. Por ello, el presente trabajo se enfoca en la aplicación del AutoML mediante técnicas de Optimización de Hiperparámetros (HPO) aplicadas a un ML. A partir de la identificación, selección y validación se definirán las técnicas de HPO más relevantes para solucionar las problemáticas de un algoritmo de ML.

En consecuencia, dada la importancia primordial del proceso de AutoML en la ejecución de modelos de *Machine Learning*, a lo largo del presente proyecto se implementa la técnica de optimización de enjambre de partículas (PSO) en un algoritmo de *Random Forest* con el fin de ajustar los hiperparámetros del modelo y a través de un proceso de *benchmarking* evaluar el desempeño del modelo en distintos escenarios, en este caso conjunto de datos equilibrados y desequilibrados. Esta investigación busca verificar que el rendimiento y efectividad de una máquina de Machine Learning depende en gran medida de los valores óptimos de hiperparámetros encontrados mediante la técnica de optimización elegida.

2. Trabajos relacionados.

En cuanto a investigaciones realizadas, a nivel nacional, en el 2020 la investigación denominada optimización de los hiperparámetros de una máquina de regresión de soporte vectorial, utilizaron enjambre de partículas para el pronóstico de casos de COVID-19, aplicaron la optimización a los hiperparámetros de una máquina de regresión de soporte vectorial (SVR), mediante la adaptación de algoritmo metaheurístico Particle Swarm Optimization (PSO) para estudiar las series de tiempo del total de casos positivos acumulados por el COVID-19 en la ciudad de Bogotá [6].

Por otro lado, a nivel internacional, la investigación titulada *An Intelligent Learning System Based on Random Search Algorithm and Optimized Random Forest Model for Improved Heart Disease Detection*, se propone un sistema de aprendizaje híbrido denominado (RSA-FA) que combina dos algoritmos: búsqueda aleatoria (RSA) y bosque aleatorio (RF) [7].

En conclusión, como aspectos importantes para tener en cuenta para el proyecto actual, se destaca el problema del sobreajuste, que se relaciona con el problema principal como el aumento en la precisión de detección cardíaca en datos de prueba y disminución en los datos de entrenamiento, también, incluye la optimización hiperparamétrica dando las pautas para aplicar el RSA y el RF [7].

3. Formulación del modelo.

Para la aplicación de *Particle Swarm Optimization* (PSO) en Random Forest (RF) se aplica la obtención y el procesamiento de los datos, técnicas utilizadas en el proceso de optimización y las métricas de evaluación.

Por otro lado, para el desarrollo se ha optado por utilizar Python como lenguaje de programación debido a su flexibilidad, capacidad y amplia implementación en el manejo de los algoritmos de *Machine Learning* e optimización. Así mismo, Google Colab fue elegido entorno de aplicación dado su alta capacidad en términos de rendimiento, además permite la aplicación de diversas librerías como NumPy, Matplotlib, Scikit-learn, Pandas y Optunity las cuales fueron necesarias para el desarrollo en el proceso de optimización.

3.1. AutoML

Un sistema de *Automated machine learning* (AutoML) es un método que posibilita la construcción y desarrollo de modelos de ML, construyendo un conjunto de modelos ordenados en función de su desempeño. Con el fin de garantizar un resultado óptimo, el modelo de un sistema de AutoML aplica la optimización de hiperparámetros para hallar la mejor configuración dentro de un espacio de búsqueda [8].

Finalmente, se busca obtener un sistema que de manera automatizada que posibilite hallar determinadas tendencias en los datos con la mínima intervención humana, dando solución a tareas de mayor complejidad. Haciendo posible que cada proceso sea más eficiente y robusto, teniendo en cuenta restricciones computacionales y tiempos de ejecución. En consecuencia, las ventajas del AutoML provienen del adecuado proceso de HPO, pertenecientes o referentes a su canalización, preparación de los datos, elección de los atributos, la mejor selección de espacio de búsqueda, el proceso de HPO y finalmente su evaluación [5].

3.2. Hiperparámetros

Dentro de los algoritmos de *Machine Learning* en el contexto del AutoML, se encuentran dos tipos de parámetros, los parámetros del modelo, que son aquellos que se inicializan y actualizan en el proceso de aprendizaje, como por ejemplo los pesos de las neuronas en las redes neuronales, y aquellos de los cuales depende el modelo cuando se instancia, denominados hiperparámetros; los cuales definen la estructura del modelo de ML, y debido a esto no se pueden estimar en el aprendizaje de datos; por tanto, estos hiperparámetros que configuran la arquitectura del modelo de ML, definen en gran medida su rendimiento, por ello, el

AutoML busca optimizarlos, mediante el uso de técnicas de optimización aplicadas al espacio de búsqueda [4].

El conjunto de valores que pueden asumir los hiperparámetros se denomina espacio de búsqueda, también conocido como espacio de configuración, el cual representa el conjunto de todas las posibles combinaciones de hiperparámetros, en el que se busca el óptimo para realizar la mejor predicción posible [8]. Todas las configuraciones posibles de un método crean un espacio de hiperparámetros que puede ser discreto, continuo o mixto, dependiendo del tipo de hiperparámetros que puedan establecerse.

3.3. Conjunto de datos.

Para el desarrollo de la presente investigación se lleva a cabo un proceso de Benchmarking, mediante el análisis del dataset *Heart Failure* y el dataset *Bank Marketing*, los cuales corresponden a conjuntos de datos equilibrados y desequilibrados, respectivamente.

En ambos casos, el problema a abordar es multivariado; no obstante, al examinar las categorías de la etiqueta objetivo, se evidencia que para el conjunto de *datos Bank Marketing* existe un desequilibrio de clases (36548 instancias negativas y 4640 instancias positivas), mientras que para el conjunto de *datos Heart Failure* este desequilibrio es casi nulo (508 instancias positivas y 410 instancias negativas).

Heart Failure Prediction Data Set. Este dataset se asocia al sector médico, concretamente en la insuficiencia cardíaca, consta de 11 características y 918 instancias. El objetivo de clasificación es identificar la presencia de cardiopatía en el paciente, mediante la predicción de valores según el grado, específicamente de cero (sin presencia) a 4 (presencia de alto riesgo). Se debe agregar que en las diversas investigaciones se reemplazó y simplificó el objetivo a clasificación binaria, enfocándose en presencia (valores 1,2,3,4) y no presencia (valores 0) de suficiencia cardíaca.

Bank Marketing Data Set. El dataset cuenta con 41.188 instancias y 20 atributos. Se relaciona con campañas de marketing de una entidad bancaria de Portugal. El objetivo de la clasificación es predecir si el cliente suscribirá (sí/no) a un depósito bancario a plazo. También, es importante considerar que las observaciones se basan en marketing directo, esto mediante varias llamadas telefónicas según los requerimientos.

3.4. Técnicas utilizadas

Para obtener un rendimiento adecuado de la máquina de Random Forest y del algoritmo de PSO, se realizó un

análisis referente a los valores específicos de las variables que componen el dataset, lo cual arrojó resultados de procesamiento. Dicho lo anterior, para las variables categóricas se seleccionó el “one hot encoding” [9]. Por su parte, las variables numéricas no fueron normalizadas, dado que el *Random Forest* no es sensible a escalas [10]. Además, en cuanto a los valores perdidos se trataron por medio de imputación, utilizando la mediana y moda según la tipología del valor nulo [9]. Los pasos anteriores son necesarios para que los conjuntos de datos sean adecuados y no se presenten problemas al aplicarlos a la máquina de RF.

Por otro lado, dado que se analizó un conjunto de datos desequilibrado, se implementó el uso del remuestreo estratificado, validación cruzada de 5 folds y SMOTE. Con el fin de balancear el sobreajuste en el grupo de datos mediante datos sintéticos [11].

3.5. Métricas

Una vez entendido el algoritmo de optimización es importante analizar las métricas que serán utilizadas en la evaluación y comparación de rendimiento. La Tabla 1 identifica la matriz de confusión, que almacena de manera matricial los resultados reales y predictivos que permiten sencillez en el análisis de la clasificación [12].

En esta matriz el valor de “1” es tomado como etiqueta positiva y el valor de “0” es tomado como etiqueta negativa. Es de aclarar que en el caso de *Bank marketing* dataset el valor “1” significa la realización del depósito a término fijo y el valor “0” es la no realización. Por otro lado, en el dataset *Heart Failure* “1” es la presencia de insuficiencia cardíaca y “0” es la normalidad del paciente.

Real \ Predicción		Negativo	Positivo
		0 (No presencia – No depósito)	1 (Presencia – Depósito)
Negativo	0 (No presencia – No depósito)	Verdadero Negativo (VN)	Falso Positivo (FP) Error tipo I
Positivo	1 (Presencia – Depósito)	Falso Negativo (FN) (Error tipo II)	Verdadero Positivo (VP)

Tabla 1. Matriz de confusión. Fuente: Adaptada de [12].

Así mismo, las diferentes métricas utilizadas en la comparación de rendimiento se muestran a continuación:

Recall: El recall o sensibilidad es el objetivo de optimización en ambos conjuntos de datos. Debido a que se centra en minimizar los falsos negativos, es decir, aquellos casos en los que el modelo predice incorrectamente que un cliente no realizará un depósito cuando en realidad sí lo hará o que un paciente no padece la enfermedad cuando en realidad sí sucede (Error tipo II). La ecuación (1) muestra su cálculo en términos de la matriz de confusión.

$$Recall = \frac{VP}{VP + FN} \quad (1)$$

Accuracy: Accuracy o exactitud en este caso aborda la capacidad del modelo *Random Forest* para clasificar asertivamente los verdaderos positivos y verdaderos negativos (VN). La ecuación (2) indica el cálculo de esta métrica.

$$Accuracy = \frac{VP + VN}{VP + VN + FP + FN} \quad (2)$$

Precisión: Identifica qué tan preciso es el modelo para identificar los falsos positivos correctamente (Error tipo I). Su cálculo es mostrado en la ecuación (3).

$$Precision = \frac{VP}{VP + FP} \quad (3)$$

F1 Score: Proporciona una medida de equilibrio entre la precisión y el recall del modelo. Es decir, tiene en cuenta tanto los falsos positivos (Error tipo I) y falsos negativos (Error tipo II). Al evaluar esta métrica se enfatiza en las predicciones incorrectas en el modelo (FP y FN), el cálculo es mostrado en la ecuación (4).

$$F1 = \frac{2}{\frac{1}{precision} + \frac{1}{recall}} \quad (4)$$

4. Random Forest

La máquina de *Random Forest* es una técnica de *Machine Learning*, el cual se basa en la combinación de múltiples árboles de decisión, tiene la ventaja de utilizar muchas variables de distintos tipos como categóricas, numéricas y regresoras [13]. *Random Forest* son árboles independientes uno de otro, es decir, no se necesita saber cómo le fue a un árbol para entrenar el siguiente.

Algunas características de RF son su amplia aplicación en el campo investigativo, debido a la capacidad que tienen de soportar grandes conjuntos de datos y diversidad de hiperparámetros, tales como categóricos, numéricos y discretos, en donde se suele combinar distintas variables sin el requerimiento de un modelo paramétrico [13].

5. Optimización de Hiperparámetros(HPO)

La optimización de hiperparámetros se usa con el fin de mejorar el rendimiento de los métodos de ML de forma eficaz, ya que, de cierta manera, ajusta de manera ideal la arquitectura de estos métodos. Su desempeño depende del *tunning* (ajuste) de los hiperparámetros. Por lo tanto, al ajustar mediante técnicas de optimización se reducen errores manuales; además, se mejora el rendimiento de los algoritmos de ML y se generan modelos e investigaciones óptimas.

5.1. Hiperparámetros a optimizar

La selección de los hiperparámetros es parte fundamental en la construcción de modelos de *Machine Learning*. La selección adecuada de hiperparámetros permitirán que una máquina de *Random Forest* presente una mayor precisión, óptima generalización y eficiencia en cuanto a tiempo de entrenamiento y recursos computacionales; logrando minimizar los riesgos de un posible sobreajuste.

Los hiperparámetros de RF a optimizar son: *n estimators*, *Maximum tree depth (Max_depth)*, *max_depth*, *measuring function (Criterion)*, *The maximum number of features (max_features)*, *The minimum number of data points (min samples split y min samples leaf)*.

6. Particle Swarm Optimizacion (PSO)

El enjambre de partículas es un algoritmo evolutivo que se fundamenta en poblaciones biológicas, concretamente en comportamientos de un individuo de manera particular o conjunta. En PSO, el espacio de búsqueda es recorrido de manera semi-aleatoria por un enjambre, este es formado por un grupo de partículas que cooperan e intercambian cierta información para encontrar la HPC óptima. Cada partícula (S_i) es representada por un vector con la posición actual (x_i), la velocidad (v_i) y la mejor posición observada (p_i), como se demuestra en la ecuación (5):

$$S_i = \langle \vec{x}_i, \vec{v}_i, \vec{p}_i \rangle \quad (5)$$

El proceso principal que rige el funcionamiento de PSO se define por el siguiente proceso según [6]:

1. Definir n partículas en un enjambre S (Tamaño del enjambre).

2. Inicializar las partículas en el enjambre, estableciendo la posición inicial y velocidad.

3. Evaluar cada partícula registrando el P_{best} (Mejor solución de cada partícula) y G_{best} (Mejor solución global).

4. Modificar la velocidad y posición en función en función de dos factores, la posición anterior (P_{best}) y el óptimo global (G_{best}) encontrado por el enjambre. Ecuación (6) y ecuación (7).

$$\begin{aligned} V_j(i) &= V_j(i-1) \\ &+ C_1 r_1 * [P_{best} - x_j(i-1)] \\ &+ C_2 r_2 * [G_{best} - x_j(i-1)] \end{aligned} \quad (6)$$

Donde, j es una partícula del enjambre ($j=1, 2, 3, \dots, n$), i es la iteración o generación actual ($i=1, 2, 3, \dots, m$), C_1 y C_2 son los ritmos de aprendizaje cognitivo y social de las partículas y r_1 y r_2 son valores de probabilidad aleatorios.

$$X_j(i) = X_j(i-1) + V_j(i) \quad (7)$$

Donde, j es una partícula del enjambre ($j=1, 2, 3, \dots, n$) y i es la iteración o generación actual ($i=1, 2, 3, \dots, m$).

5. Se itera hasta alcanzar la convergencia o criterios de terminación.

Los factores para tener en cuenta al aplicar PSO como técnica principal son los siguientes: Primero, su fácil capacidad de paralelización. Segundo, en comparación con el algoritmo evolutivo GA, PSO tiene mayor velocidad y convergencia. Tercero, eficiencia en espacios de búsqueda con gran número, tipo y valores de hiperparámetros. En concordancia, estos factores lo hacen ideal en espacios de búsqueda complejos, razón por la cual será utilizado en este documento.

6.1. Aspectos de inicialización

El método propuesto consiste en aplicar el algoritmo PSO a *Random Forest* (RF), en este sentido, se discuten factores importantes en la aplicación de PSO y RF. En primer lugar, un aspecto crucial en la aplicación de PSO es el valor inicial de sus parámetros, dado que afectan su velocidad de convergencia y rendimiento general [4], [14]. Por ende, los valores adecuados de los parámetros de PSO se establecen de la siguiente manera:

Tamaño de enjambre. En el proceso de optimización actual se utilizó un tamaño de enjambre de 50 partículas, ya que, se considera que este valor proporcionar mejores rendimientos y capacidad para encontrar soluciones [14].

Posiciones iniciales partículas. La inicialización de las partículas se da de manera equivalente, dado que el

espacio de búsqueda es inexplorado en su totalidad [15]. Por otro lado, las secuencias de Sobol son utilizadas como técnica de muestreo dado que se garantiza una mejor uniformidad [15].

Velocidades iniciales partículas. El valor de la velocidad se inicializa de manera aleatoria. Dado que evita que el enjambre quede atrapado en óptimos locales [15], [16].

Iteraciones o generaciones. Frecuentemente el número de generaciones es usado como principal limitador. Por ello, dada la complejidad abordada en este documento se tomarán 10 generaciones de optimización [16].

Parámetros de velocidad y posición. C_1 y C_2 se definió un valor de 2 con el fin de equilibrar la velocidad de convergencia y la importancia del óptimo local y global para cada partícula [14].

Límites de velocidad y posición. Se limitó la velocidad y la posición de las partículas a fin de evitar el aumento incontrolado de la velocidad que ocasiona la divergencia del enjambre en la solución global (explosión del enjambre) y el abandono de las partículas del espacio de búsqueda (consecuencia del carácter vectorial de PSO) [15], [16].

Velocidad máxima. Al limitar la velocidad máxima, las partículas convergen con mayor facilidad hacia el óptimo deseado, dado que su tamaño de paso se vuelve más pequeño. Por tanto, se limita cada componente del vector la velocidad a un valor específico. La ecuación 8 muestra dicha limitación.

$$V_j(i) = \begin{cases} V_{max}, si & V_j(i) > V_{max} \\ -V_{max}, si & V_j(i) < -V_{max} \end{cases} \quad (8)$$

El valor de V_{max} se calculó según la proporción de la longitud de los espacios de búsqueda ecuación 9. En el

actual documento la velocidad se limitó con un PR de 30% para asegurar el equilibrio entre explotación y exploración en el algoritmo.

$$V_{max} = (b_d - a_d) * PR \quad (9)$$

En donde, d es la dimensión $d = 1, 2, 3, \dots, D$, b es el límite superior de posición, a es el límite inferior de posición y PR es la proporción asignada.

Posición factible. Al realizar los cálculos vectoriales de las actualizaciones, es común que las partículas tengan valores que excedan los límites de posición especificados. Esto se soluciona restringiendo el comportamiento de las partículas mediante sujeción mostrada en la ecuación 10 [14], [15].

$$X_j(i) = \begin{cases} X_{max}, si & X_j(i) > X_{max} \\ -X_{max}, si & X_j(i) < -X_{max} \end{cases} \quad (10)$$

Donde, X_{max} es la posición máxima que puede alcanzar la partícula y está dada por los límites en cada dimensión de optimización.

7. Método propuesto

Basado en lo anteriormente descrito, la Figura 1 muestra el algoritmo de optimización aplicado en esta investigación. El algoritmo inicia con la carga y análisis del conjunto de datos, este paso busca encontrar aquellas modificaciones en términos de características y estructuras que son necesarias de realizar en el dataset.

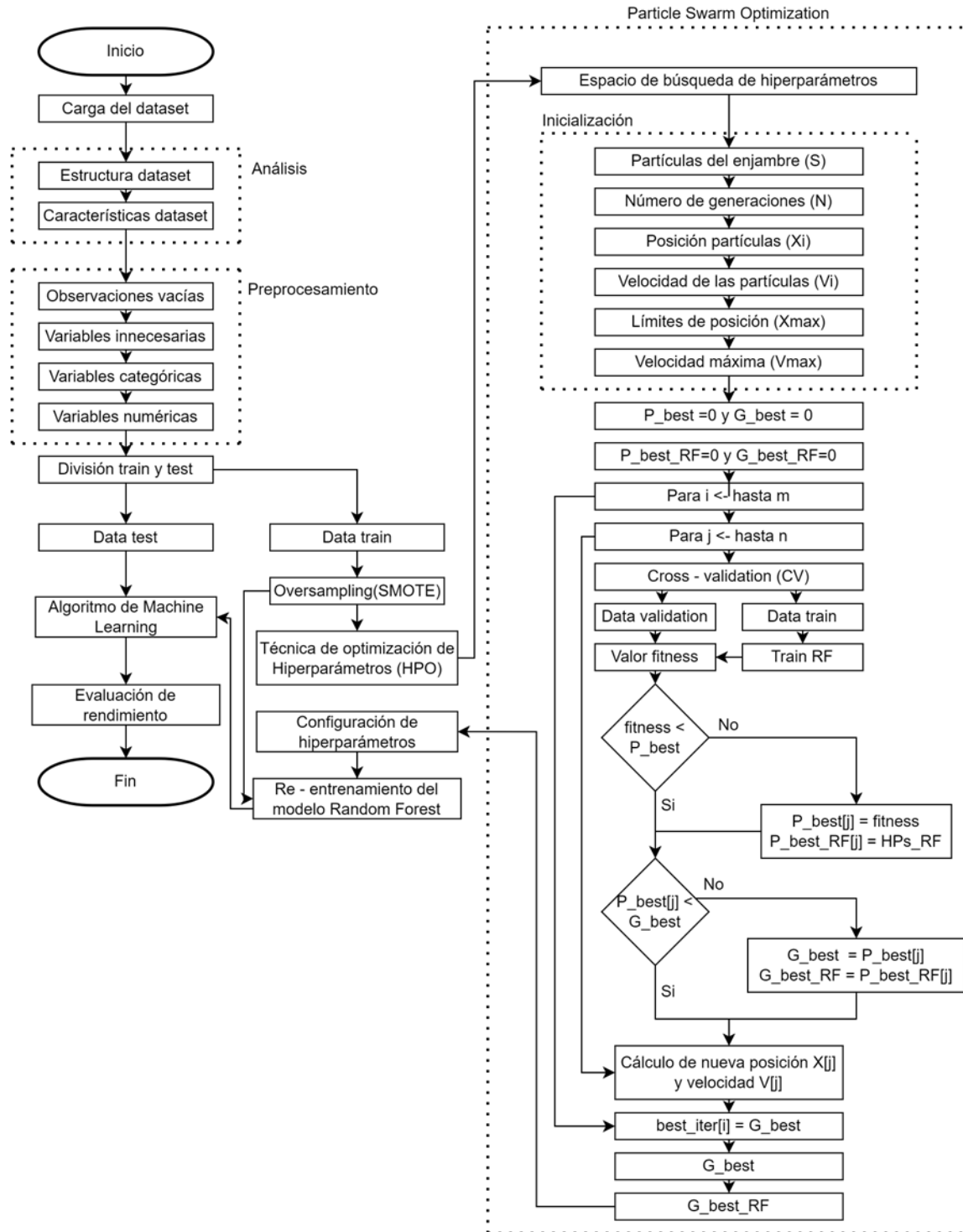


Figura 1. Algoritmo de optimización. **Fuente:** Elaboración propia.

Luego, el preprocesamiento es el encargado de realizar los cambios identificados previamente, tales como el reemplazo de observaciones vacías, eliminación de características innecesarias, codificación de variables categóricas y normalización de variables numéricas.

Luego de la etapa de preprocesamiento el conjunto de datos se dividió en entrenamiento-validación y prueba mediante muestreo estratificado en una proporción de 0.75 y 0.25 respectivamente [17]. El conjunto de prueba (test) se guardó para la evaluación final, mientras que el conjunto de entrenamiento y validación es sobremuestreado por la técnica SMOTE en conjunto de datos desequilibrados, a fin de equilibrar las instancias positivas mediante datos sintéticos [11]. Después, el conjunto resultante se ingresó en el algoritmo de optimización PSO + RF, el cual utilizó una validación cruzada de 5 folds, la cual es recomendada para tamaños de datasets medianos [18].

A continuación, el algoritmo PSO anteriormente descrito es aplicado en el modelo de *Random Forest*. Para finalmente entrenar el modelo con la configuración de HPs establecida por PSO, y reevaluarlo en un conjunto de datos de prueba, para encontrar el valor de la métrica e identificar el rendimiento real del modelo en escenarios desconocidos.

8. Comparación de métodos y benchmarking

Una vez planteado el algoritmo de optimización, se validó el rendimiento que le proporciona al modelo de RF. Para ello, en el proceso de *benchmarking* se comparó la técnica principal *Particle Swarm Optimization* (PSO) con dos métodos comúnmente utilizados en investigaciones, específicamente *Grid Search* (GS) y *Random Search* (RS), para analizar el comportamiento del modelo e identificar la técnica más efectiva en términos de funcionalidad y rendimiento en conjuntos de datos equilibrados y desequilibrados.

Las condiciones experimentales se garantizaron de forma equitativa para su comparación: preprocesamiento de variables (*One hot encoding*, imputación, y reemplazo), número de ejecuciones (500 ejecuciones dadas las 10 generaciones del PSO) y recursos computacionales (Colab notebook). A su vez, con el fin de realizar un ajuste adicional se consideró examinar dos tipos diferentes de combinaciones de espacios de búsqueda para los cuales se analiza el comportamiento de los hiperparámetros y los resultados de las métricas.

8.1. Construcción Espacios de búsqueda

Una vez revisados y analizados los hiperparámetros pertenecientes a la máquina de RF, se establecieron los rangos a optimizar para hallar la mejor configuración de hiperparámetros mediante la técnica de HPO. Los rangos son mostrados en la Tabla 2.

Tabla 2. Construcción de espacios de búsqueda. Fuente: Elaboración propia.

Hiperparámetro	Tipo	Espacios de búsqueda	
		No.1	No.2
n estimators	Entero	[10,100]	[50,200]
max depth	Entero	[5, 50]	[350,400,450]
criterion	Categorico	[Entropy, Gini]	
max features	Entero /Real	[1,64]	[12,16]
min samples split	Entero	[2,11]	[2,3]
min samples leaf	Entero	[1,11]	[1,5]

8.2. Aplicación de algoritmo

Dando inicio al procesamiento de los conjuntos de datos para la validación del algoritmo, se optó por inicializar la máquina de *Random Forest* sin aplicar las técnicas de optimización de hiperparámetros, con el fin de comparar el rendimiento del modelo. El procesamiento de la máquina se ejecutó con dos conjuntos de datos disponibles para realizar pruebas de rendimiento, un dataset con datos equilibrados (Heart Failure) y otro con datos desequilibrados (Bank Marketing).

Tabla 3. Resultados modelo básico de Random Forest.

Métricas	Bank Marketing		Heart Failure	
	Train	Test	Train	Test
Recall	1	0.56630	1	0.92125
Accuracy	1	0.91045	1	0.86086
Precision	1	0.59065	1	0.84172
F1- score	1	0.57822	1	0.87969

Al ejecutar el código en *Random Forest*, los resultados ilustrados en la Tabla 3 demuestran que el modelo se ajusta perfectamente a los datos de entrenamiento; sin

embargo, en datos no vistos la capacidad predictiva disminuyó, por tanto, se genera un sobreajuste en la totalidad de las métricas, el cual afecta negativamente el modelo, al no proporcionar un valor de estimación de rendimiento confiable en datos no vistos.

En consecuencia, se aplicó la técnica de optimización de hiperparámetros PSO para reducir el sobreajuste y aumentar el rendimiento en el problema de clasificación. Así mismo, el objetivo fue crear un espacio de búsqueda adecuado para seleccionar los mejores valores de los hiperparámetros de RF, logrando como resultado un modelo óptimo, aceptable y capaz de soportar el análisis de sistemas robustos.

9. Resultados

9.1. Heart Failure Dataset

Los resultados obtenidos en el conjunto de datos equilibrado (Tabla 4 y Tabla 5) demuestran que en el caso del espacio de búsqueda 1, PSO redujo el sobreajuste del modelo de manera significativa respecto al modelo básico de *Random Forest*. El rendimiento del modelo es mayor respecto a las demás técnicas de optimización. En concordancia, un aspecto mejorable se relaciona con el alto tiempo de ejecución, el cual puede verse ocasionado por la convergencia de las partículas hacia regiones del espacio de búsqueda que son robustas computacionalmente.

Tabla 4. Resultados valores de hiperparámetros de *Random Forest* (RF) en *Heart Failure*

Hiperparámetro	Espacio de Búsqueda	Particle Swarm Optimization	Random Search	Grid Search
n_estimators	[10,100]	71	40	10
max_depth	[5,50]	16	17	5
criterion	[Entropy, Gini]	Gini	Gini	Gini
max_features	[1,64]	1	1	1
min_samples_split	[2,11]	11	8	3
min_samples_leaf	[1,11]	4	8	1
n_estimators	[50,200]	89	117	50
max_depth	[350,450]	359	350	350
criterion	[0,2]	Entropía	Gini	Entropía
max_features	[12,16]	13	12	12
min_samples_split	[2,3]	3	3	2
min_samples_leaf	[1,5]	5	3	4

Por otro lado, del espacio de búsqueda número 2 se obtuvieron buenos resultados de sensibilidad (*Recall*)

para el conjunto tratado, siendo esta la métrica objetivo, la cual indicó que el modelo encontró de manera precisa

los pacientes que están en riesgo de padecer insuficiencia cardíaca. Igualmente, para las demás métricas el rendimiento fue superior con promedio un de 0.9279 en entrenamiento y 0.8882 en pruebas, reduciendo el sobreajuste y aumentando el desempeño en comparación

con RF y GS. Un aspecto a tener en cuenta para el *Random search*, es el mayor tiempo de ejecución respecto a las demás técnicas, esto se debe a evaluaciones computacionalmente exigentes, las cuales genera tal diferencia significativa.

Tabla 5. Métricas data set Heart Failure.

Métricas	Espacio de Búsqueda	Particle Swarm Optimization		Random Search		Grid Search	
		<i>Train</i>	<i>Test</i>	<i>Train</i>	<i>Test</i>	<i>Train</i>	<i>Test</i>
Accuracy	No.1	0.9026	0.8608	0.8720	0.8391	0.8851	0.8217
Precision		0.8886	0.8321	0.8581	0.8169	0.8813	0.8161
Recall		0.9422	0.9370	0.9212	0.9133	0.9160	0.8740
F1-score		0.9146	0.8814	0.8886	0.8624	0.8983	0.8441
Tiempo		6 min, 57 s		7 min, 42 s		1 min, 40 s	
		<i>Train</i>	<i>Test</i>	<i>Train</i>	<i>Test</i>	<i>Train</i>	<i>Test</i>
Accuracy	No.2	0.9215	0.8739	0.9520	0.8391	0.94912	0.85217
Precision		0.9181	0.8500	0.9507	0.8169	0.95052	0.82978
Recall		0.9422	0.9370	0.9632	0.9133	0.95800	0.92125
F1-score		0.9300	0.8913	0.9569	0.8624	0.95424	0.87313
Tiempo		14 min, 33 s		17 min, 6 s		7 min, 14 s	

En este caso, para manejar la eficacia de ejecución, el tiempo se puede utilizar como parámetro de comparación dado la igualdad de resultados en *Recall* en ambos espacios de búsqueda, es decir, si se requiere saber en menor tiempo cuáles pacientes necesitan ser tratados con urgencia, el que brindó una mejor solución es el espacio de búsqueda No.1, debido a que este tiene menor costo computacional en comparación con el espacio de búsqueda No.2. En esta situación específica el tiempo es un factor indispensable para el hallazgo eficaz de usuarios que padecen la enfermedad.

9.1.1. Inicialización de los métodos Particle Swarm Optimization, Random Search y Grid Search

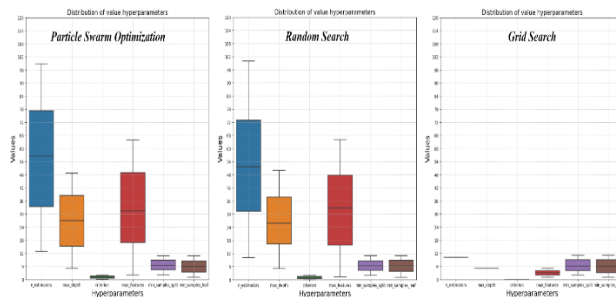


Figura 2. Inicialización de técnicas de HPO.

Se observa en la Figura 2 que en la primera generación los valores de las 50 partículas de *Particle Swarm Optimization* (PSO) se asimilan a los 500 valores aleatorios de *Random search* (RS), la simetría de los

diagramas de caja indica uniformidad en la generación de las posiciones iniciales. A su vez, el método de optimización *Grid search* (GS) en las 500 iteraciones no alcanzó a evaluar combinaciones con valores diferentes de algunos hiperparámetros, por lo que se necesitaría mayor cantidad de evaluaciones para alcanzar la uniformidad de PSO y RF.

Lo anterior, corrobora que el rendimiento de 50 iteraciones (50 partículas) de PSO es comparable con 500 iteraciones pseudo aleatorias de RS, en cuanto a explotación del espacio, lo que ocasiona que al momento de buscar el óptimo el enjambre de partículas tenga la capacidad de converger hacia regiones óptimas después de la primera iteración, mejorando el rendimiento respecto a RS y GS.

9.1.2. Comportamiento de partículas en generaciones

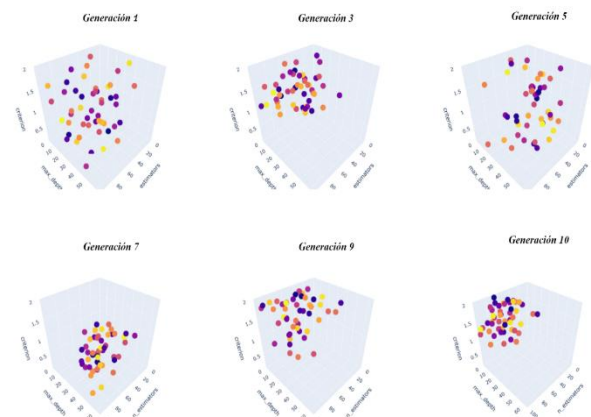


Figura 3. Comportamiento de las partículas en el espacio *criterion*, *max depth* y *n estimators*.

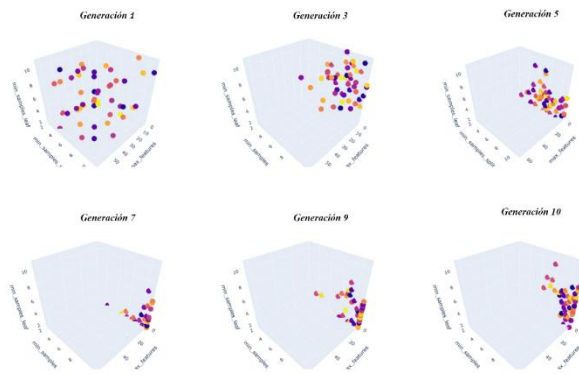


Figura 4. Comportamiento de las partículas en *min simples leaf*, *min simples Split* y *max features*.

Al analizar el comportamiento de las partículas a lo largo de las generaciones Figura 3 y Figura 4, se puede evidenciar el factor continuo propio del algoritmo de optimización. Al analizar las posiciones se encontró indicativos de regiones óptimas del espacio, en relación con los valores de hiperparámetros. Estos aspectos se discuten a continuación:

***criterion*:** Las partículas se mantuvieron en valores altos (>1) lo que sugiere que *entropy* debe ser utilizada para medir la calidad de las divisiones. Sin embargo, el valor óptimo se encontró como *Gini*. Esta situación se pudo dar por la sujeción y codificación necesaria para hiperparámetros categóricos, por tanto, sus límites deben mantenerse de manera similar.

***max depth*:** Los valores de este hiperparámetros se mantuvieron uniformes en el rango de optimización dado, por tanto, no se tiene indicios de una región óptima.

***n estimators*:** El número de árboles que se construyen en el modelo parece indicar valores adecuados mayores a treinta. Además, dado que no se abandona el espacio de búsqueda se considera que el límite superior es el óptimo.

***min simples leaf*:** Valores menores a ocho indican rendimiento superior para el número menor de muestras en un nodo hoja. Igualmente, los valores del límite superior no son rebasados al no encontrarse valores óptimos fuera del espacio de optimización.

***min simples Split*:** Valores mayores a seis para el número mínimo de muestras en la división de un nodo parecen ser los indicados, y dado que el espacio de búsqueda es rebasado podría pensarse en explorar valores superiores.

***max features*:** Valores superiores a sesenta y cuatro parecen ser reconocidos como óptimos por las partículas. Por tanto, si dentro de un contexto similar se necesita evitar que las partículas se sujeten al espacio, se recomienda aumentar el límite superior a los espacios de búsqueda proporcionados.

Los análisis anteriores sugieren regiones óptimas a explorar que podrían usarse para modificar el espacio de búsqueda inicial, a fin de mejorar la convergencia de las partículas a la solución óptima.

9.2. Bank Marketing

Inicialmente, se aplicó el remuestreo estratificado para el desarrollo del modelo, para lo cual, *Recall* se eligió como métrica clave del dataset; en el primer espacio de búsqueda el modelo logró capturar correctamente 66,1% de las muestras positivas y en los datos de prueba solo 54,39%. De manera similar, para el segundo espacio de búsqueda capturó el 71,4% de las muestras positivas, indicando adecuadamente las instancias positivas durante el entrenamiento. Sin embargo, en el conjunto de prueba se capturó solo el 54% de las muestras positivas reales, revelando que el modelo presentó dificultades al generalizar adecuadamente nuevos datos. Una razón de este inconveniente se debe al sesgo o desequilibrio aún presente en el dataset, lo que causó una disminución en la sensibilidad para detectar instancias positivas.

En consecuencia, se aplicó la técnica de SMOTE a fin de equilibrar las clases. Es importante destacar que esta técnica no interfiere en el proceso de optimización de PSO, sino que se usa exclusivamente para abordar el desequilibrio presente en el conjunto de datos. Por ello, se aplicaron las distintas técnicas de optimización de hiperparámetros para encontrar aquella con mejor rendimiento.

Tabla 6. Valores de hiperparámetros de Random Forest (RF) según la técnica de optimización.

Métricas	Espacio de Búsqueda	Particle Swarm Optimization	Random Search	Grid Search
n_estimators	[10,100]	36	57	10
max_depth	[5,50]	5	6	5
criterion	[Entropy, Gini]	Entropy	Entropy	Gini
max_features	[1,64]	25	18	1
min_samples_split	[2,11]	5	5	2
min_samples_leaf	[1,11]	5	2	5
n_estimators	[50,200]	50	68	50
max_depth	[350,450]	433	411	350
criterion	[0,3]	Entropía	Entropía	Entropía
max_features	[12,16]	13	13	14
min_samples_split	[2,3]	2	2	2
min_samples_leaf	[1,5]	5	5	4

Los resultados de la Tabla 6 y Tabla 7 se discuten a continuación. Para el espacio de búsqueda No.1 indican que la métrica objetivo obtiene su valor más alto con el algoritmo PSO. Las demás métricas que complementan el estudio arrojaron mejores resultados para *Random search* con un costo computacional más alto. Por su parte GS no obtuvo resultados favorables en el dataset, debido principalmente a la evaluación de solo una parte de las configuraciones de hiperparámetros en la cuadrícula combinatoria.

De igual forma, el espacio de búsqueda No.2 obtuvo mejor rendimiento en las métricas para PSO, RS Y GS respecto al modelo *Random Forest* con HPs predeterminados. Además, comparando PSO con RS, no se encontró un rendimiento superior. El tiempo de ejecución se redujo considerablemente, por lo cual se obtuvieron desempeños similares con presupuestos más bajos. Adicionalmente, PSO superó a GS en el objetivo de optimización indicando ser la mejor de las técnicas de optimización.

Tabla 7. Métricas data set Bank Marketing data set.

Métricas	Espacio de Búsqueda	Particle Swarm Optimization		Random Search		Grid Search	
		<i>Train</i>	<i>Test</i>	<i>Train</i>	<i>Test</i>	<i>Train</i>	<i>Test</i>
Accuracy	No.1	0.9008	0.8568	0.9116	0.8676	0.7455	0.7839
Precision		0.8568	0.4352	0.8698	0.4555	0.7777	0.2793
Recall		0.9624	0.9094	0.9682	0.8965	0.6875	0.5810
F1-score		0.9065	0.5887	0.9164	0.6041	0.7298	0.3772
Tiempo		3h,16 min		6 h,13 min		13 min	
Accuracy	No.2	0.9733	0.9096	0.9731	0.9117	0.9100	0.9789
Precision		0.9656	0.5849	0.9658	0.5915	0.9739	0.5877
Recall		0.9815	0.6827	0.9808	0.6991	0.9841	0.6758
F1- score		0.9735	0.6300	0.9733	0.6408	0.9790	0.6287
Tiempo		5h, 44 min		10 h, 49 min		3 h,64 min	

Analizando ambos espacios de búsqueda se encontró que el No.1 aportó mayor capacidad predictiva y redujo el sobreajuste en la combinación con PSO. Por su parte, el No.2 aunque logró mejoras, fueron menores al espacio No. 1 en términos de desempeño. Por ello, se muestra el

análisis de inicialización en la Figura 5, en el que se analiza las distribuciones en el espacio con mayor rendimiento.

9.2.1. Inicialización de los métodos Particle Swarm Optimization, Random Search y Grid Search.

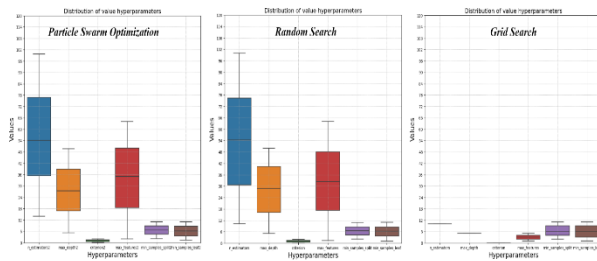


Figura 5. Inicialización de técnicas de HPO.

Dicho lo anterior, es importante considerar la figura 4, la cual describe el espacio de búsqueda abarcado por la primera iteración de PSO respecto a RS y GS. De la misma manera que para el conjunto de datos “Heart Failure”, el espacio de búsqueda abarcado por la técnica de optimización PSO es similar a RS, lo que confirma la uniformidad lograda en la primera generación. Por otro lado, GS no tuvo evaluaciones suficientes para abarcar un mayor espacio, por tanto, su bajo rendimiento en comparación con RS y PSO es evidente. A partir de lo anterior se infiere que independiente del conjunto de datos estudiado, el rendimiento de PSO en cuanto a la explotación y exploración del espacio es mayor que para las demás técnicas, dada su capacidad de convergencia.

9.2.2. Comportamiento de partículas en generaciones

Al analizar las posiciones (Figura 6 y 7) se encontraron aspectos que pueden ayudar a inferir en la elección de espacio de búsqueda a fin de mejorar los rendimientos del modelo.

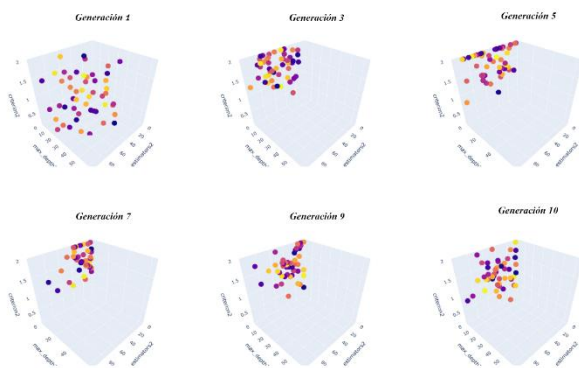


Figura 6. Comportamiento de las partículas en el espacio *criterion*, *max depth* y *n estimators*

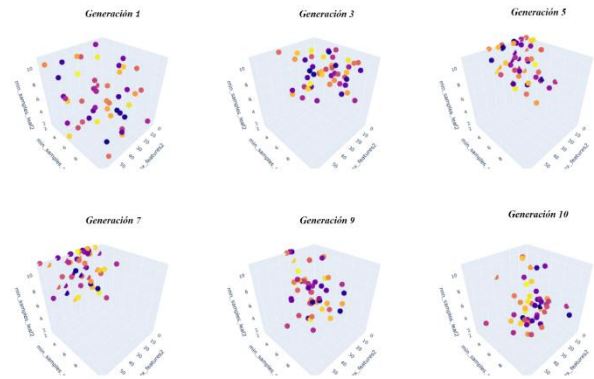


Figura 7. Comportamiento de las partículas en *min simples leaf*, *min simples Split* y *max features*.

criterion: Valores mayores a 1 se siguen manteniendo como objetivo de las partículas, sin embargo, con el pasar de las generaciones estas regresan a valores bajos, que en este caso es acorde con los resultados obtenidos. Por tanto, se recomienda mantener los valores de *criterion* para la optimización dado que los valores óptimos obtenidos varían sobre estos.

max depth: Para la profundidad, las cifras indican valores menores a 30 y dado que las partículas no abandonan el espacio de búsqueda su límite superior se puede mantener.

n estimators: El número de árboles en el modelo no indica la ubicación de una región óptima, dado por la complejidad del dataset.

min simples leaf: El valor de este hiperparámetro no sugiere regiones del espacio óptimas, por lo cual las partículas no convergen a un valor específico.

min simples Split: La cifra varía con el pasar de las iteraciones. Sin embargo, en las últimas iteraciones se presenta que valores mayores a 4 son tomados como óptimos.

max features: Para este hiperparámetro las partículas parecen explorar valores entre diez y cuarenta para las últimas generaciones lo que indicaría la región óptima del espacio.

Los análisis sugieren que las regiones óptimas a explorar podrían modificar la convergencia proporcionando mayor facilidad para encontrar la solución óptima. Adicionalmente, contrario al anterior conjunto de datos, las partículas tuvieron mayor dificultad para encontrar el óptimo, esto puede estar dado por la complejidad del problema, la calidad de los datos y los parámetros seleccionados para la técnica SMOTE.

9.3. Discusión

Al analizar los valores de entrenamiento y prueba, se encontró que las soluciones óptimas halladas por parte del modelo obtuvieron mejoras de rendimiento moderadas en comparación con las generaciones iniciales. Con el pasar del tiempo, se evidencia es la reducción del sobreajuste que proporciona un modelo más confiable a la hora de evaluar datos no vistos.

De manera general, los resultados de las métricas fueron satisfactorias y mejoraron significativamente la capacidad predictiva en comparación a los valores arrojados por el modelo de *Random Forest* ejecutado sin técnica de optimización, demostrando la efectividad de los hiperparámetros ajustados dado el óptimo desempeño en los problemas de clasificación tratados.

También, aspectos asociados a los valores arrojados indicaron que la elección del espacio de búsqueda puede influir directamente en los resultados y en el tiempo de ejecución, por tanto, es importante la construcción de espacios computacionalmente óptimos y que a su vez tengan valores adecuados de rendimiento.

Igualmente, al comparar PSO con otras técnicas de optimización de hiperparámetros, como *Random Search* y *Grid Search* que han sido las más usadas y sencillas de manejar, es notable que de *Random Forest* con PSO se obtienen resultados superiores a diferencia de las otras técnicas, siendo un indicativo de que los hiperparámetros ajustados son adecuados.

10. Conclusiones

Se destaca la relevancia significativa que tuvo para el modelo de clasificación *Random Forest* el uso de la técnica de optimización de hiperparámetros PSO, dado que no solo mejoró la eficacia general del modelo, sino que también permitió obtener valores óptimos de hiperparámetros que mejoraron el rendimiento métricas y redujeron el sobreajuste.

El uso de la validación cruzada (CV) como técnica adicional proporciona una evaluación más realista en la ejecución de la optimización de hiperparámetros, pues permitió encontrar la mejor configuración dentro del espacio de búsqueda, garantizando una mejora en el rendimiento del modelo en datos no vistos.

El algoritmo de PSO mejora significativamente el rendimiento y reduce el sobreajuste de los algoritmos de *Machine Learning*. Sin embargo, dado su naturaleza vectorial, PSO maneja principalmente hiperparámetros continuos, por lo que es necesario un proceso de codificación y decodificación en el caso de hiperparámetros discretos y categóricos. Esto ocasiona

evaluaciones de la función fitness innecesarias, que podrían eliminarse del algoritmo con alguna modificación a sus fórmulas de actualización de velocidad.

Aún sin usar la técnica de SMOTE para manejar el desequilibrio de clases en el dataset desequilibrado, se observó que la aplicación de la técnica de PSO por sí solo tiene un impacto favorable en los resultados de optimización del modelo de *Random Forest*. PSO lleva a cabo una búsqueda exhaustiva en el espacio de búsqueda de los hiperparámetros, ajustando y depurando aquellos valores óptimos para obtener un rendimiento superior del modelo.

11. Recomendaciones

Se recomienda que para conjuntos de datos desequilibrados sea usado el remuestreo estratificado como método para reducir el sesgo, el cual es dado por la falta de representación de algunas etiquetas objetivo en el conjunto de datos. Esta técnica mejora la capacidad predictiva del modelo al proporcionar información referente a las etiquetas con menor proporción al modelo *Random Forest*.

Aunque SMOTE es una técnica práctica para abordar el balance de los grupos desequilibrados en un modelo de clasificación, se recomienda tener precaución al seleccionar las instancias usadas como vecinos, ya que la calidad de las nuevas muestras generadas por la técnica depende de dicha selección. Hay que tener en cuenta que, si seleccionan instancias muy agrupadas las nuevas observaciones sintéticas generadas pueden llegar a ser redundantes y no aportar valor al modelo. Mientras que, si las instancias seleccionadas son muy diferentes entre sí, pueden afectar negativamente el desempeño del modelo de clasificación, reduciendo su rendimiento al contar con información no representativa.

El tratamiento de las partículas del enjambre que abandonan el espacio de búsqueda puede modificarse por uno más adecuado. Debido principalmente al hecho de que, al sujetar los límites, las partículas pueden seguir dirigiendo el enjambre a espacios de configuración no factibles, hecho que genera que las partículas lo abandonen en mayor medida.

PSO limita tanto la velocidad como el espacio de búsqueda a fin de evitar el fenómeno conocido como explosión del enjambre. Aunque el problema se solucionó parcialmente al incluir un tamaño de paso pequeño, aún se puede ver un rendimiento sub-óptimo de optimización. Por ello, variaciones como Lbest PSO y Gbest PSO pueden ser implementadas a fin de comparar su rendimiento con el modelo básico.

Para la realización de futuras investigaciones enfocadas en la optimización de hiperparámetros, es importante considerar la posibilidad de encontrarse con escenarios donde los rangos del espacio de búsqueda no sean óptimos para configurar la cantidad de variables o hiperparámetros existentes. Para dichos casos, se sugiere acotar el espacio de configuración mediante técnicas de reducción de la complejidad del espacio. Lo cual puede ayudar a limitar las opciones y enfocar la búsqueda en zonas más prometedoras, facilitando encontrar la configuración óptima de hiperparámetros.

12. Referencias

- [1] Z. Xu and Y. Shi, "Exploring Big Data Analysis: Fundamental Scientific Problems," *Annals of Data Science*, vol. 2, no. 4, pp. 363–372, 2015, doi: 10.1007/s40745-015-0063-7.
- [2] R. Rawat and R. Yadav, "Big Data: Big data analysis, issues and challenges and technologies," in *IOP Conference Series: Materials Science and Engineering*, 2021. doi: 10.1088/1757-899X/1022/1/012014.
- [3] A. A. Prudius, A. A. Karpunin, and A. I. Vlasov, "Analysis of machine learning methods to improve efficiency of big data processing in Industry 4.0," in *Journal of Physics: Conference Series*, Institute of Physics Publishing, Nov. 2019. doi: 10.1088/1742-6596/1333/3/032065.
- [4] L. Yang and A. Shami, "On hyperparameter optimization of machine learning algorithms: Theory and practice," *Neurocomputing*, vol. 415, pp. 295–316, Nov. 2020, doi: 10.1016/j.neucom.2020.07.061.
- [5] M. Wistuba, A. Rawat, and T. Pedapati, "AutoML: A survey of the state-of-the-art," *Knowl Based Syst*, vol. 1, pp. 1–27, Nov. 2021, [Online]. Available: <http://arxiv.org/abs/1905.01392>
- [6] N. D. Muñoz Cañón and J. A. Romero Triana, "Optimización de los hiperparámetros de una máquina de regresión de soporte vectorial utilizando enjambre de partículas para el pronóstico de casos de COVID-19," *Revista UIS Ingenierías*, vol. 20, no. 2, pp. 181–196, Feb. 2021, doi: 10.18273/revuin.v20n2-2021015.
- [7] A. Javeed, S. Zhou, L. Yongjian, I. Qasim, A. Noor, and R. Nour, "An Intelligent Learning System Based on Random Search Algorithm and Optimized Random Forest Model for Improved Heart Disease Detection," *IEEE Access*, vol. 7, pp. 180235–180243, 2019, doi: 10.1109/ACCESS.2019.2952107.
- [8] Management solutions, "Auto Machine Learning, hacia la automatización de los modelos," Madrid (España), 2020. [Online]. Available: www.managementsolutions.com
- [9] Z. Yang and A. Zhang, "Hyperparameter optimization via sequential uniform designs," *Journal of Machine Learning Research*, vol. 22, 2021.
- [10] D. Zheng, C. Qin, and P. Liu, "Adaptive Particle Swarm Optimization Algorithm Ensemble Model Applied to Classification of Unbalanced Data," *Sci Program*, vol. 2021, 2021, doi: 10.1155/2021/7589756.
- [11] A. Fernández, S. García, M. Galar, R. C. Prati, F. Herrera, and B. Krawczyk, *Learning from Imbalanced Data Sets*, 1st ed., vol. 1. Springer Nature Switzerland SG, 2018. doi: <https://doi.org/10.1007/978-3-319-98074-4>.
- [12] N. Zhu, C. Zhu, L. Zhou, Y. Zhu, and X. Zhang, "Optimization of the Random Forest Hyperparameters for Power Industrial Control Systems Intrusion Detection Using an Improved Grid Search Algorithm," *Applied Sciences (Switzerland)*, vol. 12, no. 20, 2022, doi: 10.3390/app122010456.
- [13] L. Alai, B. Taleb, D. Salgado, E. Rosa, and R. Alonso, "Imputación de datos mediante Random Forest," 2021.
- [14] J. Chand Bansal, P. Kumar Singh, and N. R. pal, *Evolutionary and Swarm Intelligence Algorithms*, 1st ed., vol. 779. Springer Cham, 2019. doi: <https://doi.org/bibliotecavirtual.uis.edu.co/10.1007/978-3-319-91341-4>.
- [15] K. E. Parsopoulos and M. N. Vrahatis, *Particle swarm optimization and intelligence : advances and applications*, 1st ed., vol. 1. New York: Information Science Reference, 2010.
- [16] D. Simon, *Evolutionary optimization algorithms*, 1st ed., vol. 1. Honoken: John Wiley & Sons, Inc., 2013.
- [17] Louis. Owen, *Hyperparameter tuning with Python : boost your machine learning model's performance via hyperparameter tuning.*, 1st ed., vol. 1. Birmingham: Packt Publishing Ltd., 2022.
- [18] B. Bischl *et al.*, "Hyperparameter optimization: Foundations, algorithms, best practices, and open challenges," *Wiley Interdiscip Rev Data Min Knowl Discov*, 2023, doi: 10.1002/widm.1484.