

# Detección de fallas en motores de turbinas de gas empleando machine learning

Laura Martir Beltrán C411  
Adrián Hernández Castellanos C412  
Yesenia Valdés Rodríguez C411  
Ariadna Velázquez Rey C411  
Lía Stephanie López Rosales C412  
Olivia Ibañez Mustelier C411

4 de enero de 2026

## Resumen

La detección temprana de fallos en turbinas de gas es un desafío crítico para la operación segura y eficiente de instalaciones energéticas. Este informe busca resolver este problema entrenando diversos modelos de machine learning y comparando su desempeño a la hora de resolver el problema planteado. Debido a la limitada disponibilidad de datos y metainformación en el proyecto original, se ha decidido utilizar el Gas Turbine Engine Fault Detection Dataset (Kaggle) como fuente principal de datos, aplicando técnicas de aumento de información para robustecer el entrenamiento de modelos de aprendizaje automático.

# Índice

<b>1. Dataset</b>	<b>3</b>
1.1. Estructura . . . . .	3
1.2. Justificación del Conjunto de Datos . . . . .	3
1.3. Análisis Exploratorio de Datos (EDA) . . . . .	3
1.3.1. Medidas de Tendencia Central . . . . .	4
1.3.2. Medidas de Dispersión . . . . .	5
1.3.3. Distribución de las variables independientes . . . . .	5
1.3.4. Distribución de la Variable Objetivo . . . . .	5
1.3.5. Análisis de Correlación y Dependencia . . . . .	7
1.4. Generación de Datos Sintéticos y Aumentación . . . . .	7
1.4.1. Metodología de Simulación y Generación de Datos . . . . .	8
1.4.2. Parámetros de la Simulación . . . . .	9
1.4.3. Modelado de Modos de Falla y Dinámica de Degradeación	10
1.5. Análisis Exploratorio del Conjunto de Datos Sintético . . . . .	11
1.5.1. Distribución de Clases y Desbalance . . . . .	11
<b>2. Tareas de Modelado</b>	<b>12</b>
2.1. Selección de Variables y Preprocesamiento . . . . .	13
2.2. Clasificación Binaria: Detección de Fallas Operacionales . . . . .	14
2.2.1. Modelo de Regresión Logística ( <i>Baseline</i> ) . . . . .	16
2.2.2. Máquinas de Soporte Vectorial (SVM) . . . . .	20
2.2.3. Modelo XGBoost ( <i>Extreme Gradient Boosting</i> ) . . . . .	23
2.2.4. Ensamble mediante Clasificador de Votación ( <i>Voting Classifier</i> ) . . . . .	27
2.2.5. Modelo de Bosques Aleatorios ( <i>Random Forest</i> ) . . . . .	27
2.2.6. Ensamble Híbrido: <i>Random Forest</i> y XGBoost . . . . .	29
2.2.7. Modelo de Red Neuronal . . . . .	30
2.2.8. Ensamble Híbrido: SVM y Regresión Logística . . . . .	35
2.3. Clasificación Multiclas: Diagnóstico del Modo de Falla . . . . .	36
2.3.1. Regresión Logística ( <i>Baseline Multiclas</i> ) . . . . .	38
2.3.2. Clasificador de $k$ Vecinos Más Cercanos (KNN) . . . . .	40
2.3.3. Máquina de Vectores de Soporte (SVM) . . . . .	45
<b>3. Conclusiones</b>	<b>47</b>

## 1. Dataset

Se empleó el [Gas Turbine Engine Fault Detection Dataset](#).

### 1.1. Estructura

El conjunto de datos comprende un total de  $N = 1386$  muestras provenientes de un sistema simulado de turbinas de gas. Las variables registradas se describen a continuación:

**Temperature (°C):** Temperatura del aire en la admisión.

**RPM:** Velocidad angular de rotación del eje principal.

**Torque (Nm):** Par motor o fuerza torsional aplicada al eje.

**Vibrations (mm/s):** Velocidad de vibración mecánica estructural.

**Power Output (MW):** Potencia eléctrica neta generada.

**Fuel Flow Rate (kg/s):** Flujo máscico del combustible.

**Air Pressure (kPa):** Presión manométrica del aire (entrada/salida).

**Exhaust Gas Temp. (°C):** Temperatura de los gases de escape.

**Oil Temperature (°C):** Temperatura del fluido lubricante.

**Fault:** Variable binaria de estado (0: Normal, 1: Falla).

### 1.2. Justificación del Conjunto de Datos

Se seleccionó el dataset de Kaggle debido a su representatividad operacional frente a la escasez de datos del entorno original ( $N < 30$ ). Debido a que la falta de etiquetas y metadatos en el corpus inicial imposibilitaba el aprendizaje supervisado con generalización estadística, se optó por esta fuente externa como base para procesos de síntesis y aumentación de datos.

### 1.3. Análisis Exploratorio de Datos (EDA)

Se procedió a realizar una evaluación estadística y visual del conjunto de datos original con el objetivo de caracterizar las distribuciones de las variables operacionales y detectar posibles anomalías. Los resultados de este análisis se presentan y discuten a continuación.

### 1.3.1. Medidas de Tendencia Central

Se observa una estabilidad operativa en los sensores principales (ver Fig. 1). Para la mayoría de las variables, la proximidad numérica entre la media aritmética y la mediana sugiere distribuciones con una simetría sustancial y una baja incidencia de valores atípicos extremos. Este equilibrio estadístico indica que el sistema se mantiene dentro de regímenes nominales controlados, comportamiento consistente con un entorno industrial monitoreado. Los estadísticos descriptivos se resumen en la Tabla 1.

Cuadro 1: Medidas de tendencia central de las variables operacionales.

Variable	Media	Mediana	Moda
Temperature (°C)	901.62	901.96	737.94
RPM	15022.59	15016.74	13490.24
Torque (Nm)	3494.76	3490.64	2901.77
Vibrations (mm/s)	1.98	1.99	0.41
Power Output (MW)	99.50	99.38	71.00
Fuel Flow Rate (kg/s)	2.51	2.50	1.35
Air Pressure (kPa)	150.32	150.46	93.96
Exhaust Gas Temp. (°C)	498.87	498.87	382.33
Oil Temperature (°C)	120.06	119.97	86.24

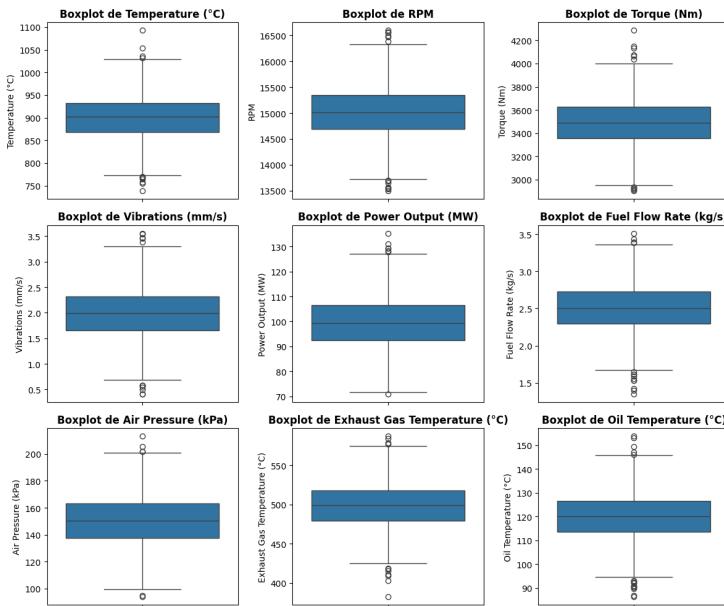


Figura 1: Distribución mediante diagramas de caja y bigotes de las variables independientes.

### 1.3.2. Medidas de Dispersión

Se observa una variabilidad significativa en la mayoría de los parámetros operativos, lo que indica que las condiciones del sistema no son estáticas, sino que fluctúan en rangos amplios en función de la carga y la demanda. Los estadísticos de dispersión se detallan en la Tabla 2.

Cuadro 2: Medidas de dispersión de las variables del sistema.

Variable	$\sigma^2$	$\sigma$	Min	Max	CV (%)
Temperature (°C)	2438.53	49.38	737.94	1092.64	161.04
RPM	240147.68	490.05	13490.24	16596.55	1598.14
Torque (Nm)	41545.92	203.83	2901.77	4285.25	664.72
Vibrations (mm/s)	0.24	0.49	0.41	3.56	1.61
Power Output (MW)	106.29	10.31	71.00	135.29	33.62
Fuel Flow Rate (kg/s)	0.10	0.32	1.35	3.51	1.03
Air Pressure (kPa)	377.53	19.43	93.96	213.16	63.37
Exhaust Gas Temp. (°C)	829.09	28.79	382.33	587.42	93.90
Oil Temperature (°C)	100.10	10.01	86.24	153.78	32.63

### 1.3.3. Distribución de las variables independientes

Se aplicó la prueba de normalidad de Shapiro-Wilk a las variables independientes para contrastar estadísticamente la hipótesis de una distribución gaussiana, sugerida por las observaciones preliminares. Los resultados, sintetizados en la Tabla 3, junto con las distribuciones visualizadas en la Fig. 2, confirman que en ningún caso se rechaza la hipótesis nula ( $p > 0,05$ ). Por consiguiente, se asume que todas las variables siguen una distribución normal.

Cuadro 3: Resultados de la prueba de normalidad Shapiro-Wilk.

Variable	Estadístico W	p-value	Distribución
Temperature (°C)	0.9991	0.7402	Normal
RPM	0.9992	0.8260	Normal
Torque (Nm)	0.9991	0.7063	Normal
Vibrations (mm/s)	0.9990	0.6231	Normal
Power Output (MW)	0.9987	0.4328	Normal
Fuel Flow Rate (kg/s)	0.9984	0.2010	Normal
Air Pressure (kPa)	0.9983	0.1889	Normal
Exhaust Gas Temp. (°C)	0.9989	0.5281	Normal
Oil Temperature (°C)	0.9985	0.2970	Normal

### 1.3.4. Distribución de la Variable Objetivo

El análisis de la variable *Fault* revela un desbalance de clases significativo, como se ilustra en la Fig. 3. Las observaciones correspondientes al estado operativo normal (clase 0) representan el 69,34 %, mientras que las instancias de

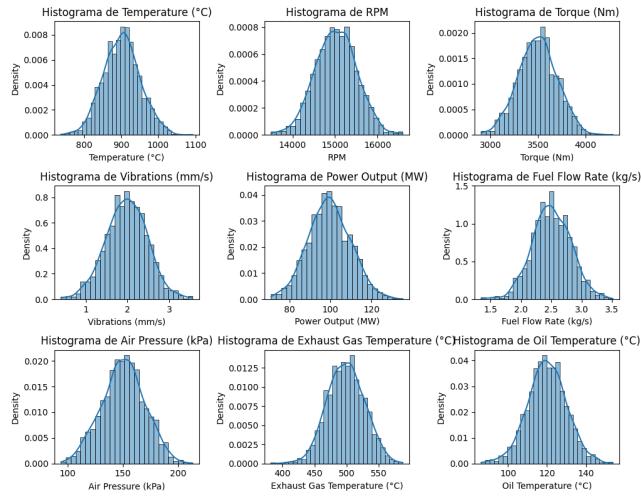


Figura 2: Distribución de densidad de las variables independientes.

falla (clase 1) constituyen únicamente el 30,66 %. Esta asimetría en la distribución de los datos introduce un sesgo hacia la clase mayoritaria, lo que podría comprometer la capacidad predictiva del modelo en la detección de fallas.

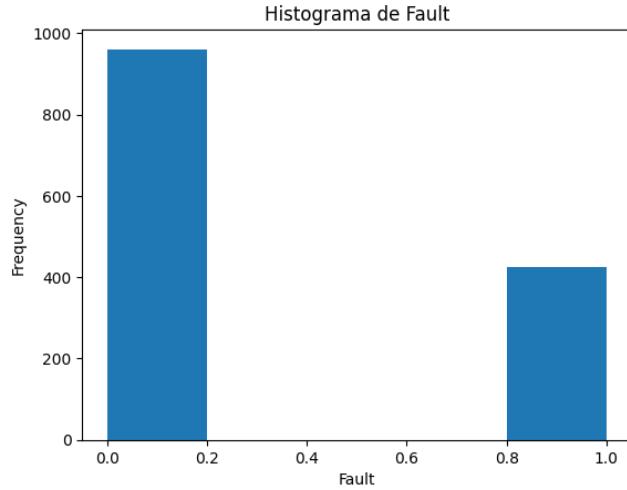


Figura 3: Distribución proporcional de la variable objetivo (Fault).

### 1.3.5. Análisis de Correlación y Dependencia

Dada la naturaleza gaussiana de las variables, se computó la matriz de correlación lineal empleando el coeficiente de Pearson ( $r$ ). Como se ilustra en la Fig. 4, el análisis revela una ausencia de correlación significativa entre los pares de variables, con coeficientes cercanos a cero en la totalidad del espacio de características.

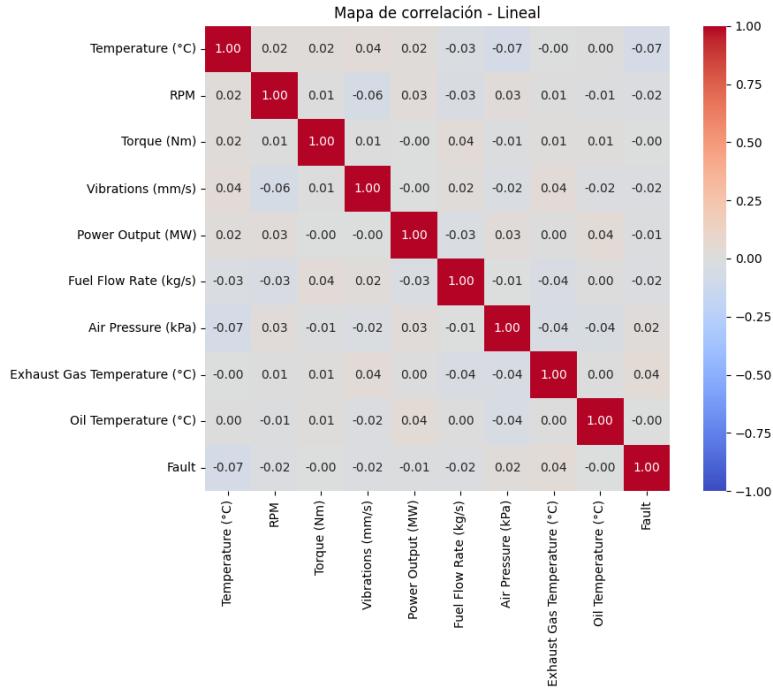


Figura 4: Matriz de correlación de Pearson para las variables operacionales.

Complementariamente, se generaron diagramas de dispersión (ver Fig. 5) con el fin de identificar posibles dependencias funcionales de carácter no lineal. La inspección visual de las nubes de puntos corrobora la inexistencia de patrones o tendencias estructurales, reafirmando la independencia estocástica entre los parámetros del sistema evaluado.

### 1.4. Generación de Datos Sintéticos y Aumentación

El análisis exploratorio previo demuestra que las variables del sistema presentan una independencia estocástica significativa, caracterizada por la ausencia de correlaciones lineales y la inexistencia de dependencias funcionales no lineales detectables en los diagramas de dispersión. Este desacoplamiento entre los parámetros operativos indica que la varianza de cada sensor no puede ser ex-

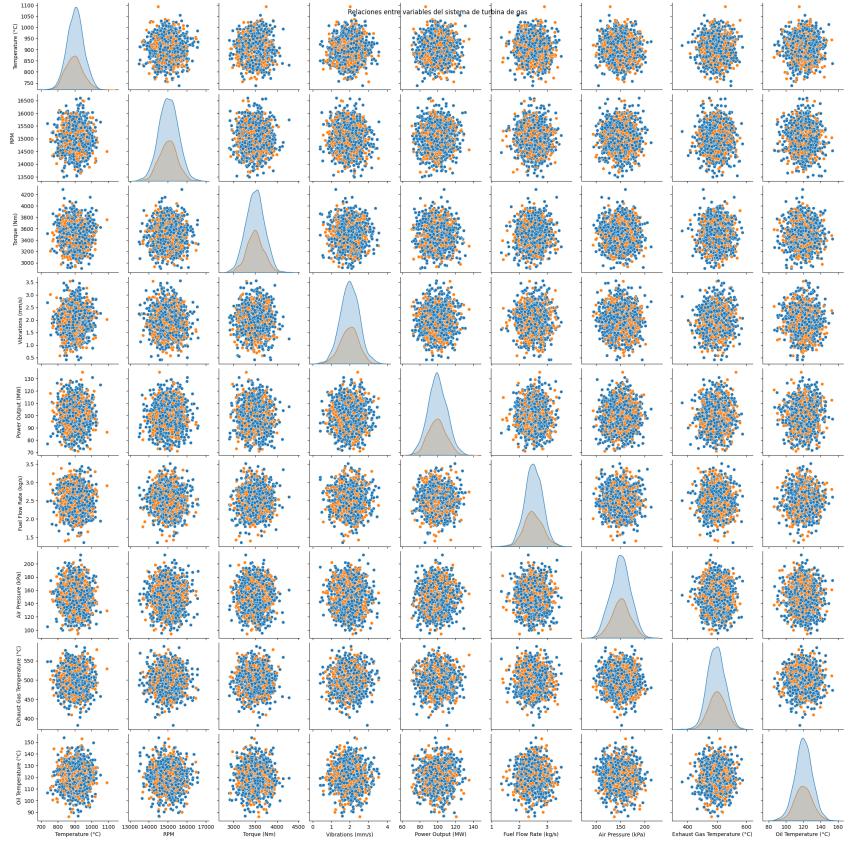


Figura 5: Matriz de diagramas de dispersión de las variables del conjunto de datos.

plicada mediante el comportamiento de los demás componentes del vector de características.

Ante esta autonomía de las variables y con el objetivo de robustecer el entrenamiento de los modelos predictivos, se determinó la necesidad de realizar una síntesis de datos. Este proceso de simulación, fundamentado en las distribuciones empíricas de las variables originales, permite enriquecer el conjunto de datos y mitigar las limitaciones derivadas de la escasez de muestras y el desbalance de clases previamente identificado.

#### 1.4.1. Metodología de Simulación y Generación de Datos

El proceso de simulación se diseñó para modelar el ciclo de vida operativo de  $N$  turbinas, cada una con una longitud de ciclo  $L$  definida. El modelo propuesto simula la transición de un estado de operación normal hacia un estado de fallo crítico mediante un proceso de degradación estocástica. A cada unidad se le

asigna de manera aleatoria una tipología de falla específica, la cual condiciona la cinética de deterioro de sus componentes hasta alcanzar el umbral de ruptura.

Para preservar la fidelidad estadística del conjunto de datos original, se empleó la Estimación de Densidad por Kernel (*Kernel Density Estimation*, KDE). Este modelo no paramétrico fue entrenado exclusivamente con las observaciones del estado operativo normal ( $Fault = 0$ ) para caracterizar y replicar la distribución de probabilidad multivariante de las variables originales.

Posteriormente, las muestras asociadas a fases de degradación progresiva o fallo inminente se generaron aplicando perturbaciones controladas sobre los valores base. Estas variaciones se parametrizaron en magnitud y dirección en función de la sensibilidad de los sensores ante el tipo de falla asignado, permitiendo una representación robusta del comportamiento dinámico de los componentes afectados bajo condiciones de estrés mecánico.

#### 1.4.2. Parámetros de la Simulación

El modelo de generación de datos se fundamenta en un conjunto de parámetros que rigen la transición entre los estados de operación nominal y de falla. A continuación, se definen las variables de control empleadas:

$\mathcal{M}_{fault}$ : Conjunto de modos de falla que comprende tres tipologías específicas de degradación sistémica.

$N_{turbanas}$ : Tamaño de la muestra poblacional, fijado en 1000 unidades experimentales para asegurar significancia estadística.

$L_{ciclo}$ : Horizonte temporal máximo de operación, definido en 100 intervalos discretos.

$U_{deterioro}$ : Umbral operativo (iteración 50) que marca el inicio de la transición del estado nominal al estado de degradación progresiva.

$U_{falla}$ : Umbral crítico (30 iteraciones antes del colapso) donde se intensifican las anomalías mecánicas y térmicas.

$K_p$ : Factor de ajuste de potencia (0,00015) que regula la respuesta energética del sistema ante el desgaste acumulado.

$\delta_{tasa}$ : Tasa nominal de degradación (0,05), que rige el decaimiento de las variables operacionales durante el ciclo de vida.

$C_{deg}$ : Vector de coeficientes de degradación específicos para cada variable del sistema, permitiendo un modelado heterogéneo del desgaste.

$\epsilon \sim \mathcal{N}(0, 0,02)$ : Variable aleatoria de ruido blanco gaussiano incorporada para modelar la variabilidad estocástica inherente a la instrumentación.

$ttc$  (*Time to Collapse*): Indicador temporal que representa el tiempo remanente hasta la inoperatividad total de la unidad.

*actual\_ttc*: Contador de la iteración actual o estado temporal de la turbina dentro de su ciclo de vida.

$\eta$  (**Eficiencia**): Parámetro que modela la integridad termodinámica y mecánica global, afectado por fenómenos de degradación y condiciones ambientales.

#### 1.4.3. Modelado de Modos de Falla y Dinámica de Degradoación

La caracterización de las anomalías y las dependencias paramétricas se fundamenta en los principios de operación de turbinas de gas descritos por Soares [1]. A continuación, se detallan los mecanismos simulados para cada modo de falla:

1. **Fallo mecánico en cojinetes ( $\mathcal{M}_{mec}$ )**: Este modo modela el deterioro estructural de los elementos de rodadura. De acuerdo con [1], el incremento de la fricción interna se manifiesta como un aumento progresivo en los niveles de vibración (*Vibrations (mm/s)*) y en la temperatura del aceite lubricante (*Oil Temperature (°C)*). El desbalanceo mecánico consume energía cinética, lo que induce una atenuación en el par motor (*Torque (Nm)*) y en la velocidad angular (*RPM*). El monitoreo de estas variables es crítico para la detección de fallas inminentes en los cojinetes [1, p. 29-30].
2. **Fallo de enfriamiento y presión ( $\mathcal{M}_{pres}$ )**: Simula la pérdida de eficiencia del compresor por fugas o ensuciamiento. La reducción de la presión de aire (*Air Pressure (kPa)*) limita la capacidad de refrigeración del ciclo, forzando un incremento en la temperatura de entrada a la turbina (*Temperature (°C)*) y de los gases de escape (*Exhaust Gas Temperature (°C)*) para sostener la carga operativa [1, p. 28]. Consecuentemente, el flujo de combustible (*Fuel Flow Rate (kg/s)*) aumenta para compensar la caída de rendimiento térmico.
3. **Anomalía en el lazo de control de combustible ( $\mathcal{M}_{ctrl}$ )**: Modela una desregulación en las válvulas de control o sensores de flujo (*Fuel Flow Rate (kg/s)*). Según [1, p. 30], un flujo excesivo no controlado provoca fenómenos de *over-firing*, elevando drásticamente las temperaturas internas (*Temperature (°C)/Exhaust Gas Temperature (°C)*) y comprometiendo la integridad estructural de los álabes por estrés térmico.
4. **Acoplamiento de Potencia y Eficiencia**: La potencia de salida ( $P$ ) se define mediante la relación fundamental:

$$P \propto \eta \times \tau \times \omega \quad (1)$$

Donde  $P$  representa *Power Output (MW)*,  $\eta$  representa la eficiencia global,  $\tau$  el *Torque (Nm)* y  $\omega$  la velocidad angular (*RPM*). El deterioro sistémico reduce  $\eta$ , disminuyendo la potencia neta producida para un flujo de combustible constante [1, p. 31].

**Formulación de la Degradación Estocástica** Para modelar la evolución temporal de las variables afectadas, se emplea una función de crecimiento exponencial ponderada por el estado de desgaste actual, definida como:

$$V_i(t) = C_{deg,i} \cdot e^{(\delta \cdot t_{actual})} + \epsilon \quad (2)$$

Donde  $V_i(t)$  es el valor de la variable  $i$  en el tiempo  $t$ ,  $C_{deg,i}$  es el coeficiente de degradación específico,  $\delta$  representa la tasa de degradación sistémica y  $\epsilon \sim \mathcal{N}(0, \sigma^2)$  es el componente de ruido blanco que captura la incertidumbre de la instrumentación y las fluctuaciones ambientales.

## 1.5. Análisis Exploratorio del Conjunto de Datos Sintético

Tras la ejecución del proceso de simulación, el conjunto de datos se ha expandido mediante la incorporación de dimensiones temporales y categóricas que permiten el análisis de la degradación sistémica. Las nuevas variables integradas se definen de la siguiente manera:

- **Turbine ID:** Identificador unívoco para cada unidad experimental, facilitando el seguimiento de trayectorias individuales.
- **TTC (*Time To Collapse*):** Índice temporal que registra la evolución cronológica del ciclo de vida de cada unidad.
- **Fault Mode:** Variable categórica que codifica la tipología de fallo: 0 (Normal), 1 ( $\mathcal{M}_{mec}$ ), 2 ( $\mathcal{M}_{pres}$ ) y 3 ( $\mathcal{M}_{ctrl}$ ).

A diferencia del conjunto de datos original, las distribuciones de las variables físicas presentan ahora diferencias notables (ver Fig. 6), consecuencia de la inclusión de fases de degradación inducida. Pruebas de normalidad de Shapiro-Wilk confirmaron que las variables han dejado de seguir una distribución gaussiana. Por consiguiente, el análisis de interdependencia se realizó mediante el **coeficiente de correlación de Spearman** en lugar de Pearson, permitiendo capturar relaciones monótonas no lineales.

Asimismo, la Fig. 9 y Fig. 7 ilustra la evolución de las variables en el espacio de características. Se observa que, conforme el ciclo de vida se aproxima al umbral crítico, emergen patrones de correlación no lineal y agrupamientos (*clusters*) condicionados por el modo de falla asignado. La matriz de correlación resultante (Fig. 8) refleja dependencias estadísticas entre los sensores y la variable temporal, validando que la simulación captura la dinámica de deterioro mecánico y térmico.

### 1.5.1. Distribución de Clases y Desbalance

Las proporciones de la variable objetivo *Fault* se mantienen consistentes con la distribución original, preservando el desbalance operativo del sistema. Por su parte, la distribución de los modos de falla específicos muestra un equilibrio notable entre las tres categorías de error simuladas, como se detalla en la Tabla 4.

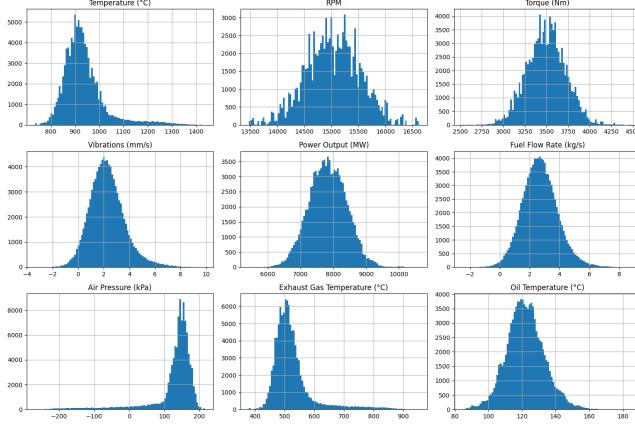


Figura 6: Distribución de densidad de las variables operacionales tras la simulación.

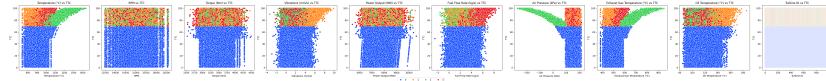


Figura 7: Degradación de los componentes de las turbinas en el tiempo

## 2. Tareas de Modelado

De acuerdo con los requerimientos técnicos y los objetivos del proyecto, la resolución del problema se ha estructurado en dos fases jerárquicas de modelado predictivo:

- 1. Detección Binaria de Fallos (Objetivo Principal):** Desarrollo de un modelo de clasificación supervisada orientado a la identificación prematura de anomalías. El sistema debe ser capaz de discriminar entre el estado de operación nominal ( $Fault = 0$ ) y el estado de falla inminente ( $Fault = 1$ ), proporcionando una alerta temprana sobre la integridad de la turbina.
- 2. Diagnóstico y Clasificación de Modos de Falla (Objetivo Secundario):** Para aquellas instancias clasificadas positivamente en la fase anterior, se implementará un modelo de clasificación multiclas. Esta tarea consiste en diagnosticar la etiología específica de la falla, categorizándola dentro de los modos definidos: falla mecánica en cojinetes ( $\mathcal{M}_{mec}$ ), falla de enfriamiento y presión ( $\mathcal{M}_{pres}$ ), o anomalía en el lazo de control de combustible ( $\mathcal{M}_{ctrl}$ ).

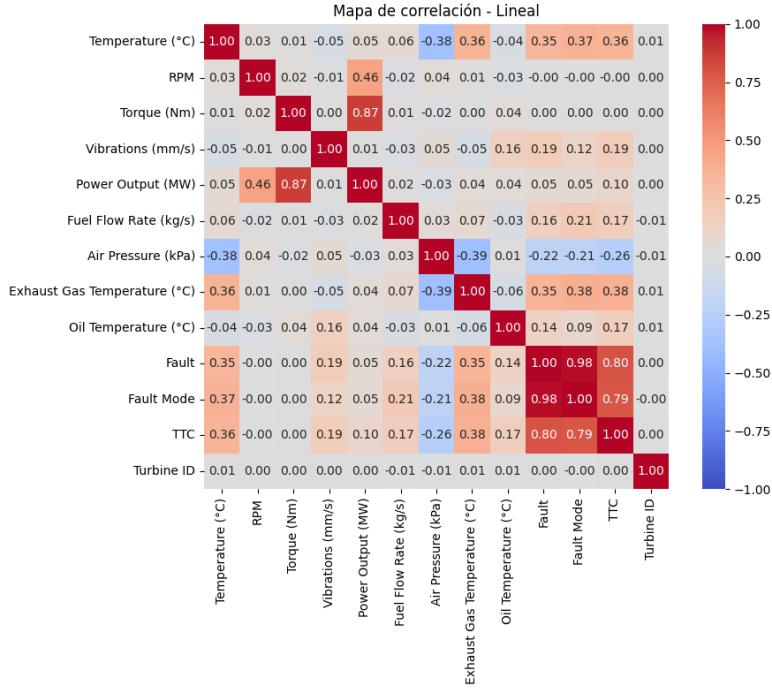


Figura 8: Matriz de correlación de Spearman del conjunto de datos enriquecido.

Cuadro 4: Distribución porcentual de las clases de fallo en el conjunto sintético.

Categoría	Etiqueta	Proporción (%)
Operación Normal	0	69.00
Falla Mecánica	1	10.29
Falla Enfriamiento	2	10.29
Falla Combustible	3	10.42

## 2.1. Selección de Variables y Preprocesamiento

Para el entrenamiento de los modelos correspondientes a cada subproblema, se excluyeron las siguientes variables del conjunto de datos:

- **Turbine ID:** Se omitió con el fin de evitar el sobreajuste (*overfitting*) hacia unidades específicas. De este modo, se garantiza que el modelo aprenda patrones operativos generales y no correlaciones espurias dependientes de la identidad de cada turbina.
- **Time-to-Collapse (TTC):** Su exclusión es necesaria para prevenir la fuga de datos (*data leakage*). Dado que los estados de degradación y falla se definen de manera determinista a partir de los umbrales  $U_{\text{deterioro}}$  y  $U_{\text{falla}}$ , la inclusión de esta variable induciría al modelo a memorizar horizontes

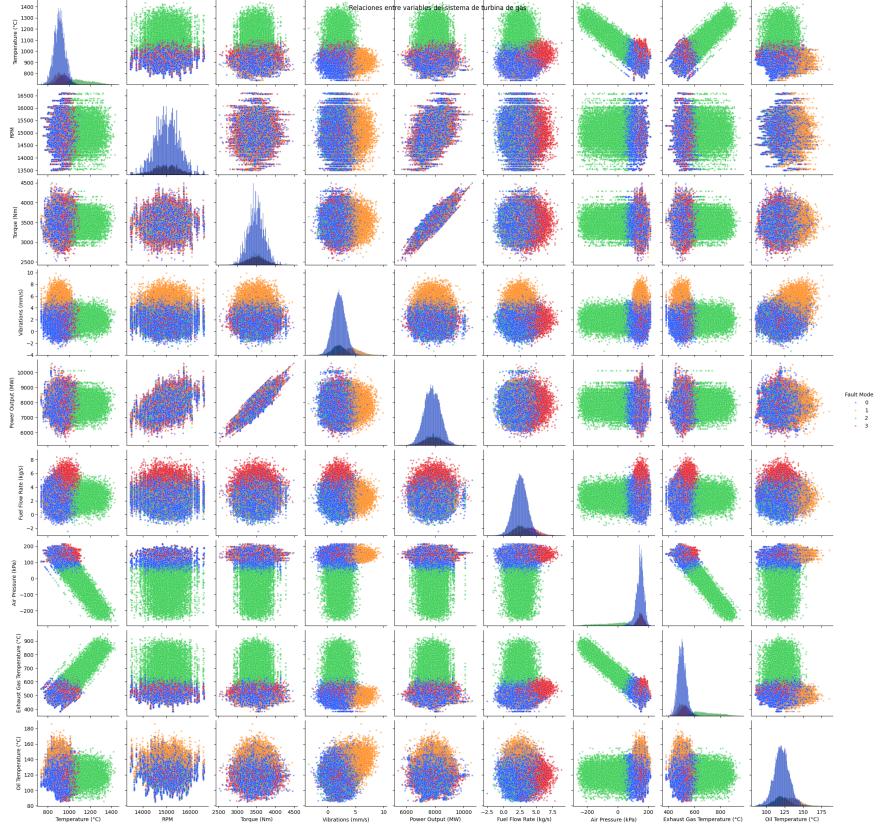


Figura 9: Matriz de dispersión multivariante de las variables sintéticas.

temporales en lugar de caracterizar el comportamiento físico y el estado de salud de los componentes.

## 2.2. Clasificación Binaria: Detección de Fallas Operacionales

El primer objetivo consiste en la discriminación estadística entre el estado normal y el estado de falla del sistema. En la Fig. 10 se presenta una visión general de la distribución del conjunto de datos en función de la variable objetivo binaria, evidenciando las fronteras de decisión entre ambas clases.

Para abordar esta tarea, se implementó un flujo de experimentación basado en la comparación competitiva de diversos algoritmos de aprendizaje supervisado. El proceso de optimización se estructuró bajo las siguientes premisas metodológicas:

- **Optimización de Hiperparámetros:** Se empleó la biblioteca *Optuna*,

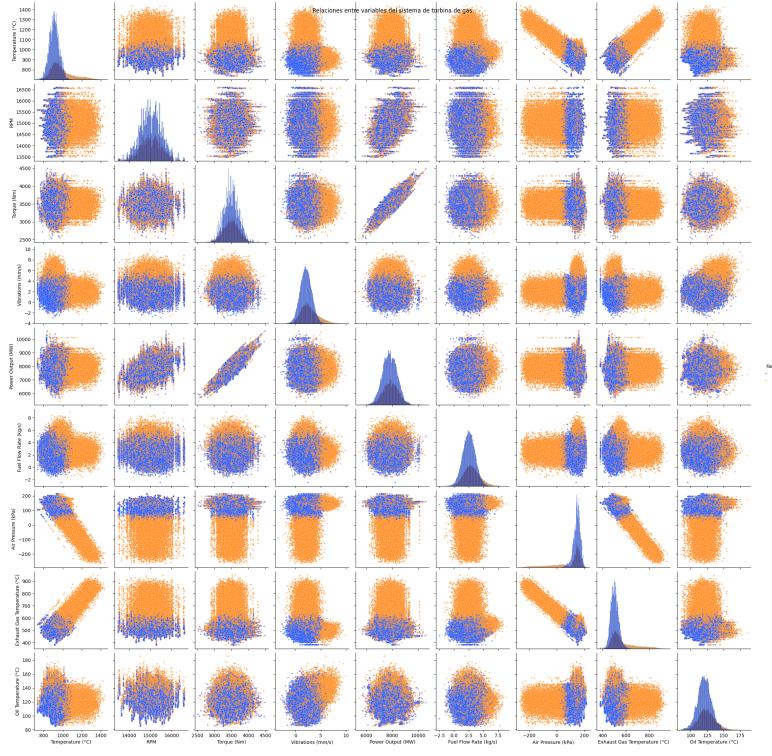


Figura 10: Distribución multivariante de las variables operacionales segmentadas por estado de falla.

utilizando algoritmos de optimización bayesiana para maximizar la eficiencia en la búsqueda de hiperparámetros óptimos.

- **Tratamiento del Desbalance de Clases:** Dada la asimetría en las proporciones de la variable objetivo, se evaluó el rendimiento de determinados modelos bajo tres escenarios de configuración de datos:

1. Conjunto de datos original (sin modificaciones).
2. Aplicación de técnicas de *Undersampling* para equilibrar la clase mayoritaria.
3. Implementación de estrategias de aumentación de datos para fortalecer la representatividad de la clase minoritaria.

La selección del modelo final se fundamenta en un análisis comparativo de métricas de desempeño, priorizando la robustez y la capacidad de generalización del clasificador ante datos no observados. En este contexto, se ha determinado el uso de la métrica  $F_\beta$ -score con  $\beta = 2$  como criterio principal de optimización y selección.

El  $F_2$ -score pondera con mayor relevancia la exhaustividad (*recall*). Esta métrica se define matemáticamente como:

$$F_\beta = (1 + \beta^2) \cdot \frac{\text{precisión} \cdot \text{recall}}{(\beta^2 \cdot \text{precisión}) + \text{recall}} \quad (3)$$

Al fijar  $\beta = 2$ , el modelo se orienta hacia la minimización de los falsos negativos. Esta decisión responde a la naturaleza crítica del mantenimiento predictivo de turbinas: el riesgo operativo y los costos derivados de una falla no detectada (falso negativo) superan significativamente los costos de una inspección preventiva motivada por una falsa alarma (falso positivo). Por lo tanto, el sistema prioriza la sensibilidad en la detección de estados de falla para garantizar la disponibilidad y la integridad de los activos industriales.

### 2.2.1. Modelo de Regresión Logística (*Baseline*)

Se seleccionó el modelo de Regresión Logística como referencia inicial (*baseline*) debido a su idoneidad para tareas de clasificación binaria y su interpretabilidad.

**Espacio de Búsqueda de Hiperparámetros** La optimización se realizó mediante la biblioteca *Optuna*, ejecutando 100 iteraciones para maximizar la métrica  $F_2$ -score. El espacio de búsqueda incluyó las siguientes dimensiones:

- **Regularización y Solver:** Se evaluaron los coeficientes de penalización  $C \in \{0,01, 0,1, 1, 5, 10\}$  junto con diversos algoritmos de optimización (*solvers*), tales como *lbfgs*, *liblinear*, *sag* y *saga*. Dependiendo del *solver*, se aplicaron penalizaciones *L1*, *L2* o *ElasticNet* (con ajustes de *l1\_ratio*).
- **Balanceo de Clases:** Se exploró el ajuste de pesos de clase (*class\_weight*) y estrategias de remuestreo dinámico, incluyendo sobremuestreo mediante SMOTE y submuestreo aleatorio (*RandomUnderSampler*).

**Arquitectura del Experimento y Validación** Para garantizar la robustez del modelo y evitar fugas de datos (*data leakage*), se implementó un *pipeline* de aprendizaje balanceado (*ImbPipeline*) con las siguientes etapas:

1. **Estandarización:** Transformación de las características mediante *StandardScaler* para asegurar una escala uniforme.
2. **Tratamiento de Desbalance:** Aplicación condicional de la técnica de muestreo seleccionada por el optimizador.
3. **Validación Cruzada:** Se empleó un esquema de *Stratified K-Fold* con  $k = 5$  particiones, asegurando que la proporción de clases se mantuviera constante en cada iteración de entrenamiento y prueba.

**Resultados de la Optimización** Tras concluir las iteraciones de búsqueda, el modelo alcanzó su desempeño óptimo bajo la configuración detallada en la Tabla 5.

Cuadro 5: Hiperparámetros óptimos para el modelo de Regresión Logística.

Parámetro	Valor Óptimo
Estrategia de Muestreo	<i>Random Under Sampler</i>
Parámetro de Regularización ( $C$ )	10
Pesos de Clase	<i>None</i>
Optimizador ( <i>Solver</i> )	<i>Saga</i>
Relación $L_1$ ( $l1\_ratio$ )	1 ( $L_1$ pura)
Mejor $F_2$ -score	<b>0.8183</b>

**Análisis de Convergencia de la Optimización** La Fig. 11 ilustra el historial de optimización del modelo de Regresión Logística a lo largo de 100 iteraciones (*trials*). A partir de un punto, la curva del "Mejor Valor" (*Best Value*) muestra una estabilización asintótica cercana a 0.818, lo que indica una convergencia satisfactoria hacia una región de parámetros óptima.

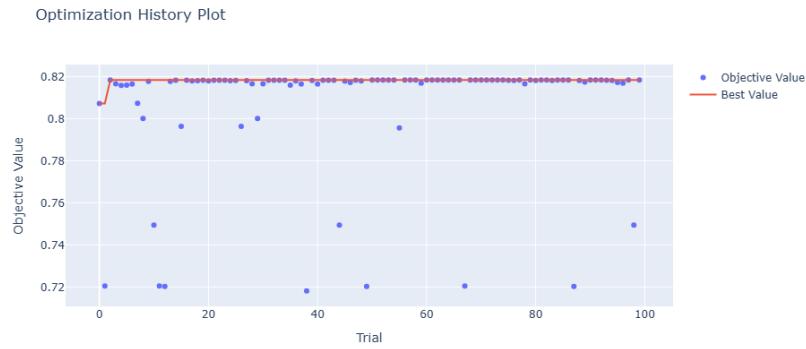


Figura 11: Historial de optimización de la función objetivo mediante búsqueda bayesiana.

**Evaluación de Importancia de Hiperparámetros** El análisis de sensibilidad mediante la métrica de importancia de hiperparámetros (Fig. 12) permite identificar los factores con mayor impacto en la maximización del  $F_2$ -score.

Los resultados indican que el parámetro *class\_weight* (0.48) y la estrategia de *sampling* (0.27) son los componentes determinantes para el rendimiento del modelo.

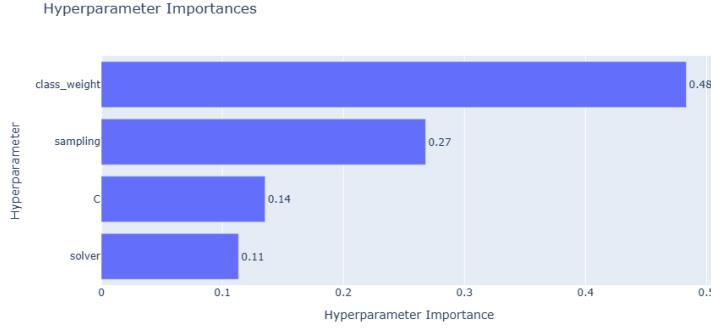


Figura 12: Importancia relativa de los hiperparámetros en el proceso de optimización.

**Análisis de Interacción de Hiperparámetros** Los diagramas de contorno (Fig. 2.2.1) permiten identificar las regiones de máximo rendimiento en el espacio de búsqueda:

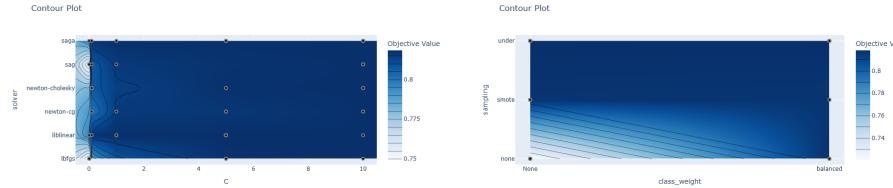


Figura 13: Interacción entre el Solver y el parámetro  $C$ .

Figura 14: Interacción entre Sampling y Class Weight.

**Análisis de la Curva de Aprendizaje** La Fig. 15 revela un sistema con un desempeño robusto y estable, aunque condicionado por las limitaciones intrínsecas de la arquitectura lineal.

Se observa una convergencia asintótica de la métrica de validación a partir de las 30,000 muestras, lo que sugiere que el modelo ha alcanzado un estado de saturación de aprendizaje bajo la configuración actual. La persistencia de una brecha (*gap*) moderada entre las curvas de entrenamiento y validación indica que, si bien la optimización bayesiana mediante *Optuna* ha maximizado el ajuste de los parámetros, el modelo presenta un sesgo (*bias*) residual.

Este estancamiento por debajo del umbral de 0.82 en validación denota una limitación en la capacidad expresiva de la Regresión Logística para capturar relaciones de mayor complejidad en los datos.

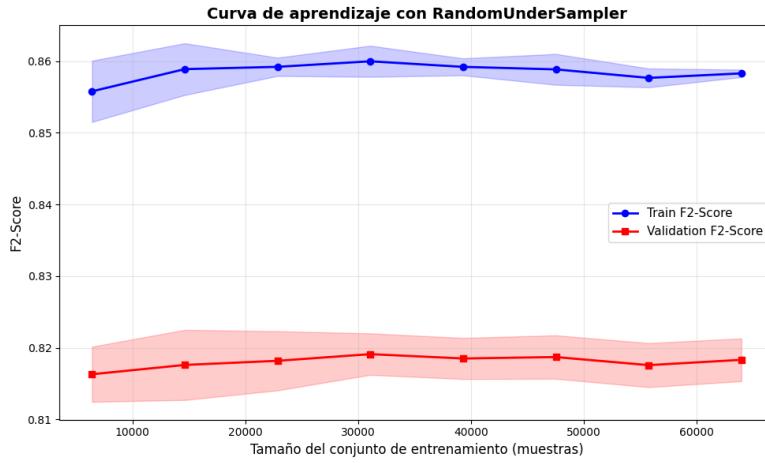


Figura 15: Curva de aprendizaje del modelo de Regresión Logística.

**Evaluación del Modelo en el Conjunto de Prueba** El desempeño final del clasificador de Regresión Logística se evaluó utilizando un conjunto de datos independiente (*test set*) de 20,000 instancias. El modelo demostró una capacidad de generalización robusta, alcanzando un  $F_2$ -score de 0.806.

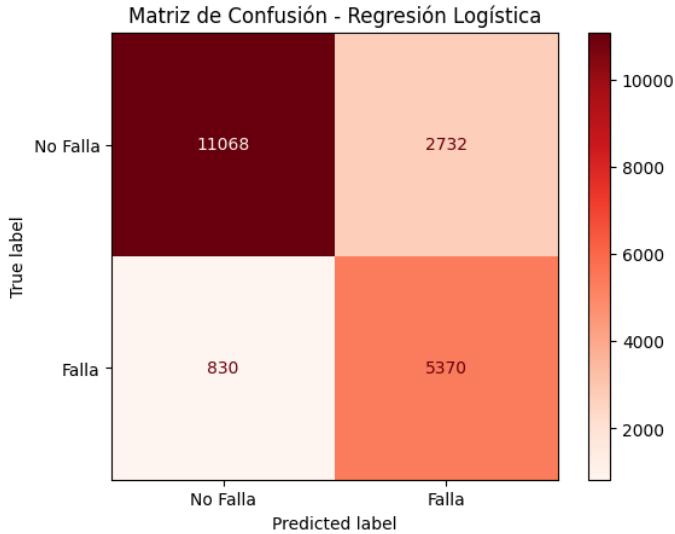


Figura 16: Matriz de confusión del modelo de Regresión Logística en el conjunto de prueba.

Como se observa en la matriz de confusión (Fig. 16), el modelo exhibe una

alta sensibilidad o *recall* (87 %), lo que garantiza la identificación proactiva de la mayoría de los eventos de falla. Aunque la precisión resultante (66 %) conlleva un volumen de 2,732 falsos positivos, esta configuración se considera óptima bajo una filosofía de mantenimiento basado en la condición. En dicho entorno, el impacto económico y de seguridad derivado de un fallo no detectado (*missed detection*) es órdenes de magnitud superior al costo de una inspección técnica preventiva.

En la Tabla 6 se detallan las métricas de desempeño por clase y los promedios ponderados del sistema.

Cuadro 6: Reporte de clasificación detallado para el modelo de Regresión Logística.

Clase	Precisión	Recall	$F_1$ -score	Soporte
0 (Normal)	0.93	0.80	0.86	13,800
1 (Falla)	0.66	0.87	0.75	6,200
<b>Exactitud (Accuracy)</b>			<b>0.82</b>	20,000
<b>Promedio Ponderado</b>	0.85	0.82	0.83	20,000
<b><math>F_2</math>-Score Global</b>			<b>0.8062</b>	

### 2.2.2. Máquinas de Soporte Vectorial (SVM)

Dadas las limitaciones inherentes al modelo de regresión logística para capturar dependencias no lineales y patrones complejos en el espacio de características, se optó por la implementación de Máquinas de Soporte Vectorial (SVM).

**Optimización de Hiperparámetros y Espacio de Búsqueda** Dada la elevada demanda computacional de las Máquinas de Soporte Vectorial (SVM) y las limitaciones de infraestructura disponibles, el proceso de optimización se restringió a 30 iteraciones empleando el marco de trabajo *Optuna*. El espacio de búsqueda se diseñó para priorizar la eficiencia operativa, evaluando los siguientes hiperparámetros:

- **Función de núcleo (*kernel*):** Se contrastaron las funciones de base radial (*RBF*) y sigmoide.
- **Parámetro de regularización ( $C$ ):** Exploración en un rango logarítmico entre  $10^{-2}$  y  $10^2$ .
- **Coeficiente de núcleo ( $\gamma$ ):** Evaluado bajo las configuraciones *scale* y *auto*.

En cuanto al tratamiento del desequilibrio de clases, se limitaron las opciones a la ausencia de balanceo o la aplicación de *Random Undersampling*. Se descartaron técnicas de aumentación de datos (sobremuestreo) para mitigar la introducción de ruido en la distribución y prevenir un incremento prohibitivo en los tiempos de convergencia del modelo. Adicionalmente, se integró una etapa

de estandarización de características para garantizar el correcto funcionamiento del espacio de búsqueda de la SVM.

**Resultados de la Optimización** Tras completar el proceso de optimización bayesiana, se obtuvo un valor máximo para la métrica  $F_2$  de 0,8687. La configuración óptima de hiperparámetros derivada del espacio de búsqueda se detalla a continuación:

Cuadro 7: Hiperparámetros óptimos para el modelo SVM.

Parámetro	Valor Óptimo
Estrategia de Muestreo	<i>None</i>
Función de Núcleo ( <i>Kernel</i> )	<i>RBF</i>
Parámetro de Regularización ( $C$ )	69.848
Coeficiente de Núcleo ( $\gamma$ )	<i>Auto</i>
<b>Mejor <math>F_2</math>-score</b>	<b>0.8687</b>

**Análisis de Convergencia** Como se ilustra en la Fig. 17, la trayectoria de optimización muestra una tendencia de convergencia hacia valores de la métrica  $F_2$  superiores a 0,85. A pesar de la restricción en el presupuesto computacional (limitado a 30 iteraciones), el algoritmo de búsqueda bayesiana logró estabilizarse en una región de parámetros de alto rendimiento.

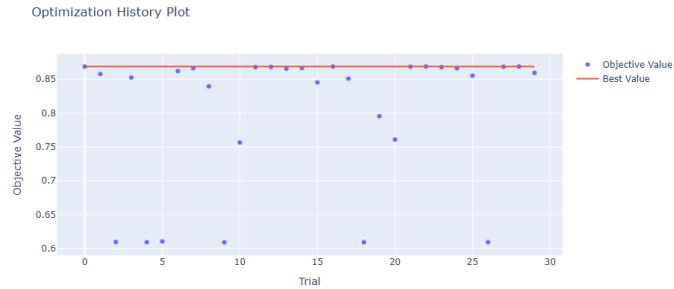


Figura 17: Evolución de la función objetivo y convergencia de la métrica  $F_2$  mediante optimización bayesiana.

**Análisis de Sensibilidad de Hiperparámetros** El análisis de importancia relativa presentado en la Fig. 18 revela que la configuración del núcleo (*kernel*) constituye el factor determinante en el desempeño predictivo del modelo. Por otro lado, la aplicación de estrategias de remuestreo exhibió una influencia marginal en la optimización, sugiriendo que la arquitectura de la frontera de decisión es intrínsecamente robusta ante el desequilibrio de clases para este problema específico.

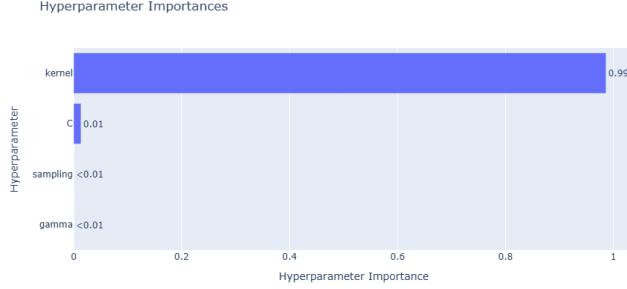


Figura 18: Importancia relativa de los hiperparámetros en el proceso de optimización.

**Análisis de Interacción de Hiperparámetros** El mapa de contorno presentado en la Fig. 19 permite identificar una región de rendimiento óptimo altamente localizada dentro del espacio de búsqueda. Se observa que el desempeño del modelo presenta una alta sensibilidad ante variaciones en la configuración; cualquier transición fuera del núcleo *RBF* resulta en una degradación sustancial de la métrica  $F_2$ , con valores inferiores a 0,7.

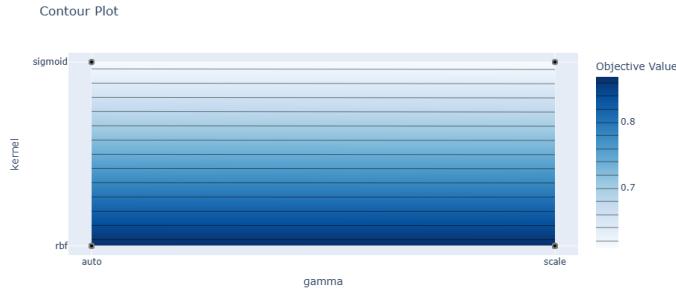


Figura 19: Superficie de respuesta para la interacción entre  $\gamma$  y la función de núcleo.

**Análisis de la Curva de Aprendizaje** El comportamiento de las curvas (Fig. 20) revela una convergencia asintótica saludable, característica de un modelo con un equilibrio adecuado entre sesgo y varianza. Se observa que, mientras el  $F_2$ -score de entrenamiento experimenta un descenso marginal al incrementar la complejidad del conjunto de datos, la métrica de validación muestra una tendencia ascendente sostenida, estabilizándose por encima de 0,87. La reducción progresiva del margen (*gap*) entre ambas curvas confirma una sólida capacidad de generalización, sugiriendo que el modelo ha capturado los patrones operati-

vos fundamentales y que el volumen de datos actual es suficiente para alcanzar la estabilidad predictiva sin incurrir en sobreajuste significativo.

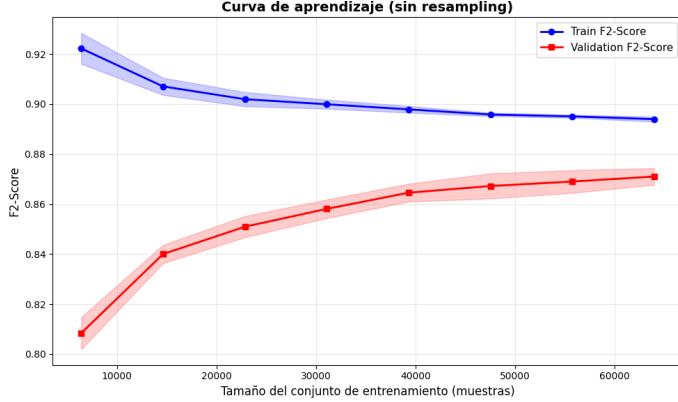


Figura 20: Evolución del desempeño ( $F_2$ -score) y diagnóstico de capacidad de generalización del modelo SVM.

**Evaluación del Modelo en el Conjunto de Prueba** El desempeño final del clasificador SVM se evaluó mediante un conjunto de prueba independiente de 20,000 muestras. El modelo alcanzó un  $F_2$ -score de 0,8737, lo que confirma una alta eficacia en la identificación de fallas. Como se observa en la matriz de confusión (Fig. 21) y en la Tabla 8, el modelo exhibe una robustez notable:

Cuadro 8: Reporte de clasificación detallado para el modelo SVM.

Clase	Precisión	Sensibilidad	F1-Score	Soporte
No Falla (0.0)	0.96	0.85	0.90	13,800
Falla (1.0)	0.73	0.92	0.82	6,200
<b>Exactitud (Accuracy)</b>	<b>0.87</b>		20,000	
<b>Mejor <math>F_2</math>-score</b>	<b>0.8737</b>		—	

### 2.2.3. Modelo XGBoost (*Extreme Gradient Boosting*)

Se seleccionó el algoritmo XGBoost debido a su alto desempeño en tareas de clasificación con datos tabulares, su capacidad para capturar dependencias no lineales complejas y ofrecer una eficiencia computacional superior. Presenta robustez frente a datos balanceados por lo que se optó por prescindir de técnicas de remuestreo externo (*sampling*) en el conjunto de entrenamiento, preservando así la integridad de la distribución original de los datos.

**Optimización de Hiperparámetros** La sintonización de hiperparámetros se ejecutó mediante 100 iteraciones de optimización bayesiana. Para optimizar

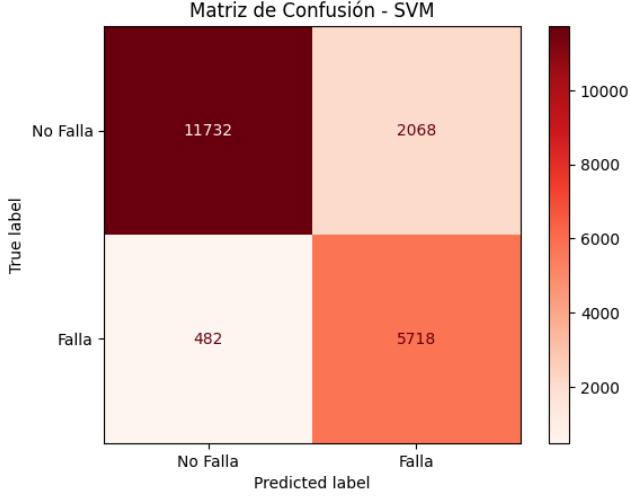


Figura 21: Matriz de confusión del modelo SVM en el conjunto de prueba, evidenciando el balance entre falsos positivos y negativos.

la eficiencia y prevenir el sobreajuste, se implementó una estrategia de parada temprana (*early stopping*) con un margen de paciencia de 50 rondas, permitiendo determinar dinámicamente el número óptimo de estimadores según la convergencia de la función de pérdida. El espacio de búsqueda incluyó:

- **Arquitectura del árbol:** Profundidad máxima (*max\_depth*) entre 3 y 10, y peso mínimo del nodo hijo (*min\_child\_weight*) para controlar la complejidad de las hojas.
- **Regularización:** Se integraron parámetros  $\gamma$  (reducción mínima de pérdida),  $\lambda$  (regularización  $L_2$ ) y  $\alpha$  (regularización  $L_1$ ) con variaciones en escala logarítmica.
- **Robustez y Muestreo:** Se evaluaron fracciones de submuestreo por instancias (*subsample*) y por características (*colsample\_bytree*) entre 0,6 y 1,0.
- **Tratamiento del desbalanceo:** Se ajustó el parámetro *scale\_pos\_weight* en torno al ratio empírico de las clases ( $\frac{y=0}{y=1}$ ), permitiendo una oscilación del  $\pm 10\%$  para optimizar la sensibilidad del modelo hacia la clase minoritaria.

**Resultados de la Optimización** Tras completar las 100 iteraciones de búsqueda bayesiana, el modelo XGBoost alcanzó un valor máximo en la métrica  $F_2$  de 0,8488. La configuración de hiperparámetros que maximizó la función objetivo se presenta en la Tabla 9.

Cuadro 9: Hiperparámetros óptimos para el modelo XGBoost.

Parámetro	Valor Óptimo
Tasa de aprendizaje ( <i>learning_rate</i> )	0.0273
Profundidad máxima ( <i>max_depth</i> )	10
Peso mínimo del nodo hijo ( <i>min_child_weight</i> )	17
$\gamma$ ( <i>Lagrangian multiplier</i> )	$3,459 \times 10^{-7}$
$\lambda$ (Regularización $L_2$ )	7.739
$\alpha$ (Regularización $L_1$ )	0.0067
Submuestreo de instancias ( <i>subsample</i> )	0.997
Submuestreo de características ( <i>colsample_bytree</i> )	0.986
Ponderación de clase positiva ( <i>scale_pos_weight</i> )	2.438
<b>Mejor <math>F_2</math>-score</b>	<b>0.8488</b>

**Análisis de la Curva de Aprendizaje** La Fig. 22 presenta la evolución del desempeño del modelo XGBoost en relación con el tamaño del conjunto de entrenamiento. El análisis de las trayectorias permite extraer las siguientes conclusiones técnicas:

- **Convergencia y Estabilidad:** Se observa una reducción progresiva de la brecha (*gap*) entre las métricas de entrenamiento y validación a medida que aumenta el volumen de muestras. Ambas curvas tienden a estabilizarse asintóticamente hacia un  $F_2$ -score superior a 0,76 en validación, lo que indica una convergencia exitosa del algoritmo.
- **Diagnóstico de Varianza:** La proximidad final entre las curvas sugiere que el modelo posee una alta capacidad de generalización. La varianza se mantiene controlada, demostrando que los mecanismos de regularización aplicados ( $\alpha$  y  $\lambda$ ) han mitigado efectivamente el riesgo de sobreajuste (*overfitting*).
- **Suficiencia de Datos:** La pendiente de la curva de validación se suaviza tras superar las 40,000 muestras, sugiriendo que el tamaño del conjunto de datos actual es adecuado para que el modelo capture la estructura subyacente del problema de detección de fallas sin necesidad de incrementar exponencialmente la muestra.

**Evaluación del Modelo en el Conjunto de Prueba** La evaluación final del modelo XGBoost sobre el conjunto de prueba independiente ( $n = 20,000$ ) arrojó un  $F_2$ -score de 0,8325. Este resultado ratifica la consistencia del modelo frente a los valores obtenidos durante la fase de optimización. Los indicadores clave de desempeño, detallados en la Tabla 10 y la Fig. 23, se analizan a continuación:

Se observa una precisión de 0,76 en la identificación de fallas, lo que representa una reducción en el volumen de falsos positivos (1,557 casos) en comparación con modelos previamente analizados. Esto sugiere una frontera de decisión más conservadora y específica.

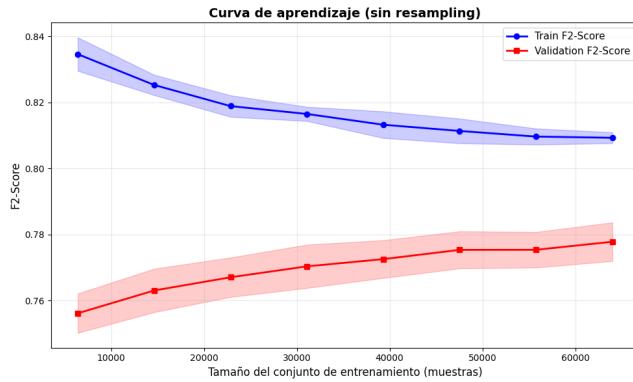


Figura 22: Evolución del  $F_2$ -score y análisis de convergencia para el modelo XGBoost.

Cuadro 10: Reporte de clasificación para el modelo XGBoost en el conjunto de prueba.

Clase	Precisión	Sensibilidad	F1-Score	Soporte
No Falla (0.0)	0.90	0.89	0.89	13,800
Falla (1.0)	0.76	0.78	0.77	6,200
<b>Exactitud (Accuracy)</b>	<b>0.85</b>			20,000
<b>Mejor <math>F_2</math>-score</b>	<b>0.8325</b>			—

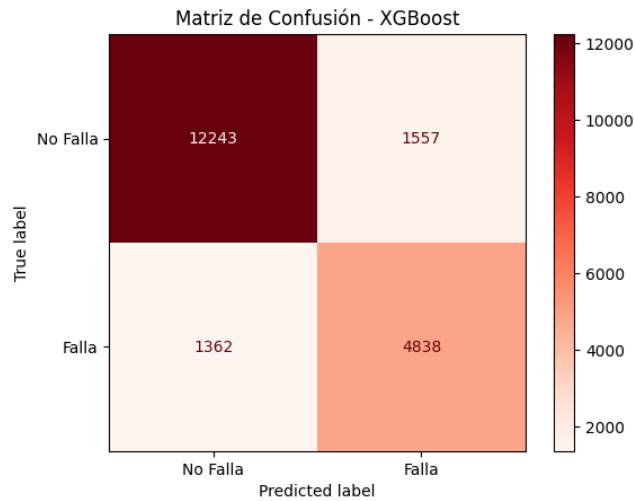


Figura 23: Matriz de confusión del modelo XGBoost, reflejando el desempeño predictivo en condiciones de prueba.

#### **2.2.4. Ensamble mediante Clasificador de Votación (*Voting Classifier*)**

Ante la incapacidad de las configuraciones individuales de XGBoost para superar el desempeño obtenido por la SVM y la regresión logística, se exploró una arquitectura de ensamble basada en un *VotingClassifier*. Para ello, se seleccionaron los tres mejores modelos (Top 3) derivados del proceso de optimización bayesiana de XGBoost, integrándolos mediante una estrategia de votación suave (*soft voting*) basada en la promediación de probabilidades posteriores.

No obstante, los resultados en el conjunto de prueba no mostraron una mejora incremental significativa. El ensamble alcanzó un  $F_2$ -score de 0,8336, valor inferior a los registros previos. Asimismo, se observó una degradación simultánea en las métricas de precisión y sensibilidad (*recall*), lo que sugiere que la combinación de modelos altamente correlacionados dentro del mismo espacio de búsqueda no aportó la diversidad estocástica necesaria para mejorar la frontera de decisión.

#### **2.2.5. Modelo de Bosques Aleatorios (*Random Forest*)**

Tras los resultados limitados de los métodos de *boosting*, se optó por un enfoque basado en *bagging* mediante Bosques Aleatorios. Esta arquitectura ofrece una frontera de decisión más estable y robusta frente al sobreajuste, permitiendo capturar estructuras de datos que no fueron optimizadas por los algoritmos anteriores.

**Configuración de la Optimización y Estrategias de Balanceo** Se evaluaron dos estrategias competitivas: i) un submuestreo aleatorio externo (*Random Under Sampling*) aplicado exclusivamente al conjunto de entrenamiento, y ii) el ajuste intrínseco de pesos mediante el parámetro *balanced\_subsample*, el cual recalcula los pesos de clase para cada partición de *bootstrap*.

El espacio de búsqueda se definió con un límite de 300 estimadores para optimizar la eficiencia computacional, explorando profundidades máximas (*max\_depth*) entre 5 y 20, y una restricción en el tamaño mínimo de hoja (*min\_samples\_leaf*) de 2 a 10 para prevenir el sobreajuste. Este esquema se validó mediante un proceso de validación cruzada donde la técnica de balanceo se seleccionó de forma categórica en cada iteración de la optimización bayesiana.

**Resultados de la Optimización** Tras el proceso de búsqueda bayesiana, el modelo de Bosques Aleatorios alcanzó un  $F_2$ -score máximo de 0,7919. La configuración óptima identificada (Tabla 11) seleccionó el submuestreo aleatorio (*undersampling*) como estrategia de balanceo, con un ensamble de 179 estimadores y una profundidad máxima de 19 niveles. Cabe destacar que este desempeño es significativamente inferior a los resultados obtenidos por los modelos SVM y XGBoost. Esta disparidad sugiere que, para este espacio de características, los métodos basados en la promediación de árboles independientes presentan una

capacidad de discriminación menor frente a las fronteras de decisión generadas por los vectores de soporte o el aumento de gradiente.

Cuadro 11: Hiperparámetros óptimos para el modelo Random Forest.

Parámetro	Valor Óptimo
Estrategia de Balanceo	<i>Random Under Sampling</i>
Número de Estimadores ( <i>n_estimators</i> )	179
Profundidad Máxima ( <i>max_depth</i> )	19
Mínimo de Muestras por Hoja ( <i>min_samples_leaf</i> )	3
<b>Mejor <math>F_2</math>-score</b>	<b>0.7928</b>

**Análisis de la Curva de Aprendizaje** La Fig. 24 muestra el desempeño de *Random Forest* bajo submuestreo. Se observan los siguientes puntos clave:

- **Alta Varianza:** Existe una brecha persistente entre el entrenamiento ( $\approx 0.94$ ) y la validación ( $\approx 0.79$ ), evidenciando un sobreajuste (*overfitting*) que la arquitectura no logró mitigar.
- **Sesgo de Remuestreo:** El alto rendimiento en entrenamiento sugiere una memorización del subconjunto balanceado con baja capacidad de transferencia a la distribución original.
- **Saturación:** La estabilización de la curva de validación tras las 40,000 muestras indica que el modelo ha alcanzado su techo predictivo, por lo que aumentar el volumen de datos no mejoraría la generalización.

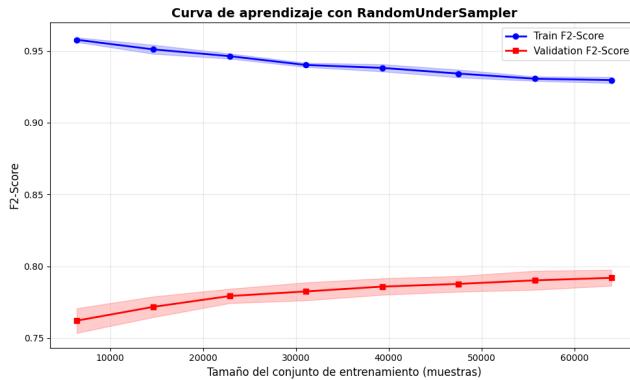


Figura 24: Curva de aprendizaje del modelo Random Forest con técnica de submuestreo aleatorio.

**Evaluación del Modelo en el Conjunto de Prueba** El desempeño final del clasificador *Random Forest* se validó con el conjunto de prueba independiente, obteniendo un  $F_2$ -score de 0,8484. A pesar de ser un resultado competitivo, el análisis detallado de la matriz de confusión (Fig. 25) y el reporte de clasificación (Tabla 12) revela limitaciones estructurales frente a otros modelos:

- **Capacidad de Recuperación (*Recall*):** El modelo logró identificar 4,955 fallas reales de un total de 6,200, lo que se traduce en un *recall* de 0,80. Aunque es un valor sólido, representa una pérdida de sensibilidad del 12 % en comparación con el modelo SVM, aumentando el riesgo de omitir eventos críticos.
- **Precisión y Especificidad:** El modelo muestra un equilibrio notable con una precisión del 0,78 para la clase de falla y una especificidad del 0,91 para la clase sana. Esto indica una reducción efectiva de falsos positivos (1,388 casos), superando en este aspecto específico a la configuración de la SVM.
- **Métricas Agregadas:** La exactitud (*accuracy*) del 0,87 y un F1-score de 0,79 para la clase minoritaria confirman que el ensamble de árboles por *bagging* es robusto, aunque menos eficaz que el enfoque de vectores de soporte para el objetivo principal de maximizar el  $F_2$ -score.

Cuadro 12: Reporte de clasificación detallado para el modelo Random Forest.

Clase	Precisión	Sensibilidad	F1-Score	Soporte
No Falla (0.0)	0.91	0.90	0.90	13,800
Falla (1.0)	0.78	0.80	0.79	6,200
<b>Exactitud (<i>Accuracy</i>)</b>		<b>0.87</b>		20,000
<b>Mejor <math>F_2</math>-score</b>		<b>0.8484</b>		—

#### 2.2.6. Ensamble Híbrido: *Random Forest* y *XGBoost*

La integración de *bagging* (Random Forest) y *boosting* (XGBoost) busca mejorar la generalización mediante la combinación de distintos sesgos inductivos. Mientras el primero reduce la varianza por promediación, el segundo optimiza secuencialmente los residuos, permitiendo teóricamente una frontera de decisión más equilibrada y resiliente al ruido.

Para este experimento, se implementó un clasificador de votación (*Voting-Classifier*) con una estrategia de ponderación suave (*soft voting*), promediando las probabilidades predichas por ambos algoritmos optimizados. Sin embargo, la evaluación en el conjunto de prueba resultó en un  $F_2$ -score de 0,8426, valor que no superó el desempeño de los modelos individuales ni el de la SVM.

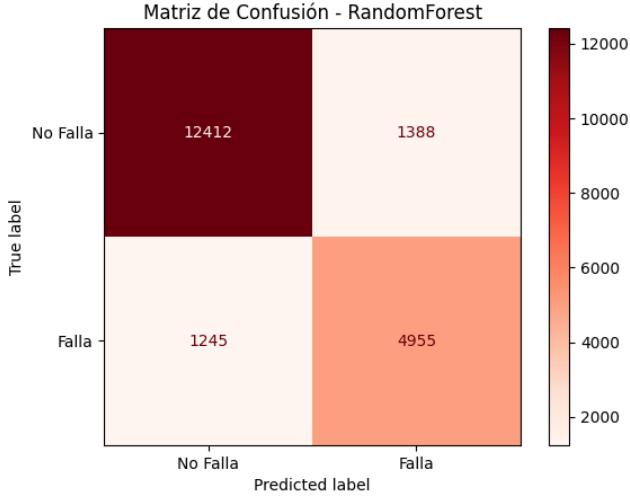


Figura 25: Matriz de confusión del modelo Random Forest. Se observa una mayor tasa de falsos negativos en comparación con el modelo SVM.

#### 2.2.7. Modelo de Red Neuronal

Se ha implementado un Perceptrón Multicapa (MLP). La estructura se basa en una red densamente conectada con capacidad de adaptación mediante el ajuste de su profundidad (1 a 2 capas ocultas). El modelo se integra en un flujo de trabajo (*pipeline*) que incluye una etapa de preprocesamiento con `StandardScaler` y una etapa de remuestreo para gestionar el desbalance de clases, utilizando técnicas como SMOTE o *Undersampling*.

La arquitectura final seleccionada tras la optimización consta de una capa de entrada (9 variables), seguida de dos capas ocultas de 32 y 128 neuronas respectivamente, y una capa de salida con activación logística para la clasificación final. Esta configuración suma un total de 320 parámetros entrenables.

**Optimización de Hiperparámetros y Espacio de Búsqueda** Se empleó el framework `Optuna` para realizar una búsqueda bayesiana sobre el espacio de hiperparámetros. El objetivo fue maximizar la métrica  $F_2$ -Score mediante validación cruzada estratificada ( $k = 3$ ). El espacio de búsqueda definido fue:

- **Estrategia de muestreo:** {None, SMOTE, UnderSampling}.
- **Número de capas ocultas:** 1 o 2.
- **Número de neuronas por capa:** {32, 64, 128}.
- **Regularización ( $\alpha$ ):**  $[1 \times 10^{-5}, 1 \times 10^{-2}]$  (escala logarítmica).
- **Tasa de aprendizaje inicial:**  $[0,001, 0,01]$  (escala logarítmica).

- **Máximo de iteraciones:** [100, 300].
- **Optimizador:** Adam (fijo).
- **Función de activación:** ReLU (fija).

**Resultados de la optimización** Tras realizar 20 iteraciones (*trials*), se identificó que la combinación óptima de parámetros mejora significativamente la capacidad de detección del modelo. Los mejores hiperparámetros encontrados se detallan en la tabla 13:

Hiperparámetro	Valor Óptimo
Estrategia de Muestreo	SMOTE
Arquitectura	(32, 128)
$\alpha$ (Regularización)	$1,249 \times 10^{-5}$
Tasa de Aprendizaje	0.00851
Máximo de Iteraciones	148
<b>Mejor <math>F_2</math>-Score</b>	<b>0.8684</b>

Cuadro 13: Resultados finales de la optimización con Optuna.

**Análisis de convergencia** El proceso de optimización mostró una convergencia estable hacia soluciones de alto rendimiento. El  $F_2$ -Score promedio a través de los 20 *trials* fue de 0.8470, con una desviación estándar de 0.0284, lo que indica que el espacio de búsqueda estaba bien definido. Se observó que el uso de *Early Stopping* (activado con una fracción de validación del 0.1) fue crucial para prevenir el sobreajuste, permitiendo que el modelo final se detuviera de manera eficiente al alcanzar la estabilidad en la pérdida (Fig. 26).

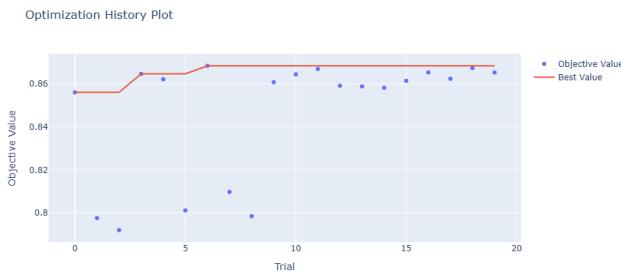


Figura 26: Curva de convergencia de las iteraciones en optuna.

**Análisis de sensibilidad de hiperparámetros** El análisis de importancia de hiperparámetros revela que el factor determinante en el rendimiento del modelo es la estrategia de manejo de datos desbalanceados (Fig. 27 y 2.2.7):

1. **Sampling (91.11 %)**: Es el parámetro más crítico. El uso de SMOTE elevó el score promedio a 0.8631, mientras que no aplicar remuestreo (*none*) resultó en un score significativamente inferior (0.7997).
2. **Max\_iter (6.28 %)**: La duración del entrenamiento tiene un impacto secundario pero relevante para asegurar la convergencia del optimizador Adam.
3. **Learning Rate (1.89 %)**: La tasa de aprendizaje mostró una sensibilidad baja dentro del rango evaluado, sugiriendo robustez en el proceso de descenso de gradiente.
4. **Arquitectura (0.58 %)**: El número de capas y neuronas tuvo la menor importancia relativa, lo que indica que una vez balanceados los datos, diferentes configuraciones de red logran resultados competitivos similares.

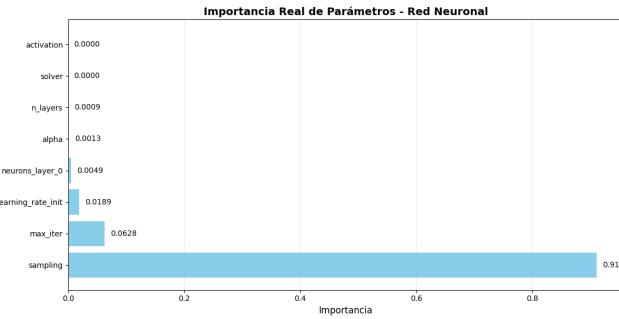


Figura 27: Importancia real de hiperparametros.

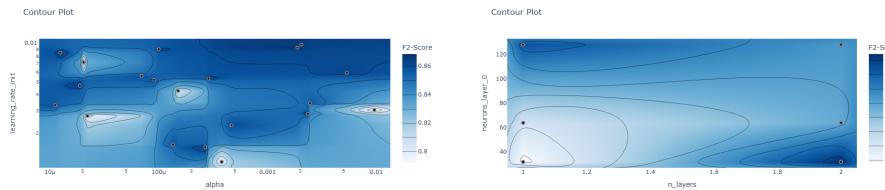


Figura 28: Regularización ( $\alpha$ ) vs Tasa de Aprendizaje.

Figura 29: Número de Capas vs Neuronas Primera Capa.

**Análisis de rendimiento de las tres variantes de sampling** El análisis estadístico de los experimentos (*trials*) demuestra que las técnicas de remuestreo son fundamentales para la eficacia del modelo (Fig. 30).

De acuerdo con los datos de rendimiento y la distribución observada en los *trials* de Optuna:

- **Eficacia del Balanceo:** Tanto SMOTE como *Undersampling* superan significativamente a la configuración base (*none*), elevando el promedio del  $F_2$ -Score de 0.7997 a valores superiores a 0.86.
- **Estabilidad y Consistencia:** La estrategia SMOTE no solo obtuvo el valor máximo absoluto (0.8684), sino que presenta una dispersión de resultados competitiva, consolidándose como la opción óptima para el modelo final.

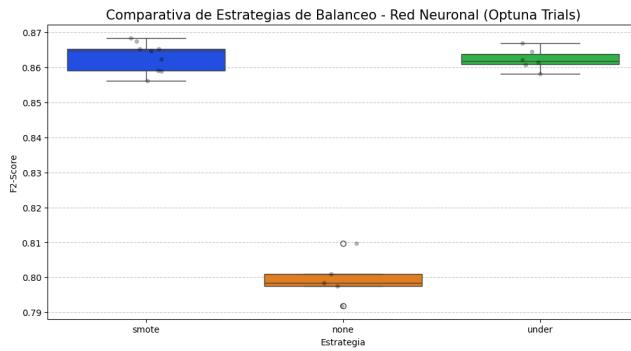


Figura 30: Comparación de las tres variantes de sampling.

**Análisis de curva de aprendizaje** El análisis de la progresión del aprendizaje revela que el modelo alcanza su estabilidad operativa a partir de las 64,000 muestras originales (expandidas a 88,320 mediante SMOTE). Los resultados consolidados se presentan en la tabla 14 y en la figura 31:

Muestras (Post-SMOTE)	Train $F_2$ -Score	Val $F_2$ -Score	Brecha (Gap)
8,832	0.9319	0.8085	0.1234
42,898	0.9161	0.8539	0.0622
<b>88,320</b>	<b>0.9209</b>	<b>0.8677 (<math>\pm 0.0071</math>)</b>	<b>0.0532</b>

Cuadro 14: Resumen de rendimiento y convergencia.

- **Diagnóstico de Varianza:** Se observa una brecha constante de 0,05 entre entrenamiento y validación. Esto indica un sobreajuste (*overfitting*) ligero y controlado, propio de la capacidad de memorización de la red neuronal ante datos sintéticos (SMOTE).

- **Punto de Saturación:** La curva de validación muestra una pendiente positiva decreciente hacia el final del intervalo, sugiriendo que el modelo ha extraído casi la totalidad de la información útil del dataset actual; incrementos adicionales de datos tendrían retornos marginales.
- **Estabilidad Técnica:** La desviación estándar mínima ( $\pm 0,0071$ ) confirma que el modelo es altamente robusto y que los resultados no dependen de una partición específica de los datos, validando la fiabilidad del  $F_2$ -Score final.

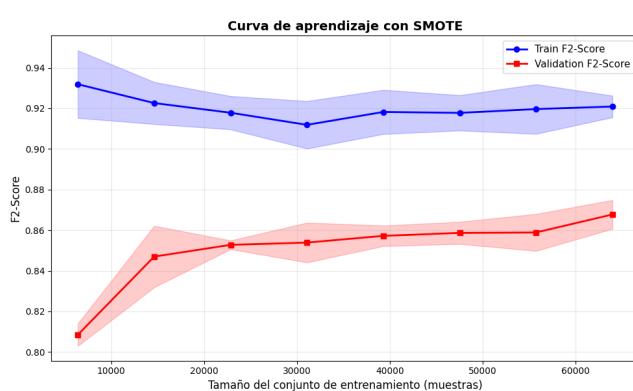


Figura 31: Curva de Aprendizaje.

**Evaluación del modelo en el conjunto prueba** Los resultados confirman que la red neuronal mantiene un alto rendimiento fuera del entrenamiento, con un enfoque efectivo en la detección de fallas. El modelo alcanzó un  **$F_2$ -Score de 0.8653**, lo cual es consistente con los resultados de la validación cruzada y demuestra que el modelo no presenta un sobreajuste significativo (Tabla 15).

Métrica	Valor en Test
$F_2$ -Score (Optimizado)	0.8653
$F_1$ -Score (Macro)	0.8616
Precisión Global (Accuracy)	0.8800

Cuadro 15: Resumen de métricas en el conjunto de prueba.

La matriz de confusión revela el comportamiento del modelo frente a la criticidad de las clases (Fig. 32):

- **Sensibilidad (Recall) de Fallas (90 %):** El modelo identifica correctamente a 5,582 de las 6,200 fallas reales. Solo se omitieron 618 casos (falsos negativos), lo cual es fundamental en este caso donde una falla no detectada implica costos elevados.

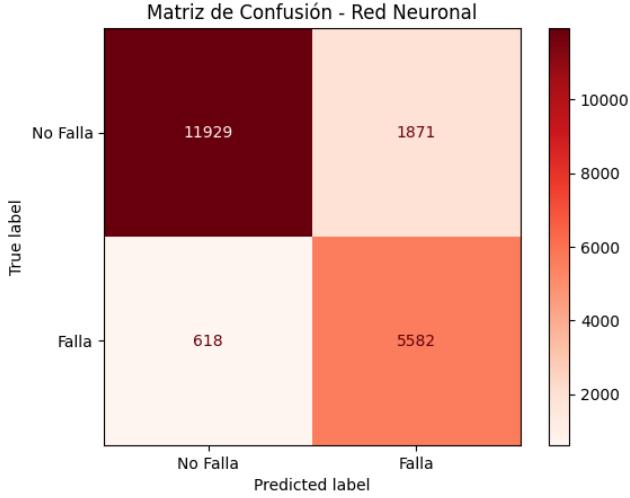


Figura 32: Curva de Aprendizaje.

- **Precisión de Fallas (75 %):** De todas las alarmas de falla generadas, el 75 % son correctas. Los 1,871 falsos positivos representan "falsas alarmas" que, aunque requieren inspección, son preferibles a la omisión de una falla crítica bajo la métrica  $F_2$ .
- **Robustez en Clase Mayoritaria:** La precisión para la clase "No Falla" es del 95 %, asegurando que el modelo es altamente confiable cuando predice un estado operativo normal.

#### 2.2.8. Ensamble Híbrido: SVM y Regresión Logística

Considerando que la Máquina de Vectores de Soporte (SVM) y la Regresión Logística presentaron un desempeño individual bastante bueno, se evaluó su integración mediante un clasificador de votación (*VotingClassifier*). Se empleó una estrategia de votación suave (*soft voting*) con una asignación de pesos asimétrica, otorgando mayor relevancia a las probabilidades estimadas por la SVM debido a su superioridad métrica previa.

Tras la evaluación en el conjunto de prueba, el ensamble alcanzó un  $F_2$ -score de 0,8745 (Tabla 16). Si bien este valor representa una mejora marginal en el puntaje global respecto a la SVM individual, el análisis detallado revela una ligera degradación en la precisión de la clase positiva y en la estructura de la matriz de confusión (Fig. 33).

Debido a que el incremento en la complejidad temporal y computacional del ensamble no se traduce en una mejora significativa de las métricas de clasificación, se descarta esta arquitectura. En consecuencia, se selecciona la SVM como el modelo óptimo final, priorizando un equilibrio superior entre capacidad predictiva y eficiencia operativa.

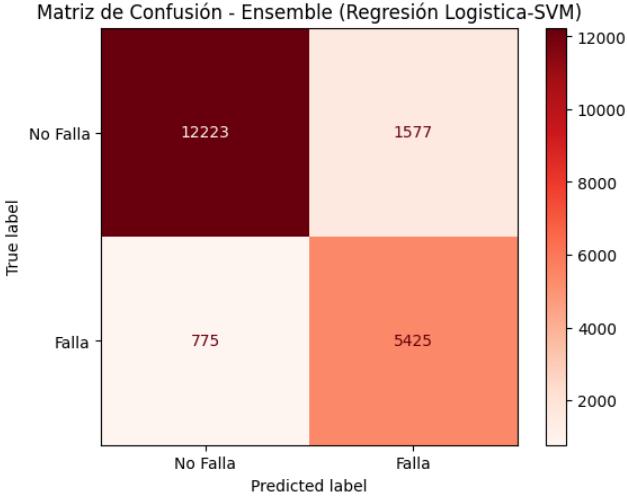


Figura 33: Matriz de confusión del ensemble de Regresión Logística con SVM

Cuadro 16: Reporte de clasificación del ensamble ponderado (SVM + Regresión Logística).

Clase	Precisión	Sensibilidad	F1-Score	Soporte
No Falla (0.0)	0.94	0.89	0.91	13,800
Falla (1.0)	0.77	0.88	0.82	6,200
<b>Exactitud (Accuracy)</b>		<b>0.88</b>		20,000
<b>Mejor <math>F_2</math>-score</b>		<b>0.8745</b>		—

### 2.3. Clasificación Multiclas: Diagnóstico del Modo de Falla

El segundo objetivo se centra en la identificación específica de la etiología de la falla para aquellas unidades que han superado el umbral de detección inicial. A diferencia del primer problema, este análisis se restringe exclusivamente al subconjunto de datos donde se ha verificado una anomalía, excluyendo la etiqueta de operación nominal (clase 0). La distribución de las variables operacionales para las tres categorías de falla remanentes se ilustra en la Fig. 35.

**Metodología de Experimentación y Evaluación** El proceso de selección de modelos y optimización de hiperparámetros sigue una estructura similar a la fase anterior, aunque con ajustes críticos derivados de la naturaleza del subconjunto de datos:

- **Balanceo de Clases:** Debido a que la distribución de las categorías en la variable objetivo *Fault Mode* presenta una paridad intrínseca (como se analizó en la Tabla 4), se determinó omitir las técnicas de remuestreo

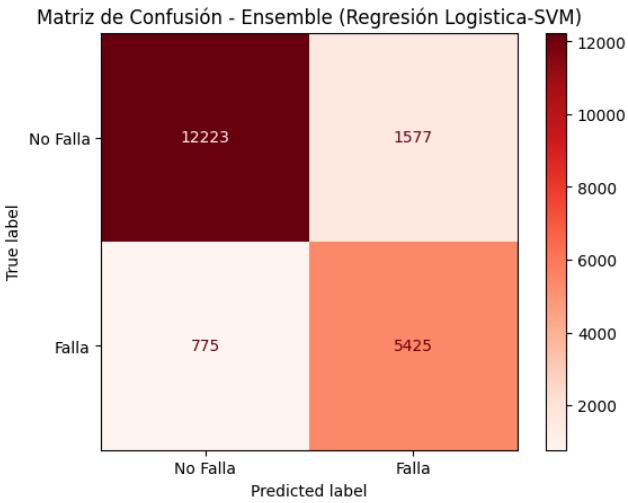


Figura 34: Matriz de confusión del ensemble

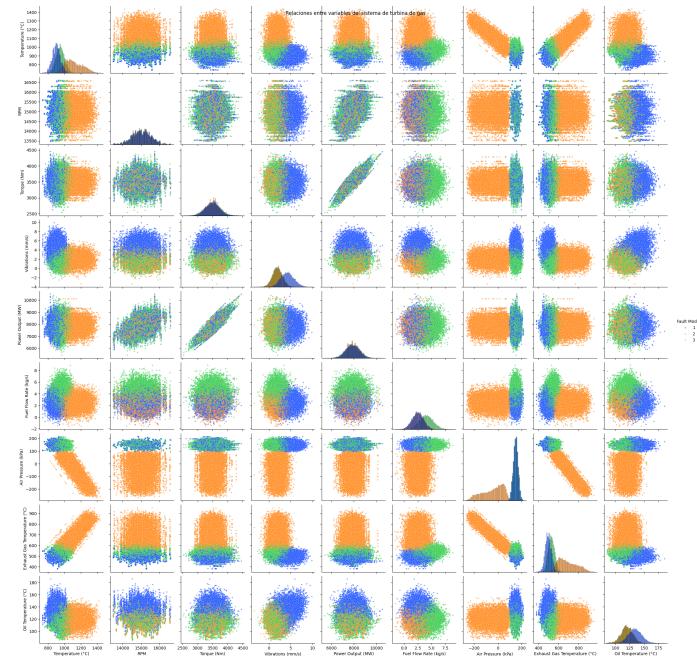


Figura 35: Distribución multivariante y fronteras de separación de los distintos modos de falla.

(*undersampling* o aumentación), permitiendo al modelo aprender directamente

mente de las proporciones originales.

- **Criterio de Optimización:** Dada la importancia equivalente de diagnosticar correctamente cada uno de los tres tipos de falla, se seleccionó el *F<sub>1</sub>-macro score* como métrica principal para la optimización. Esta métrica otorga el mismo peso a cada categoría independientemente de su soporte, lo que garantiza un diagnóstico equitativo.

Este enfoque permite evaluar la capacidad del clasificador para capturar las sutilezas mecánicas y térmicas que distinguen, por ejemplo, una falla en el lazo de combustible de una falla por enfriamiento.

### 2.3.1. Regresión Logística (*Baseline* Multiclas)

Se implementó un modelo de Regresión Logística multinomial como punto de referencia para el diagnóstico de fallos. La optimización se realizó sobre el subconjunto de instancias con anomalías, utilizando un *pipeline* que integra estandarización de características y búsqueda exhaustiva en malla (*Grid Search*) con validación cruzada ( $k = 5$ ). El objetivo primordial fue maximizar el *F<sub>1</sub>-macro* para garantizar un diagnóstico equitativo entre las categorías.

**Optimización y Sensibilidad de Parámetros** La búsqueda determinó que la configuración óptima consiste en un penalizador  $L_2$  con  $C = 5$  y el algoritmo *lbfgs* (Tabla 17).

Cuadro 17: Configuración óptima y métricas de validación para la Regresión Logística.

Hiperparámetro	Valor Óptimo
Algoritmo de Optimización ( <i>Solver</i> )	<i>lbfgs</i>
Tipo de Penalización ( <i>Penalty</i> )	$L_2$
Parámetro de Regularización ( $C$ )	5
Pesos de Clase ( <i>Class Weight</i> )	<i>None</i>
<b>Mejor <i>F<sub>1</sub>-macro</i> (CV)</b>	<b>0.9477</b>

La figuras 36 y 37 revela una alta consistencia del sistema, con una varianza entre *splits* inferior a 0.005. Los mapas de calor y eficiencia (Fig. 38 y 39) confirman que la convergencia se estabiliza para  $C \geq 0.1$ , destacando la superioridad computacional de los métodos de segundo orden en este espacio de características.

**Dinámica de Aprendizaje** La curva de aprendizaje (Fig. 40) muestra una convergencia asintótica a partir de las 15,000 muestras. La desaparición de la brecha entre las métricas de entrenamiento y validación confirma que el modelo ha alcanzado su capacidad máxima de aprendizaje (*saturation point*) sin incurrir en sobreajuste.

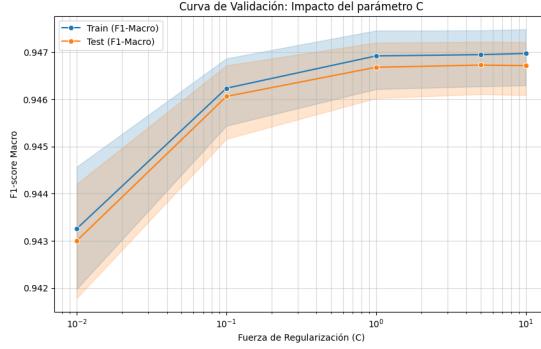


Figura 36: Curva de validación: Impacto del parámetro  $C$  en el  $F_1$ -macro.

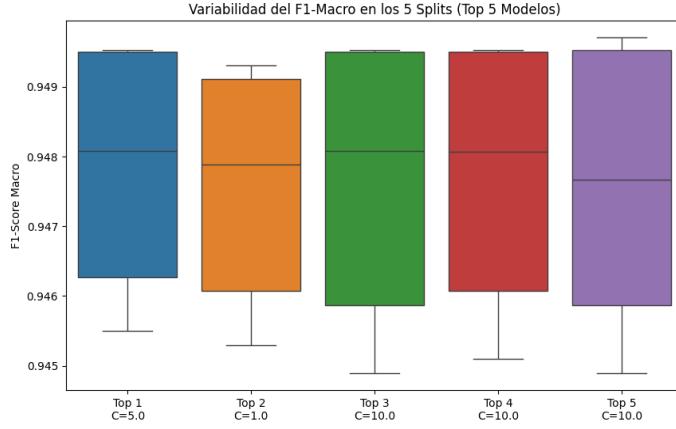


Figura 37: Variabilidad de la metrica en los 5 splits

**Evaluación en el Conjunto de Prueba** El modelo alcanzó un  $F_1$ -macro global de 0.9467 en el conjunto de prueba, demostrando una robusta capacidad de generalización. Los resultados detallados se presentan en la Tabla 18 y la matriz de confusión de la Fig. 41.

#### Interpretación del Diagnóstico:

- **Separabilidad Lineal:** La Clase 2 se identifica con precisión perfecta ( $F_1 = 1.00$ ), indicando firmas estadísticas unívocas.
- **Solapamiento de Características:** Se detectó una confusión cruzada del 8 % entre las Clases 1 y 3. Esto sugiere que las fallas mecánicas y de combustible comparten similitudes en las señales de los sensores que limitan la resolución de un discriminador lineal.



Figura 38: Mapa de calor del rendimiento de los hiperparametros

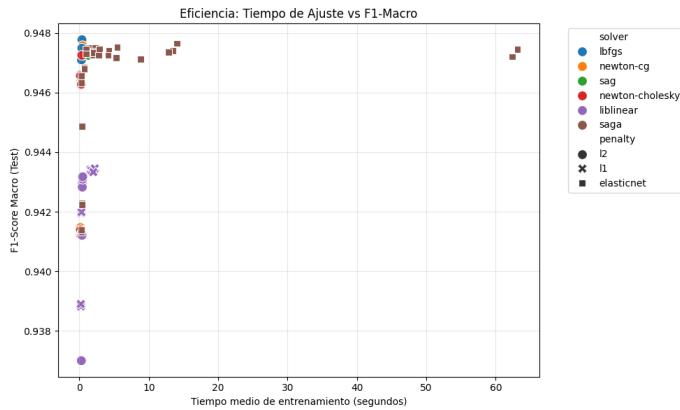


Figura 39: Eficiencia de los modelos (Metrica vs Tiempo)

- **Estabilidad:** La paridad entre precisión y exhaustividad (*recall*) confirma un modelo balanceado y exento de sesgos por clase.

### 2.3.2. Clasificador de $k$ Vecinos Más Cercanos (KNN)

La selección del modelo KNN se fundamenta en el análisis exploratorio previo, donde los diagramas de dispersión revelaron una estructura de datos caracterizada por la formación de conglomerados (*clusters*) bien definidos en el espacio de características.

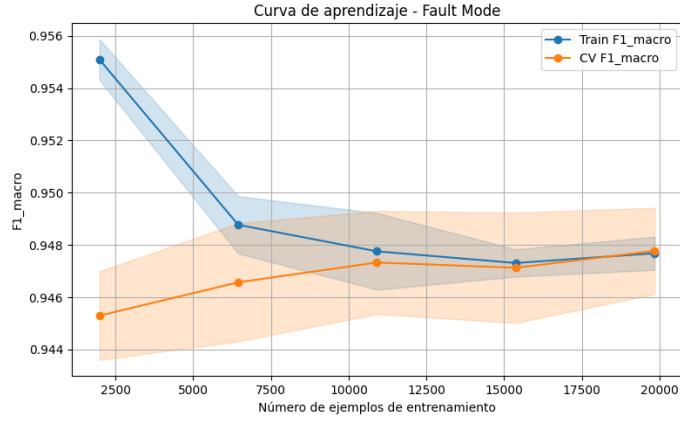


Figura 40: Curva de aprendizaje

Cuadro 18: Reporte de clasificación detallado (Conjunto de Prueba).

Modo de Falla	Precisión	Recall	$F_1$ -score	Sopporte
1 (Mecánica)	0.92	0.92	0.92	2,059
2 (Enfriamiento)	1.00	1.00	1.00	2,058
3 (Combustible)	0.92	0.92	0.92	2,083
<b>Promedio Macro</b>	<b>0.95</b>	<b>0.95</b>	<b>0.95</b>	6,200

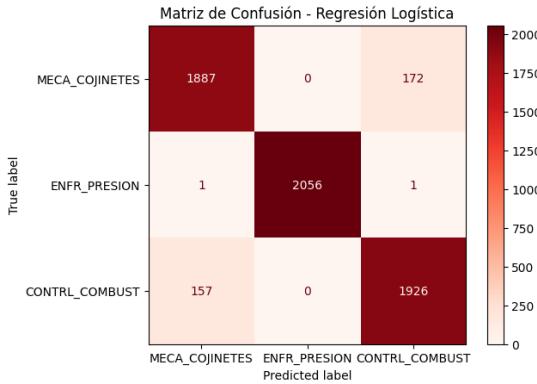


Figura 41: Matriz de confusión: Diagnóstico multiclas de fallos.

**Optimización de Hiperparámetros y Análisis de Validación** Se realizó una búsqueda exhaustiva mediante validación cruzada de 5 pliegues (*5-fold CV*), evaluando un total de 5,040 combinaciones. La configuración óptima identificada empleó el algoritmo *auto* con  $k = 11$  vecinos, una distancia de Minkowski con  $p = 1$  (Manhattan) y ponderación por distancia, alcanzando un  $F_1$ -score macro de 0,9469.

**Análisis Comparativo del Top 5 de Configuraciones** La evaluación de los cinco mejores candidatos derivados de la optimización (Fig. 42) permite extraer las siguientes conclusiones sobre la estabilidad del clasificador:

- **Convergencia de Hiperparámetros:** Se observa un escenario de cuasiempate técnico, donde todas las variantes convergen en  $k = 11$  y ponderación por distancia (*weights='distance'*). La invarianza del desempeño ante la métrica de distancia (Minkowski vs. Manhattan) sugiere una estructura geométrica robusta en el espacio de características tridimensional.
- **Análisis de Robustez y Dispersión:** El  $F_1$ -score promedio se sitúa en 0,948, con una oscilación mínima entre 0,943 y 0,949. La asimetría superior en los diagramas de caja indica que, si bien el modelo es altamente preciso en la mayoría de las particiones, existen subconjuntos de datos con mayor complejidad de clasificación o presencia de valores atípicos (*outliers*), lo que desplaza la mediana hacia los valores superiores del rango.

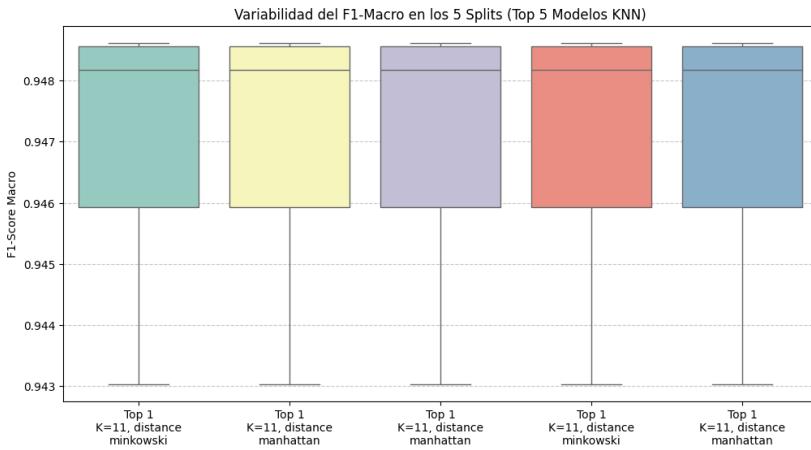


Figura 42: Distribución del desempeño para las cinco mejores configuraciones del modelo KNN.

A partir del análisis de la Fig. 43, se identifica que las regiones de máximo desempeño convergen en configuraciones que emplean la métrica de Manhattan bajo un esquema de ponderación por distancia (*distance weighting*). Este comportamiento es particularmente consistente para un rango de proximidad situado entre  $k = 5$  y  $k = 11$  vecinos.

**Análisis de Eficiencia Computacional y Rendimiento** La Fig. 44 permite evaluar el compromiso entre la precisión del modelo y el costo computacional invertido. Se observa que la configuración basada en el algoritmo de búsqueda automática (*auto*), un número moderado de vecinos ( $k$ ) y ponderación por distancia (*distance weights*), constituye el punto de operación óptimo en términos

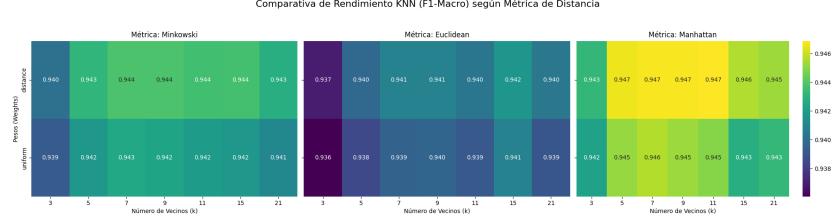


Figura 43: Evaluación del desempeño según la interacción entre la métrica de distancia y el número de vecinos ( $k$ ).

de eficiencia. Esta combinación no solo maximiza las métricas de clasificación, sino que reduce significativamente la latencia de procesamiento en comparación con configuraciones más complejas o búsquedas exhaustivas en rangos de  $k$  elevados. En consecuencia, se identifica una región de saturación donde incrementos adicionales en los recursos de cómputo no derivan en mejoras marginales significativas de la precisión, validando la selección de los hiperparámetros actuales.

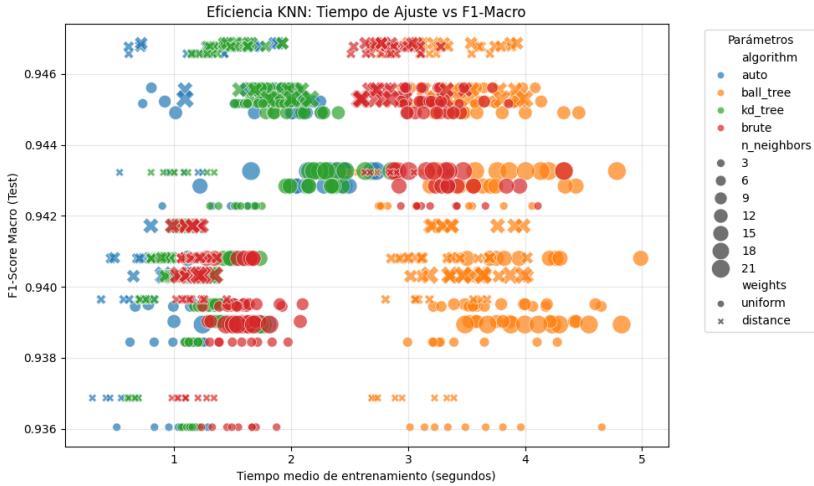


Figura 44: Análisis de eficiencia: Relación entre la precisión y el tiempo de ejecución para distintas configuraciones de KNN.

**Análisis de la Curva de Aprendizaje** La Fig. 45 detalla la evolución del clasificador KNN según el volumen de datos, destacando los siguientes puntos:

- **Convergencia:** El  $F_1$ -macro de validación asciende de 0,932 a 0,947, indicando una mayor capacidad de generalización al densificar el espacio de características y refinar las fronteras de decisión.

- **Estabilidad:** La reducción de las bandas de confianza al aproximarse a las 20,000 muestras evidencia una menor variabilidad y confirma la robustez de los hiperparámetros seleccionados.
- **Generalización:** Aunque persiste un rendimiento perfecto en entrenamiento (1,0), la reducción progresiva de la brecha diagnóstica y la pendiente final sugieren que el modelo está próximo a su madurez operativa, con mejoras marginales ante incrementos adicionales de datos. El modelo parece haber memorizado en vez de aprendido habiendo *overfitting*.

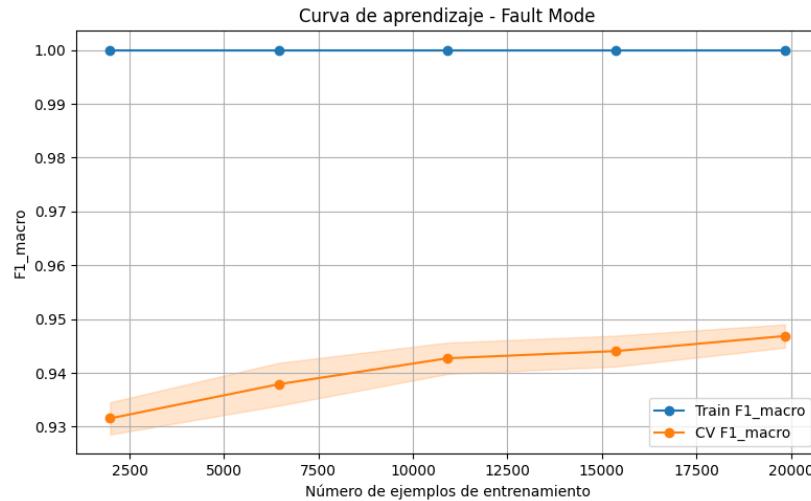


Figura 45: Curva de aprendizaje del modelo KNN para la clasificación de modos de falla.

**Evaluación Final en el Conjunto de Prueba** El clasificador KNN seleccionado fue evaluado en un conjunto de prueba independiente de 6,200 muestras, alcanzando un  $F_1$ -macro global de 0,9522. Los resultados detallados en el reporte de clasificación (Tabla 19) y la matriz de confusión (Fig. 46) permiten realizar las siguientes observaciones:

- **Precisión por Categoría:** El modelo demuestra un desempeño excepcional en la identificación de la clase 2 (*ENFR\_PRESION*), con una sensibilidad (*recall*) cercana a la unidad (0,99). Esto indica una separación casi lineal de esta falla en el espacio de características.
- **Confusión entre Clases:** Se observa un ligero solapamiento entre las clases 1 (*MECA\_COJINETES*) y 3 (*CTRL\_COMBUST*), con 156 y 126 falsos positivos cruzados, respectivamente. Esta interacción sugiere una similitud en los patrones de señal de ambos modos de falla, aunque el modelo logra mantener precisiones individuales superiores al 0,92.

- **Robustez Multiclasa:** La consistencia entre el promedio macro y ponderado (0,95) confirma que el modelo no presenta sesgos significativos hacia ninguna de las categorías, logrando un equilibrio óptimo en la detección de fallas críticas.

Cuadro 19: Reporte de clasificación final para el modelo KNN (Prueba).

Modo de Falla	Precisión	Sensibilidad	F1-Score	Soporte
MECA_COJINETES (1)	0.94	0.92	0.93	2,059
ENFR_PRESION (2)	1.00	0.99	1.00	2,058
CONTRL_COMBUST (3)	0.92	0.94	0.93	2,083
<b>Exactitud (Accuracy)</b>		<b>0.95</b>		6,200
<i>F<sub>1</sub>-macro final</i>		<b>0.9522</b>		—

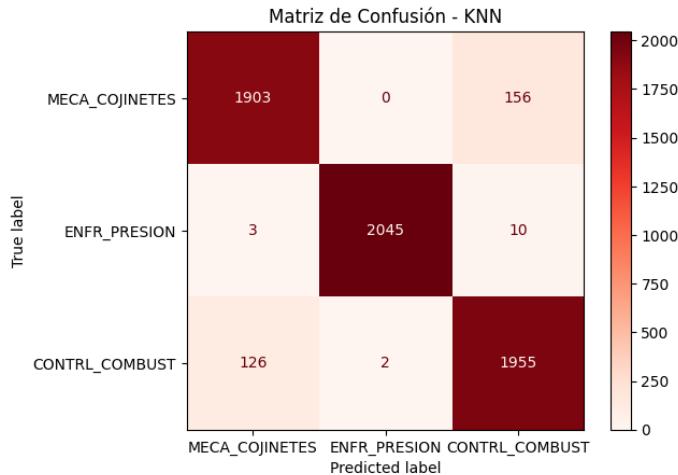


Figura 46: Matriz de confusión final del modelo KNN. Se destaca la alta tasa de aciertos en la diagonal principal para las tres categorías de falla.

### 2.3.3. Máquina de Vectores de Soporte (SVM)

Tras observar un solapamiento marginal entre las clases 1 (*MECA\_COJINETES*) y 3 (*CONTRL\_COMBUST*) en los modelos anteriores, se procedió a la implementación de una Máquina de Vectores de Soporte (SVM) buscando identificar hiperplanos de separación óptimos que minimicen el error de clasificación en las regiones de mayor ambigüedad topográfica entre ambas categorías.

**Optimización de Hiperparámetros** Se llevó a cabo un proceso de ajuste fino de hiperparámetros utilizando una estrategia de búsqueda sistemática. Los

resultados determinaron que la configuración óptima para maximizar la generalización del modelo se basa en un *kernel* de Función de Base Radial (RBF), alcanzando un  $F_1$ -macro de 0,9483 en validación cruzada.

La arquitectura final se define en la Tabla 20.

Cuadro 20: Hiperparámetros óptimos para el modelo SVM.

Parámetro	Valor Óptimo
Escalado ( <i>scaler</i> )	Estandarización ( <i>standard</i> )
Kernel ( <i>kernel</i> )	RBF ( <i>radial basis function</i> )
Parámetro de regularización ( $C$ )	195.606
Parámetro del kernel ( $\gamma$ )	0.0422
<b>Mejor <math>F_1</math>-score macro</b>	<b>0.9483</b>

**Análisis de la Curva de Aprendizaje** La Fig. 47 ilustra la evolución del desempeño de la SVM frente al volumen de datos. El análisis revela los siguientes puntos clave:

- **Convergencia:** Se observa un acercamiento asintótico entre las curvas de entrenamiento y validación, con el  $F_1$ -macro de validación estabilizándose en aproximadamente 0,948.
- **Reducción de Varianza:** A diferencia de modelos previos, la métrica de entrenamiento desciende de 0,982 a 0,962, mientras que la de validación asciende de forma constante. Este comportamiento indica una reducción efectiva del sobreajuste (*overfitting*) y una mejor generalización del hiperplano.
- **Estabilidad:** El estrechamiento de los intervalos de confianza confirma que la configuración de los hiperparámetros  $C$  y  $\gamma$  es robusta frente a diferentes particiones del conjunto de datos.

**Evaluación Final en el Conjunto de Prueba** El modelo SVM fue validado con el conjunto de prueba independiente, alcanzando un  $F_1$ -macro de 0,9427. A continuación, se analizan los resultados derivados del reporte de clasificación (Tabla 21) y la matriz de confusión (Fig. 48):

- **Desempeño por Clase:** Se ratifica una precisión perfecta (1,00) y una sensibilidad total en la identificación de la clase 2 (*ENFR\_PRESION*), consolidándola como la categoría con mayor separabilidad lineal en el espacio transformado.
- **Análisis de Errores de Clasificación:** Sigue existiendo confusión a la hora de diferenciar correctamente entre las clases 1 (*MECA\_COJINETES*) y 3 (*CTRL\_COMBUST*).

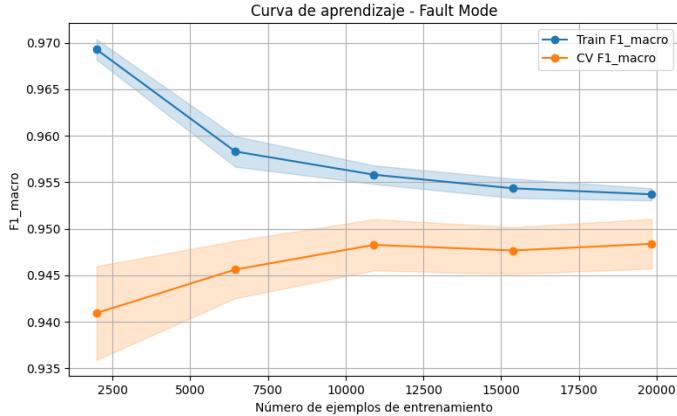


Figura 47: Curva de aprendizaje del modelo SVM. Se aprecia una convergencia saludable indicativa de una generalización óptima.

- **Generalización Global:** Con una exactitud (*accuracy*) del 0,94, el modelo demuestra una robustez operativa ligeramente peor que KNN.

Cuadro 21: Reporte de clasificación detallado para el modelo SVM (Prueba).

Modo de Falla	Precisión	Sensibilidad	F1-Score	Soporte
MECA_COJINETES (1)	0.91	0.92	0.91	2,059
ENFR_PRESION (2)	1.00	1.00	1.00	2,058
CONTRL_COMBUST (3)	0.92	0.91	0.91	2,083
<b>Exactitud (<i>Accuracy</i>)</b>	<b>0.94</b>			6,200
<i>F<sub>1</sub>-macro final</i>	<b>0.9427</b>			—

### 3. Conclusiones

Tras el análisis exhaustivo de los diversos modelos y arquitecturas de aprendizaje supervisado, se presentan las siguientes conclusiones para la resolución de ambos problemas:

- **Problema 1 (Detección de Fallas):** El modelo seleccionado es la **Máquina de Vectores de Soporte (SVM)**. Esta decisión se fundamenta en su capacidad superior de generalización y, principalmente, en su alto índice de sensibilidad (*recall*). Dado que el objetivo primordial en este escenario es la detección temprana de eventos críticos para el mantenimiento preventivo, la SVM demostró ser la arquitectura más eficaz para minimizar los falsos negativos en comparación con modelos de ensamble como *Random Forest*.

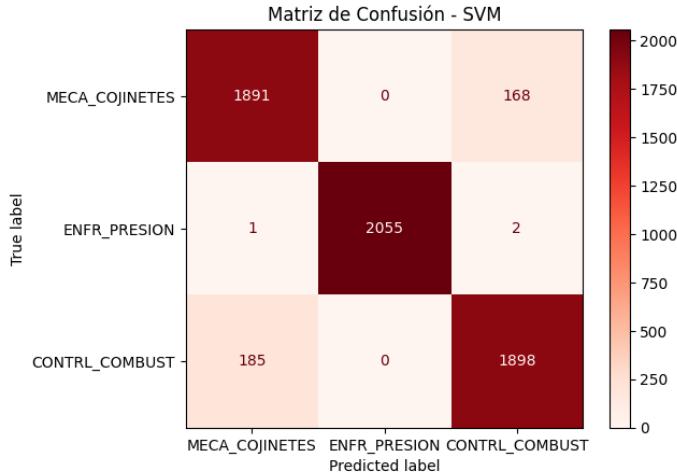


Figura 48: Matriz de confusión del modelo SVM. Se observa un equilibrio en la distribución de errores entre las clases 1 y 3.

- **Problema 2 (Clasificación Multiclasa):** El modelo seleccionado es el de  $k$  **Vecinos Más Cercanos (KNN)**. A pesar de que la curva de aprendizaje exhibió un escenario de sobreajuste (*overfitting*) debido a un rendimiento perfecto en el conjunto de entrenamiento, este algoritmo superó consistentemente a la SVM en las métricas de validación y prueba. Con un  $F_1$ -macro de 0,9522 en el conjunto de test, KNN demostró una capacidad robusta para discernir entre los tres modos de falla, aprovechando eficientemente la densidad y proximidad de los datos en el espacio de características.

En resumen, la selección final priorizó el balance entre la métrica objetivo ( $F_2$ -score para el Problema 1 y  $F_1$ -macro para el Problema 2) y la fiabilidad de los resultados en entornos de prueba independientes.

## Referencias

- [1] Claire Soares. Gas turbines in simple cycle and combined cycle applications, 2008. Condensed extracts from “Gas Turbines: A Handbook of Land, Sea and Air Applications”.
- [2] Ahmed Maged, Salah Hardiy, and Herman Shen. Explainable artificial intelligence techniques for accurate fault detection and diagnosis – a review. *arXiv preprint arXiv:2404.11597v2*, 2024.
- [3] Denis Leite, Emmanuel Andrade, Diego Rativa, and Alexandre M. A. Maciel. Fault detection and diagnosis in industry 4.0: A review on challenges and opportunities. *Sensors*, 25(1), 2025. doi: 10.3390/s25010060.
- [4] Shiyu Xing, Zinan Wang, Rui Zhao, Xirui Guo, Aoxiang Liu, and Wenfeng Liang. Time-frequency-domain fusion cross-attention fault diagnosis method based on dynamic modeling of bearing rotor system. *Applied Sciences*, 15(14), 2025. doi: 10.3390/app15147908.
- [5] Sridhar Alla and Suman Kalyan Adari. *Beginning Anomaly Detection Using Python-Based Deep Learning: With Keras and PyTorch*. Apress, New York, 2019.
- [6] Tuan-Anh Le Nguyen, Tamás Ruppert, and János Abonyi. The use of explainable artificial intelligence and machine learning operation principles to support the continuous development of machine learning-based solutions in fault detection and identification. *Applied Sciences*, 13(8):4937, 2023. doi: 10.3390/app13084937.
- [7] Kevin P. Murphy. *Machine Learning: A Probabilistic Perspective*. MIT Press, Cambridge, MA, 2012.
- [8] Erroumayssae Sabani, El Mehdi Loualid, Hicham El Hadraoui, Kossai Fakir, El Mehdi Laadissi, Adil Balhamri, Chouaib Ennawaoui, and Azzeddine Azim. Bearing faults diagnosis of a turbogenerator system using machine learning training functions. *Diagnostics*, 14(19):2356, 2024. doi: 10.3390/diagnostics14192356.
- [9] Richard O. Duda, Peter E. Hart, and David G. Stork. *Pattern Classification*. Wiley, New York, 2nd edition, 2001.
- [10] Stuart Russell and Peter Norvig. *Artificial Intelligence: A Modern Approach*. Pearson, Hoboken, NJ, 4th edition, 2020.
- [11] Helen E. Bechara, Rony Ibrahim, Ryad Zemouri, Bachir Kedjar, Arezki Merkhouf, Antoine Tahan, and Kamal Al-Haddad. Review of artificial intelligence methods for faults monitoring, diagnosis, and prognosis in hydroelectric synchronous generators. *IEEE Access*, 12:78296–78312, 2024. doi: 10.1109/ACCESS.2024.3409142.

- [12] Youpeng Zhang and Hongsheng Su. Turbo-generator vibration fault diagnosis based on pso-bp neural networks. *WSEAS Transactions on Systems and Control*, 5(1):37–47, 2010.