ISSN 1870-4069

Selección óptima de variables mediante metaheurísticas binarias para la detección de fallas en un proceso industrial multivariado usando aprendizaje de máquina

Jesús Alejandro Navarro Acosta[1], Edgar O. Reséndiz Flores[2], Irma D. García Calvillo[1]

[1] Universidad Autónoma de Coahuila, Centro de Investigación en Matemáticas Aplicadas, Saltillo, Coahuila
[2] Instituto Tecnológico de Saltillo, División de Estudios de Posgrado e Investigación. Saltillo, Coahuila
alejandro.navarro@uadec.edu.mx, eresendiz@itsaltillo.edu.mx

Resumen. Este trabajo presenta la comparación entre la optimización por enjambre de partículas y el algoritmo de búsqueda gravitacional en sus versiones binarias en combinación con la distancia Mahalanobis para la selección óptima de variables en un proceso de soldadura de discos de soporte para el mecanismo de los asientos en el automóvil. Los resultados muestran que la metodología implementada es capaz de seleccionar un subconjunto de variables capaz de mantener e incluso aumentar la exactitud en la detección de fallas en el proceso, así como brindar a los encargados del proceso las variables que intervienen en la variabilidad del mismo. Lo anterior se valida mediante la implementación de 16 algoritmos y enfoques de aprendizaje automático supervisado y otro no supervisado ampliamente utilizados para tareas de reconocimiento de patrones y detección de fallas.

Palabras clave: metaheurísticas binarias, distancia Mahalanobis, aprendizaje automático.

Optimal Feature Selection through Binary Metaheuristics for Fault Detection in a Multivariate Industrial Process Using Machine Learning

Abstract. This work presents the comparison between binary particle swarm optimization and binary gravitational search algorithm in combination with the Mahalanobis distance for the optimal feature selection in a welding process of support discs for the seat mechanism in the car. The results show that the implemented methodology is able to select a subset of variables capable of maintaining and even increasing the accuracy for fault detection in the process. This is validated through

the implementation of several supervised and one unsupervised machine learning algorithms, which are widely used for pattern recognition tasks.

Keywords: binary metaheuristics, Mahalanobis distance, machine learning.

1. Introducción

En la actualidad el sector industrial se enfrenta día a día con el reto de lograr altos niveles de calidad en sus productos. Por tal motivo, los diferentes departamentos involucrados en la producción de dichos productos implementan diversos procedimientos y técnicas con el fin de minimizar los defectos de calidad. Sin embargo debido al gran avance de la tecnología, los procesos son cada vez más complejos demandando técnicas y metodologías cada vez más sofisticadas y robustas para su análisis. En la industria actual los procesos pueden presentar características no lineales, presencia de ruido y una alta cantidad de variables que pueden presentar correlación entre si. Dichas características han hecho complicado y en muchas veces ineficiente el análisis de estos procesos mediante técnicas clásicas. Actualmente la inteligencia artificial ha demostrado resultados satisfactorios al analizar procesos y conjuntos de datos con las características antes mencionadas mediante el desarrollo de sistemas para la detección temprana de fallas, y así evitar defectos de calidad en componentes o en el producto final. De acuerdo con Chiang [4] una falla se define como una desviación no autorizada de al menos una característica o variable del sistema. En este sentido un algoritmo que sea capaz de reconocer patrones puede ser implementado con el fin de detectar fallas en un sistema o proceso. En [19], Vapnik establece que junto con regresión y estimación de la densidad, la clasificación es una de las tres tareas principales del aprendizaje automático, es decir clasificación se refiere a construir un algoritmo capaz de clasificar una muestra en un conjunto de clases dadas sin implicación humana. Siguiendo esta lógica un algoritmo de aprendizaje automático funcional será capaz de distinguir a partir de un conjunto de datos una falla presente en el proceso con un determinado porcentaje de error. Debido a que obviamente se busca disminuir el error, además del desarrollo de algoritmos más potentes se llevan a cabo varios enfoques para disminuir el error en algoritmos ya existentes. Uno de estos enfoques es el preprocesamiento de datos el cual tiene como objetivo adecuar la información para su posterior análisis, dicho preprocesamiento puede incluir la extracción de características, la limpieza de ruido en señales y la selección de variables entre otros. Debido al avance de la tecnología y la informática la capacidad de adquirir información de los procesos industriales ha aumentado en gran manera. Por lo tanto, los conjuntos de datos a analizar son cada vez más grandes no solo en el número de muestras u observaciones que se pueden obtener sino también en el número de variables que interactúan entre si. Por tal motivo el enfoque de selección de variables se ha vuelto uno de los preprocesamientos mas estudiados. El objetivo de este enfoque es encontrar un subconjunto de variables

el cual permita describir los procesos de igual manera o mejor que el conjunto completo de datos. Esta idea se sustenta en el hecho de que en conjuntos de alta dimensionalidad puede existir información redundante o poco útil la cual aumenta la complejidad de los modelos para la detección de fallas. Además estos se vuelven computacionalmente costosos [3], es decir produce modelos con baja exactitud y que consumen gran tiempo al llevar a cabo la fase de entrenamiento. Como se mencionó anteriormente el analizar un proceso industrial con el fin de detectar fallas es una tarea desafiante tomando en cuenta las características de los mismos en la actualidad. Por tal motivo este trabajo de investigación se enfoca en la implementación de dos metaheurísticas binarias para la construcción de un modelo el cual sea capaz de seleccionar las variables útiles y así mejorar la detección de fallas en un proceso industrial multivariado.

2. Técnicas

2.1. Metaheurísticas

La optimización estocástica es la clase general de algoritmos y técnicas que emplean cierto grado de aleatoriedad para encontrar soluciones óptimas (o tan óptimas como sea posible) a problemas difíciles. Las metaheurísticas son los más generales de este tipo de algoritmos, y se aplican a una gama muy amplia de problemas [9]. En la literatura no existe una única definición de metaheurística, una de ellas es: "Una metaheurística es un conjunto de conceptos que pueden ser utilizados para definir métodos heurísticos que se pueden aplicar a un amplio conjunto de problemas diferentes. En otras palabras, una metaheurística puede ser vista como un marco algorítmico general que puede aplicarse a diferentes problemas de optimización con relativamente pocas modificaciones para hacerlas adaptables a un problema específico" [13]. Los algoritmos metaheurísticos son capaces de resolver problemas complejos de optimización donde otros métodos de optimización no son eficaces o eficientes.

Optimización por enjambre de partículas (PSO). El algoritmo de optimización con enjambre de partículas (PSO) fue desarrollado por J. Kennedy y R. C. Eberhart, el cual es un método de optimización metaheurístico basado en la simulación del comportamiento social de las aves, abejas, bancos de peces, entre otros. A diferencia de otros algoritmos evolutivos como algoritmos genéticos (GA), PSO es más simple ya que no contiene operaciones de mutación o cruce, lo que reduce la complejidad del modelo [7]. En PSO cada individuo en el enjambre es representado por un vector en un espacio de búsqueda multidimensional. Este vector tiene un vector de velocidad asignado que determina el siguiente movimiento de la partícula. Cada partícula actualiza su velocidad basándose en la velocidad actual y la mejor posición que ha explorado hasta ahora. Y también basándose en la mejor posición explorada por todo el enjambre [17]. PSO fue originalmente desarrollado para espacios de valores continuos, sin embargo muchos problemas son definidos para espacios de valor discreto donde el dominio de las variables es finito.

PSO binario. En 1997, Kennedy and Eberhart [8] introducen una versión binaria de PSO para problemas de optimización discretos. En PSO binario (BPSO) cada partícula representa su posición en valores binarios, donde cada valor de la partícula puede ser cambiado de 1 a 0 o viceversa.

Considere un espacio de búsqueda d-dimensional y a la i-ésima partícula del enjambre denotada como $\mathbf{x}_i = (x_{i1}, x_{i2}, ..., x_{id})^t$ cuya velocidad es $\mathbf{v}_i = (v_{i1}, v_{i2}, ..., v_{id})^t$. Por lo tanto las ecuaciones que guían la dinámica de la partícula en PSO continuo son:

$$\mathbf{v}_i(t+1) = w\mathbf{v}_i(t) + c_1\phi_1(\mathbf{p}_{ibest} - \mathbf{y}_i(t)) + c_2\phi_2(\mathbf{p}_{gbest} - \mathbf{y}_i(t)), \tag{1}$$

$$\mathbf{x}_i(t+1) = \mathbf{y}_i(t) + \mathbf{v}_i(t+1), \tag{2}$$

donde $\mathbf{v}_i(t+1)$ y $\mathbf{x}_i(t+1)$ son la velocidad y posición siguientes de cada partícula. Mientras c_1 y c_2 son constantes positivas, y ϕ_1 y ϕ_2 son variables aleatorias con distribución uniforme entre 0 y 1. Las mejores posiciones locales y globales son $\mathbf{p}_{ibest}$ y $\mathbf{p}_{gbest}$ respectivamente. El peso de inercia sobre un nuevo vector se denota por w[5].

La principal diferencia entre PSO y BPSO es la forma en que las velocidades cambian para actualizar las posiciones de las partículas. La versión binaria necesita una función para generar valores en el conjunto binario $\{0, 1\}$ que son los únicos estados posibles de la partícula, por lo tanto la función sigmoide es utilizada para decidir la probabilidad de generar tales números binarios. Esta función se aplica bit a bit en una partícula dada y se define como sigue:

$$sigm(v_{ik}) = \frac{1}{1 + e^{-v_{ik}}}. \tag{3}$$

La posición de la partícula correspondiente se actualiza en base a la siguiente regla

$$y_{ik}(t+1) = \begin{cases} 1 \text{ si } r < sigm(v_{ik}) \\ 0 \text{ De lo contrario,} \end{cases} \tag{4}$$

donde r es un número pseudo aleatorio seleccionado de una distribución uniforme entre $[0, 1]$.

Algoritmo de busqueda gravitacional (GSA). GSA es un algoritmo de optimización basado en la ley de la gravedad y la interacción de las masas. En la ley gravitacional de Newton las partículas se atraen unas a otras con una fuerza gravitacional. Dicha fuerza entre dos cuerpos es directamente proporcional al producto de sus masas e inversamente proporcional al cuadrado de sus distancias. En este algoritmo cada masa cuenta con posición, masa inercial, masa gravitacional activa y otra pasiva. La posición de la masa corresponde a una posible solución, y sus masas gravitacionales e inerciales son determinadas

usando una función de ajuste. Los autores en [15] establecen que GSA obedece a las siguientes leyes:

- Ley de gravedad: Cada partícula atrae a todas las demás partículas y la fuerza gravitacional entre dos partículas es directamente proporcional al producto de sus masas e inversamente proporcional a la distancia entre ellas.
- Ley de movimiento: La velocidad actual de cualquier masa es igual a la suma de la fracción de su velocidad anterior y la variación en la misma. La variación en la velocidad o aceleración de cualquier masa es igual a la fuerza en el sistema dividida por la inercia.

La velocidad y la posición de cada masa está dada por:

$$\mathbf{v}_i^d(t+1) = rand_i \times \mathbf{v}_i^d(t) + a_i^d(t), \tag{5}$$

$$X_i^d(t+1) = x_i^d(t) + \mathbf{v}_i^d(t+1), \tag{6}$$

donde $a_i^d(t)$ es la aceleración del agente i en el tiempo t y en dirección a d, y se calcula de la siguiente manera:

$$a_i^d(t) = \frac{F_i^d(t)}{M_{ii}(t)}. \tag{7}$$

En un espacio $d-$dimensional la fuerza total que actúa sobre un agente i es

$$F_i^d(t) = \sum_{i \neq j} rand_j F_{ij}^d. \tag{8}$$

Las masas gravitacionales e inerciales estan dadas por:

$$M_i(t) = \frac{q_i(t)}{\sum_{j=1} q_j(t)}, \tag{9}$$

donde,

$$q_i(t) = \frac{fit_i - worst(t)}{best(t) - worst(t)}, \tag{10}$$

para problemas de minimización $best(t)$ y $worst(t)$ se calculan como:

$$best(t) = \min\ fit_j(t), \tag{11}$$

$$worst(t) = \max\ fit_j(t), \tag{12}$$

$$F_{ij}^d = G(t)\frac{M_{pi}(t) \times M_{aj}(t)}{R_{ij}(t) + \varepsilon}(x_j^d(t) - x_i^d(t)). \tag{13}$$

M_{aj} es la masa gravitacional activa aplicada al agente j, M_{pi} es la masa gravitacional pasiva relacionada al agente i. $G(t)$ es la constante gravitacional en el tiempo t, ε es una constante, y $R_{ij}(t)$ es la distancia Euclidea entre los agentes i y j. Para detalles de las ecuaciones revise [15] y [11].

Al igual que PSO, GSA se modificó para resolver problemas binarios. En [16] se presenta una versión binaria de GSA (BGSA) donde la actualización del valor de la posición de la masa alterna entre 0 y 1. Esto se hace de acorde a su velocidad usando la función *tanh* de la siguiente manera:

$$\mathrm{X}_i^d(t+1) = \begin{cases} \mathrm{x}_i^d(t) & \text{if } rand_i \leq \left|tanh(\mathrm{v}_i^d(t+1))\right|, \\ \overline{\mathrm{x}_i^d(t)} & \text{De lo contrario.} \end{cases} \tag{14}$$

Para detalles del algoritmo revisar [12] y [16].

3. Distancia Mahalanobis (MD)

La MD es una distancia generalizada que es útil para determinar las similitudes entre conjuntos de datos desconocidos y conocidos. Mide las distancias en espacios multidimensionales; teniendo en cuenta las correlaciones entre cualquier variable o característica que pueda existir [18]. La MD es una medida de distancia que se ha utilizado en aplicaciones como la detección de anomalías, el reconocimiento de patrones y el control de procesos. Además, no es sensible a las diferentes escalas de los parámetros monitoreados, ya que los valores de MD se calculan utilizando parámetros normalizados. La distancia Mahalanobis de un conjunto de datos se calcula de la siguiente manera:

$$MD = \frac{1}{k} Z_{ij} C^{-1} Z_{ij}^T, \tag{15}$$

donde Z_i es el vector normalizado obtenido al normalizar los valores de X_i donde $i = 1, 2, ..., k$:

$$Z_{ij} = \frac{X_{ij} - \overline{X_i}}{S_i}, \quad i = 1, 2, ..., k; \quad j = 1, 2, ..., n \tag{16}$$

$$\overline{X}_i = \frac{1}{n} \sum_{j=1}^{n} X_{ij}, \quad S_i = \sqrt{\frac{\sum_{j=1}^{n} (X_{ij} - \overline{X_i})^2}{(n-1)}} \tag{17}$$

donde X_{ij} es el valor correspondiente a la i-ésima variable en la j-ésima observación y C es la matriz de correlación [14]:

$$C = \frac{1}{(n-1)} \sum_{j=1}^{n} Z_j Z_j^T. \tag{18}$$

4. Metodología para la detección de variables

Para llevar a cabo la selección de variables se implementa una metaheurística binaria con el fin de minimizar la siguiente función de aptitud:

$$\min\ f(\mathbf{x}) = \beta f_1(\mathbf{x}) + (1-\beta)\frac{p_{\text{seleccionadas}}}{p}, \tag{19}$$

sujeto a:

$$\sum_{i=1}^{p} x_i \leq p, \quad \sum_{i=1}^{p} x_i = p_{\text{seleccionadas}}, \quad f_1(\mathbf{x}) \leq f_1^{\max},$$

donde $f_1(\mathbf{x})$ es una función que mide el error en la clasificación binaria de los datos, mientras $p_{\text{seleccionadas}}$ es el subconjunto de variables seleccionado. Siguiendo el enfoque de la metodología llamada sistema Mahalanobis-Taguchi (MTS) [6]. La distancia Mahalanobis es utilizada con el fin de clasificar los datos en dos clases, conocidas como datos saludables y no saludables. En este sentido $f_1(\mathbf{x})$ se calcula de la siguiente manera:

$$f_1(\mathbf{x}) = w_1\frac{n_1^e}{n_1} + w_2\frac{n_2^e}{n_2}, \tag{20}$$

donde n_1^e es el número de observaciones saludables clasificadas como no saludables, n_2^e es el número de las no saludables clasificadas como saludables, n_1 y n_2 es el total de observaciones saludables y no saludables respectivamente. Mientras que w_1 y w_2 son la contribución a cada tipo de clasificación errónea. Para obtener n_1^e y n_2^e se proceden a calcular las distancias mahalanobis de los datos saludables y no saludables y se calcula lo siguiente:

$$n_1^e = \sum_{j=1}^{n_1} MD_j^1 \quad \text{s.t} \quad MD_1^2 \leq MD_j^1, \tag{21}$$

$$n_2^e = \sum_{j=1}^{n_2} MD_j^2 \quad \text{s.t} \quad MD_{n_1}^1 \geq MD_j^2. \tag{22}$$

Por último, para llevar a cabo la selección óptima de variables se usa la metaheurística binaria de la siguiente manera:

Sea $\mathbf{x} = (x_1, x_2, ..., x_p)^t$ un vector de dimensión p donde

$$x_i = \begin{cases} 0 \text{ si la variable } i \text{ no es seleccionada,} \\ 1 \text{ si la variable } i \text{ es seleccionada.} \end{cases} \tag{23}$$

De esta manera la metaheurística binaria retiene solo el subconjunto de variables ($p_{\text{seleccionadas}}$) que minimicen la función de aptitud 19. En este enfoque se comparará el desempeño de las metaheurísticas binarias BPSO y GSA.

5. Aplicación industrial: Soldadura en mecanismo para asiento de coche

A continuación se presenta la metodología antes descrita usando BPSO y BGSA con el fin de comparar el desempeño de cada metaheurística. Dicha

metodología se aplica en un proceso de soldadura de discos de soporte para el mecanismo de los asientos en el automóvil. La figura 1 muestra algunos de los componentes a los cuales el disco es unido mediante soldadura láser. El proceso de soldadura implica 18 variables dimensionales relevantes. Butwell y Lapwell son dos modos de soldadura en el proceso y las características se miden en diferentes cortes realizados en una unión de material, las variables dimensionales correspondientes a diferentes cortes en las uniones soldadas se muestran en la Figura 2 y se enlistan en la Tabla 1. Para fines experimentales, se recolectaron un total de 94 observaciones, de las cuales 63 corresponden a medidas correspondientes a piezas que satisfacen los criterios de calidad (operación normal) y las 31 restantes no las cumplieron (operación defectuosa).

Fig. 1. Soporte de disco y asiento de automóvil.

5.1. Resultados experimentales

Para el cálculo de la función de clasificación errónea (Ec. 20) se consideran tres configuraciones, caso 1: $w_1 = w_2 = 0{,}5$, caso 2: $w_1 = 0{,}95$, $w_2 = 0{,}05$ y caso 3: $w_1 = 0{,}05$, $w_2 = 0{,}95$. β se fija en 0.9 en concordancia con Vignolo [20], quien recomienda fijar este valor entre 0.7 y 0.9. Para las metaheurísticas BPSO y GSA, el número de individuos que conforman la población es de 100 y el número máximo de iteraciones es de 200. En la Tabla 2 se muestra que el valor alcanzado para la función de aptitud (Ec. 19) por BPSO y BGSA es el mismo, de igual manera en este caso de estudio las variables detectadas por ambos enfoques coinciden. En las Figuras 3-5 se muestra que BGSA converge mas rápido que BPSO en los tres casos de experimentación planteados. Con el fin de probar el efecto de las variables seleccionadas, se entrenan otros algoritmos de aprendizaje automático supervisado y uno no supervisado ampliamente usados en la detección de fallas. En este experimento se entrenan dichos enfoques con las 63 observaciones en operación normal y las 31 observaciones en operación defectuosa con el fin de obtener la exactitud en la clasificación de este conjunto de datos por parte de los algoritmos, dicho porcentaje de exactitud es obtenido

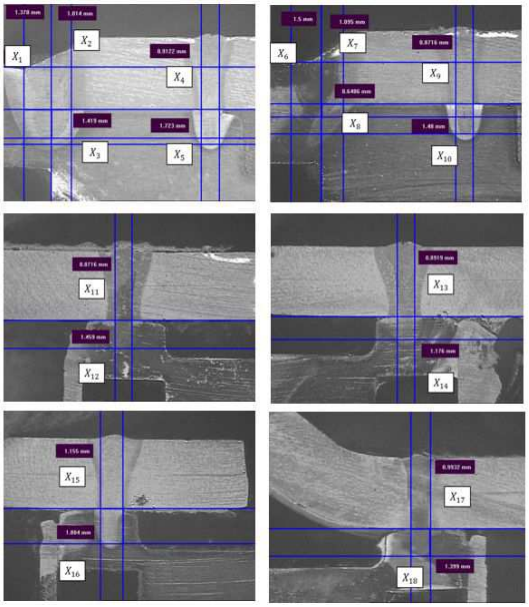

Fig. 2. Variables dimensionales en cortes del disco.

usando validación cruzada de 10 subconjuntos. Primero estos se entrenan considerando la totalidad de las variables (18 variables), y posteriormente se entrenan usando solo el subconjunto de variables seleccionadas. Podemos ver en la Tabla 3 que de 16 enfoques de aprendizaje automático supervisado cuatro de ellos obtuvieron el mismo porcentaje de exactitud para ambos experimentos. Otros cuatro obtuvieron un porcentaje mayor usando la totalidad de las variables, mientras que 8 de los algoritmos incluso aumentaron su exactitud al ser entrenados solo con el subconjunto de variables detectados por la metodología.

Finalmente se prueba el efecto de la selección de variables usando el algoritmo de aprendizaje no supervisado k-medias el cual es uno de los métodos más usados para agrupamiento o clustering. Sus posibles aplicaciones son agrupamiento por similitud, predicción no lineal, aproximaciones de distribuciones multivariadas y pruebas no paramétricas de independencia entre varias variables, además de ser relativamente sencillo de implementar y computacionalmente económico [10]. Para detalles referirse a [1]. Para este experimento se usa la base de datos con las 63 observaciones en operación normal y las 31 observaciones en operación defectuosa formando una base de datos de 94 observaciones con 18 variables para el primer caso y otra usando solo las variables seleccionadas, es decir de 94 observaciones y 7 variables. Posteriormente se elige k=2, es decir k-medias formará dos conglomerados de manera no supervisada. Posteriormente se evaluará el contenido de los mismos, es decir como agrupó las observaciones normales y las defectuosas ya que de antemano se conoce cuáles de estas observaciones pertenecen a una u otra clase. Para evaluar este algoritmo se usaron diversas

Tabla 1. Descripción de las variables

	Descripción
x_1	Ancho 1 de soldadura Butwell corte 1, en unión (disco A)
x_2	Ancho 2 de soldadura Butwell corte 1, en unión (disco A)
x_3	Penetración de soldadura, Butwell corte 1, en unión (disco A)
x_4	Ancho 1 de soldadura Butwell corte 2, en unión (disco A)
x_5	Ancho 2 de soldadura Butwell corte 2, en unión (disco A)
x_6	Penetración de soldadura, Butwell corte 2, en unión (disco A)
x_7	Ancho de soldadura Lapwell corte 1, en unión (disco A)
x_8	Penetración de soldadura, Lapwell corte 1, en unión (disco A)
x_9	Ancho de soldadura Lapwell corte 2, en unión (disco A)
x_{10}	Penetración de soldadura, Lapwell corte 2, en unión (disco A)
x_{11}	Ancho de soldadura Lapwell corte 1, en unión (disco B)
x_{12}	Penetración de soldadura, Lapwell corte 1, en unión (disco B)
x_{13}	Ancho de soldadura Lapwell corte 2, en unión (disco B)
x_{14}	Penetración de soldadura, Lapwell corte 2, en unión (disco B)
x_{15}	Ancho de soldadura Lapwell corte 3, en unión (disco B)
x_{16}	Penetración de soldadura, Lapwell corte 3, en unión (disco B)
x_{17}	Ancho de soldadura Lapwell corte 4, en unión (disco B)
x_{18}	Penetración de soldadura, Lapwell corte 4, en unión (disco B)

Tabla 2. Valores óptimos y variables seleccionadas para las diferentes configuraciones.

	Metaheurística	Valores óptimos	Variables seleccionadas
Caso 1	PSO	0.0389	$x_2, x_3, x_6, x_8, x_{10}, x_{11}, x_{13}$
	GSA	0.0389	$x_2, x_3, x_6, x_8, x_{10}, x_{11}, x_{13}$
Caso 2	PSO	0.0389	$x_2, x_3, x_6, x_8, x_{10}, x_{11}, x_{13}$
	GSA	0.0389	$x_2, x_3, x_6, x_8, x_{10}, x_{11}, x_{13}$
Caso 3	PSO	0.0389	$x_2, x_3, x_6, x_8, x_{10}, x_{11}, x_{13}$
	GSA	0.0389	$x_2, x_3, x_6, x_8, x_{10}, x_{11}, x_{13}$

métricas como la exactitud (Ac), la precisión (Pr), exhaustividad (Re) y la conocida como F-score [2], las cuales se calculan de la siguiente manera.

$$\text{Ac} = \frac{\text{TP+TN}}{\text{TP+FN+FP+TN}}, \tag{24}$$

$$\text{Pr} = \frac{\text{TP}}{\text{TP+FP}}, \tag{25}$$

$$\text{Re} = \frac{\text{TP}}{\text{TP+FN}}, \tag{26}$$

$$\text{F-score} = \frac{2 \times \text{Re} \times \text{Pr}}{\text{Re+Pr}}, \tag{27}$$

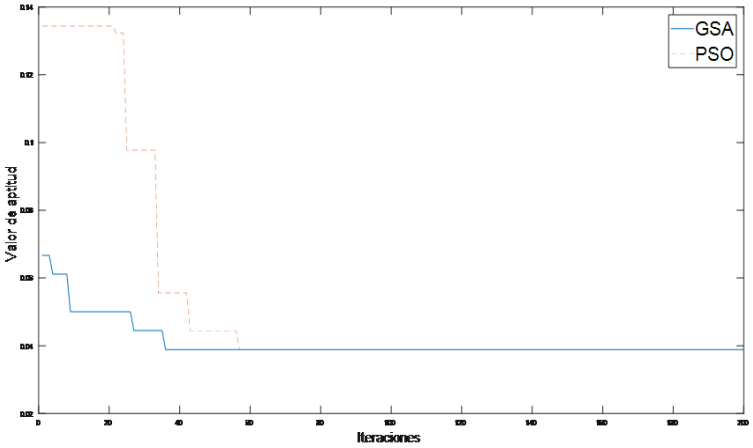

Fig. 3. Convergencia en el caso 1: $w_1 = w_2 = 0{,}5$

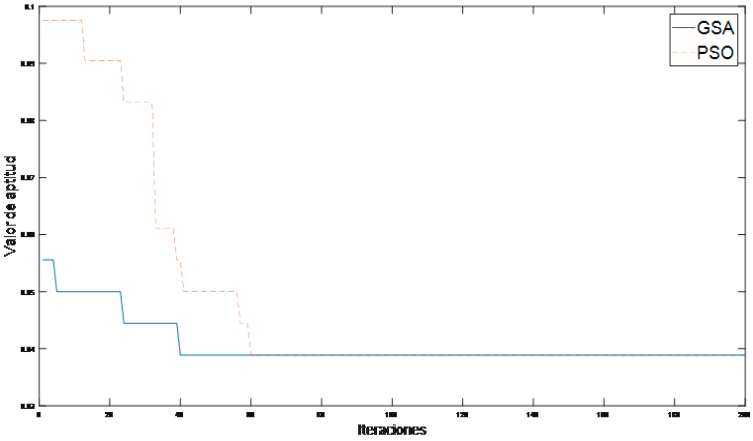

Fig. 4. Convergencia en el caso 2: $w_1 = 0{,}95, w_2 = 0{,}05$

donde TP y TN se refieren a los verdaderos positivos y negativos respectivamente, mientras FN y FP son los falsos negativos y falsos positivos. Los resultados de ambos casos se presentan en la Tabla 4.

Como se observa, los valores de las métricas alcanzados por k-medias son bajos para implementar dicho algoritmo para la detección de fallas. Sin embargo cabe señalar que se usó la versión básica del algoritmo, y es importante que el lector esté consciente de que existen diversos enfoques y estrategias para mejorar el rendimiento de k-medias. En este estudio solo se probó el efecto de implementar el algoritmo con el total de las variables y con las variables seleccionadas por el enfoque presentado. Siendo que este último logro valores ligeramente superiores.

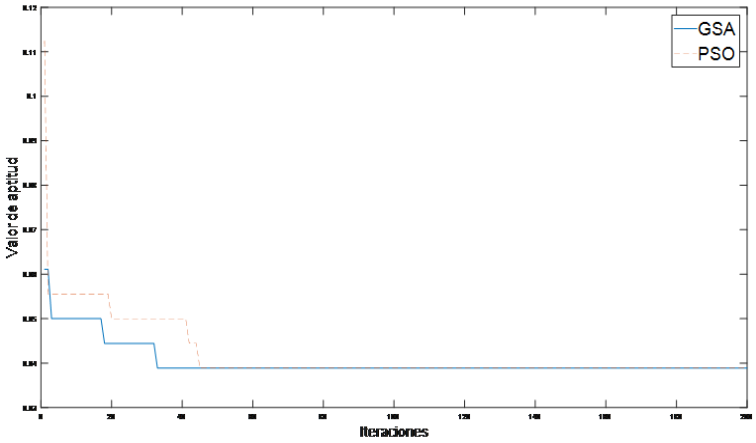

Fig. 5. Convergencia en el caso 3: $w_1 = 0{,}05, w_2 = 0{,}95$

Tabla 3. Exactitud % de otros algoritmos de aprendizaje de máquina supervisado

Algoritmo	Todas las variables	Variables seleccionadas
Complex tree	80.9	**85.1**
Medium tree	80.9	**85.1**
Simple tree	72.3	**81.9**
Logistic regression	64.9	**71.3**
Linear SVM	68.1	**74.5**
Quadratic SVM	84.0	**86.2**
Cubic SVM	79.8	**80.9**
Fine Gaussian SVM	69.1	69.1
Medium Gaussian SVM	**94.7**	90.4
Coarse Gaussian SVM	68.1	68.1
Fine KNN	**80.9**	79.8
Medium KNN	67.0	**68.1**
Coarse KNN	67.0	67.0
Cosine KNN	**71.3**	68.1
Cubic KNN	**68.1**	67.0
Weighted KNN	69.1	69.1

6. Conclusiones y trabajo futuro

En este artículo se presentó una herramienta basada en metaheurísticas binarias y la distancia Mahalanobis para la selección óptima de características aplicada a un proceso de soldadura de discos de soporte para el mecanismo de los asientos en el automóvil. La metodología combina las ventajas discriminantes de la distancia Mahalanobis y las capacidades de PSO y GSA para encontrar la solución óptima de un problema de optimización con variables binarias. Esto

Tabla 4. Evaluación del algoritmo k-medias

Métrica	Todas las variables	Variables seleccionadas
Exactitud	0.5638	**0.5745**
Precisión	**0.7037**	0.7018
Exhaustividad	0.6032	**0.6349**
F-score	0.6496	**0.6667**

con el fin de seleccionar las variables que impactan en los defectos del producto. Como se demostró en la experimentación, ambas metaheurísticas alcanzaron el mismo valor en la función de la aptitud. De igual manera ambas detectaron el mismo subconjunto de variables, lo que permite a las personas encargadas del proceso tomar acciones para controlar estas variables. Por otro lado GSA convergió mas rápido que PSO en los tres casos de experimentación. Además, el efecto de este subconjunto de características seleccionadas se ha implementado en diversos algoritmos de aprendizaje automático supervisado y no supervisado. Donde en ocho de los supervisados y el no supervisado obtuvieron valores mayores de exactitud al ser entrenados solo con el subconjunto de variables seleccionadas por la metodología. Como trabajo futuro se esta interesado en la incorporación de otras metaurísticas híbridas para resolver el problema de minimización presentado en este trabajo. Así como implementar enfoques para mejorar la capacidad de clasificación de la distancia Mahalanobis.

Referencias

1. Abualigah, L.M., Khader, A.T., Hanandeh, E.S.: Hybrid clustering analysis using improved krill herd algorithm. Applied Intelligence 48(11), 4047–4071 (Nov 2018)
2. Abualigah, L.M., Khader, A.T., Hanandeh, E.S.: A new feature selection method to improve the document clustering using particle swarm optimization algorithm. Journal of Computational Science 25, 456–466 (2018)
3. Chandrashekar, G., Sahin, F.: A survey on feature selection methods. Computers and Electrical Engineering 40, 16–28 (2014)
4. Chiang, L.H., Russell, E.L., Braatz, R.D.: Fault Detection and Diagnosis in Industrial Systems. Springer-Verlag London Ltd. (2001)
5. Das, S., Abraham, A., Konar, A.: Particle swarm optimization and differential evolution algorithms. technical analysis, applications and hybridization perspectives. Studies in Computational Intelligence (2008)
6. Ghasemi, E., Aaghaie, A., Cudney, E.A.: Mahalanobis taguchi system: a review. International Journal of Quality & Reliability Management Vol. 32 (2015)
7. Haixiang, G., Yijing, L., Yanan, L., Xiao, L., Jinling, L.: BPSO-adaboost-KNN ensemble learning algorithm for multi-class imbalanced data classification. Engineering Applications of Artificial Intelligence (2016)
8. Kennedy, J., Eberhart, R.: Particle swarm optimization. International conference on neural networks, IEEE (1995)
9. Luke, S.: Essentials of Metaheuristics. Lulu, second edn. (2013), available for free at http://cs.gmu.edu/∼sean/book/metaheuristics/

10. MacQueen, J.: Some methods for classification and analysis of multivariate observations. In: Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability, Volume 1: Statistics. pp. 281–297. University of California Press, Berkeley, Calif. (1967)
11. Mirjalili, S., Hashim, S.Z.M.: A new hybrid PSOGSA algorithm for function optimization. In: 2010 International Conference on Computer and Information Application. pp. 374–377 (Dec 2010)
12. Mirjalili, S., Wang, G.G., Coelho, L.d.S.: Binary optimization using hybrid particle swarm optimization and gravitational search algorithm. Neural Computing and Applications 25(6), 1423–1435 (Nov 2014), https://doi.org/10.1007/s00521-014-1629-6
13. Osman, I., Laporte, G.: Metaheuristics: A bibliography. Ann Oper Res (1996)
14. Patil, N., Das, D., Pecht, M.: Anomaly detection for igbts using mahalanobis distance. Microelectronics Reliability 55(7), 1054–1059 (2015), http://www.sciencedirect.com/science/article/pii/S0026271415000888
15. Rashedi, E., Nezamabadi-pour, H., Saryazdi, S.: GSA: A Gravitational Search Algorithm. Information Sciences 179(13), 2232–2248 (2009), special Section on High Order Fuzzy Sets
16. Rashedi, E., Nezamabadi-pour, H., Saryazdi, S.: BGSA: binary gravitational search algorithm. Natural Computing 9(3), 727–745 (Sep 2010)
17. Reséndiz-Flores, E.O., Rull-Flores, C.A.: Mahalanobis-Taguchi system applied to variable selection in automotive pedals components using gompertz binary particle swarm optimization. Expert Systems with Applications (2013)
18. Rizal, M., Ghani, J., Nuawi, M., Haron, C.: Cutting tool wear classification and detection using multi-sensor signals and Mahalanobis-Taguchi system. Wear 376-377, 1759–1765 (2017), 21st International Conference on Wear of Materials
19. Vapnik, V.: The Nature of Statistical Learning Theory. Springer (2000)
20. Vignolo, L.D., Milone, D.H., Scharcanski, J.: Feature selection for face recognition based on multi-objective evolutionary wrappers. Expert Systems with Applications 40(13), 5077–5084 (2013)