Texto

Descripción generada automáticamente

Universidad Internacional de La Rioja

Escuela Superior de Ingeniería y

Tecnología

Máster Universitario en Inteligencia artificial

Plataforma de Industria 4.0 basada en IA para predecir fallas en máquinas industriales.

|  |  |
| --- | --- |
| Trabajo fin de estudio presentado por: | Harol Estevez |
| Tipo de trabajo: | Piloto Experimental |
| Director/a: | Fermín Rodríguez Lalanne |
| Fecha: | 30 de junio de 2025 |

Resumen

El presente Trabajo de Fin de Máster aborda el desafío de las paradas no planificadas en la industria de envasado mediante el diseño, desarrollo y validación de una plataforma de mantenimiento predictivo basada en Inteligencia Artificial (IA) para la Industria 4.0. El objetivo principal fue anticipar fallos en máquinas envasadoras de refrescos en polvo con hasta tres días de antelación. La metodología se basó en la adquisición de datos multivariados en tiempo real (vibración, temperatura, corriente, etc.) a través de sensores y PLCs, los cuales se transmiten a un ecosistema en la nube de AWS. Para el análisis de estas series temporales, se implementó un modelo de red neuronal recurrente de tipo Long Short-Term Memory (LSTM), entrenado con datos históricos y balanceado mediante la técnica SMOTE. Los resultados demostraron una alta efectividad del modelo, alcanzando una precisión global del 96% y un recall del 99% para la detección de fallos, lo que minimiza la ocurrencia de paradas inesperadas. La validación económica, mediante un análisis de costos, proyectó una reducción del 75.66% en los gastos operativos asociados a fallos, lo que representa un ahorro de más de 1.8 millones de dólares. Se concluye que la solución es técnicamente robusta y ofrece un alto retorno de la inversión, validando el uso de la IA para optimizar la eficiencia y competitividad en la manufactura.

Palabras clave: Mantenimiento Predictivo, Industria 4.0, Inteligencia Artificial, LSTM, Series Temporales.

Abstract

This Master's Thesis addresses the challenge of unplanned downtime in the packaging industry through the design, development, and validation of a predictive maintenance platform based on Artificial Intelligence (AI) for Industry 4.0. The main objective was to anticipate failures in powder drink packaging machines up to three days in advance. The methodology was based on the acquisition of real-time multivariate data (vibration, temperature, current, etc.) through sensors and PLCs, which are transmitted to an AWS cloud ecosystem. For the analysis of these time series, a Long Short-Term Memory (LSTM) recurrent neural network model was implemented, trained with historical data, and balanced using the SMOTE technique. The results showed the high effectiveness of the model, achieving an overall accuracy of 96% and a recall of 99% for fault detection, thereby minimizing the occurrence of unexpected shutdowns. The economic validation, through a cost analysis, projected a 75.66% reduction in operating expenses associated with failures, representing savings of over 1.8 million dollars. It is concluded that the solution is technically robust and offers a high return on investment, validating the use of AI to optimize efficiency and competitiveness in manufacturing.

Keywords: Predictive Maintenance, Industry 4.0, Artificial Intelligence, LSTM, Time Series.

Índice de contenidos

[1. Introducción 1](#_Toc202213535)

[1.1. Motivación 1](#_Toc202213536)

[1.2. Planteamiento del trabajo 3](#_Toc202213537)

[1.3. Estructura del trabajo 4](#_Toc202213538)

[2. Contexto y estado del arte 5](#_Toc202213539)

[2.1. Contexto del problema 5](#_Toc202213540)

[2.2. Estado del arte 8](#_Toc202213541)

[2.3. Conclusiones 11](#_Toc202213542)

[3. Objetivos concretos y metodología de trabajo 12](#_Toc202213543)

[3.1. Objetivo general 12](#_Toc202213544)

[3.2. Objetivos específicos 14](#_Toc202213545)

[3.3. Metodología del trabajo 15](#_Toc202213546)

[4. Descripción Detallada del Experimento 17](#_Toc202213547)

[4.1. Introducción al Diseño Experimental 17](#_Toc202213548)

[4.2. Tecnologías y Herramientas Empleadas 18](#_Toc202213549)

[4.3. Organización y Fases del Piloto Experimental 22](#_Toc202213550)

[4.4. Participantes y Análisis Estadístico 28](#_Toc202213551)

[5. Descripción de los resultados 29](#_Toc202213552)

[5.1. Introducción a la Presentación de Resultados 29](#_Toc202213553)

[5.2. Resultados del Análisis Exploratorio de Datos (EDA) 29](#_Toc202213554)

[5.3. Resultados del Entrenamiento del Modelo 36](#_Toc202213555)

[5.4. Resultados de la Evaluación del Modelo en el Conjunto de Prueba 38](#_Toc202213556)

[6. Discusión 42](#_Toc202213557)

[6.1. Introducción a la Discusión 42](#_Toc202213558)

[6.2. Interpretación Profunda de los Hallazgos del Análisis Exploratorio (eda) 42](#_Toc202213559)

[6.3. Discusión Sobre la Arquitectura y el Rendimiento del Modelo LTSM 43](#_Toc202213560)

[6.4 Análisis Crítico de las Métricas de Evaluación 45](#_Toc202213561)

[6.4. Discusión Sobre las Limitaciones y Validez del Experimento 46](#_Toc202213562)

[7. Conclusiones y trabajo futuro 47](#_Toc202213563)

[7.1. Conclusiones 47](#_Toc202213564)

[7.2. Síntesis De Las Contribuciones Del Trabajo 47](#_Toc202213565)

[7.3. Reflexión Sobre La Consecución De Objetivos 48](#_Toc202213566)

[7.4. Líneas De Trabajo Futuro Y Perspectivas De Evolución 49](#_Toc202213567)

[Referencias bibliográficas 54](#_Toc202213568)

[Anexo A. Código fuente y datos analizados 56](#_Toc202213569)

[Anexo B. Índice de acrónimos 56](#_Toc202213570)

Índice de figuras

[**Figura 1.** *Arquitectura Modelo LTSM* 24](#_Toc202213571)

[Figura 2. *Evolución de Vibración RMS* 29](#_Toc202213572)

[**Figura 3.** *Evolución de Temperatura Motor* 30](#_Toc202213573)

[**Figura 4.** *Evolución de Consumo Corriente* 30](#_Toc202213574)

[**Figura 5.** *Evolución de Presión Neumatica* 31](#_Toc202213575)

[**Figura 6.** *Distribución de vibración RMS* 32](#_Toc202213576)

[**Figura 7.** *Distribución de Temperatura Motor* 33](#_Toc202213577)

[**Figura 8.** *Distribución Consumo de Corriente y Presión Neumática* 34](#_Toc202213578)

[**Figura 9.** *Matriz de Correlación* 35](#_Toc202213579)

[**Figura 10.** *Curvas de Entrenamiento y Validación* 36](#_Toc202213580)

[**Figura 11.** *Métricas del Modelo* 37](#_Toc202213581)

[**Figura 12.** *Evaluación de Conjunto de Prueba* 38](#_Toc202213582)

[**Figura 13.** *Matriz de Confusión* 39](#_Toc202213583)

[**Figura 14.** *Curva Característica Operativa de Receptor* 40](#_Toc202213584)

[**Figura 15.** *Arquitectura de Infraestructura de Componentes* 49](#_Toc202213585)

[**Figura 16.** *Dashboard de Control Estado OK* 50](#_Toc202213586)

[**Figura 17.** *Dashboard de Control Estado Parada* 51](#_Toc202213587)

[**Figura 17.** *Dashboard de Control Estado Predicción Falla* 51](#_Toc202213588)

[**Figura 19.** *Correo de Predicción de Falla* 52](#_Toc202213589)

Índice de tablas

[**Tabla 1.** *análisis de Costos.* 7](#_Toc202213590)

# Introducción

La cuarta revolución industrial, o Industria 4.0, está redefiniendo los paradigmas de la manufactura a nivel global. La convergencia de tecnologías como el Internet de las Cosas (IoT), el Big Data y la Inteligencia Artificial (IA) está permitiendo a las empresas optimizar sus procesos productivos, reducir costos y aumentar su competitividad de maneras antes inimaginables. Uno de los campos donde esta transformación está generando mayor impacto es en el mantenimiento industrial, evolucionando desde un enfoque reactivo (reparar tras el fallo) hacia uno proactivo y, finalmente, predictivo. El mantenimiento predictivo, en particular, promete anticipar los fallos de la maquinaria antes de que ocurran, permitiendo intervenciones planificadas que maximizan la disponibilidad de los activos y minimizan las interrupciones operativas.

Este Trabajo de Fin de Máster se sitúa en este contexto de innovación, abordando un problema crítico y recurrente en la industria de consumo masivo: las paradas de producción no planificadas.

## Motivación

En la era actual, la industria manufacturera global se encuentra en medio de una profunda transformación impulsada por la Cuarta Revolución Industrial, o Industria 4.0. Este paradigma se define por la convergencia de tecnologías digitales avanzadas, como el Internet de las Cosas (IoT), la computación en la nube (Cloud Computing) y la Inteligencia Artificial (IA), con los procesos de producción tradicionales.1 El objetivo fundamental de esta integración es la creación de "fábricas inteligentes" capaces de operar con niveles de eficiencia, flexibilidad y autonomía sin precedentes. La capacidad de recopilar y analizar datos masivos en tiempo real desde el corazón de las operaciones está redefiniendo los límites de la optimización industrial.

A pesar de estos avances, uno de los desafíos más persistentes y costosos para el sector sigue siendo la gestión de las paradas de producción no planificadas. Estas interrupciones, a menudo causadas por fallos imprevistos en la maquinaria, no son meros inconvenientes técnicos; representan una amenaza directa a la rentabilidad y competitividad de una empresa. Una parada inesperada desencadena una cascada de consecuencias negativas que se extienden mucho más allá de la línea de producción: se generan cuellos de botella que paralizan operaciones aguas abajo, se incumple la planificación de la producción, se agotan los inventarios y, en última instancia, se producen retrasos en las entregas a los clientes.4 El impacto financiero es severo, abarcando no solo la pérdida de ingresos por productos no fabricados, sino también los costes de reparaciones de emergencia, el pago de mano de obra inactiva y el potencial daño a la reputación de la marca.6

Este problema es particularmente agudo en el sector de Bienes de Consumo de Rápido Movimiento (FMCG/CPG, por sus siglas en inglés), donde los márgenes son ajustados y la demanda es constante. En este contexto se sitúa el caso de estudio que motiva el presente trabajo: la empresa multinacional XYZ. Con una sólida presencia en 11 países de Latinoamérica, XYZ es un actor principal en la manufactura de productos de consumo masivo, destacando su línea de refrescos en polvo. Sus plantas de producción operan en un régimen de 24/7 para satisfacer la demanda del mercado, sometiendo a su maquinaria, en particular a las máquinas envasadoras industriales, a un estrés operativo constante. Esta alta exigencia resulta en una tasa de fallos elevada y, consecuentemente, en paradas inoportunas que erosionan la eficiencia y la competitividad de la compañía. La necesidad de una solución que trascienda los enfoques de mantenimiento tradicionales es, por tanto, no solo una oportunidad de mejora, sino una necesidad estratégica.

## Planteamiento del trabajo

Para abordar la problemática crítica de las paradas no planificadas en la empresa XYZ, este Trabajo Fin de Máster (TFM) propone el diseño, la validación y la documentación de "Industr\_IA", una plataforma integral de mantenimiento predictivo (PdM) enmarcada en los principios de la Industria 4.0. La propuesta va más allá del desarrollo de un mero modelo de clasificación; se concibe como un ecosistema tecnológico completo y holístico que abarca todo el ciclo de vida del dato, desde su origen en la planta de producción hasta su conversión en inteligencia accionable.

El núcleo de la solución es la implementación de un sistema capaz de anticipar fallos en las máquinas envasadoras con una antelación de hasta tres días. Este sistema se fundamenta en un modelo de Inteligencia Artificial, específicamente una red neuronal de tipo Long Short-Term Memory (LSTM), entrenado para reconocer patrones sutiles en los datos de los sensores que preceden a un fallo.

La plataforma está diseñada para:

* Adquirir datos en tiempo real desde el edge, es decir, directamente desde las máquinas en la planta, utilizando sensores críticos y Controladores Lógicos Programables (PLC).
* Transmitir estos datos de forma segura y eficiente a la nube de Amazon Web Services (AWS) para su procesamiento centralizado.
* Analizar las secuencias de datos mediante el modelo de IA para generar una probabilidad de fallo en tiempo real.
* Entregar inteligencia accionable al personal de mantenimiento a través de un sistema de alertas proactivas y dashboards de monitorización intuitivos.

El objetivo final es facilitar una transición estratégica desde un modelo de mantenimiento reactivo (reparar después del fallo) y preventivo (basado en calendarios) hacia un paradigma puramente predictivo, donde las intervenciones se planifican con base en la condición real y futura del activo. Esta transición busca optimizar el uso de recursos, minimizar drásticamente el tiempo de inactividad no planificado y, en consecuencia, potenciar la productividad y la rentabilidad de la operación.1

## Estructura del trabajo

El presente documento se organiza en varios capítulos para guiar al lector de manera lógica y estructurada a través del desarrollo del proyecto, de acuerdo con las directrices académicas establecidas.8

* **Capítulo 2: Contexto y Estado del Arte.** Se realiza un análisis profundo del problema de negocio, cuantificando el impacto del tiempo de inactividad, y se revisa la literatura científica sobre mantenimiento industrial y las tecnologías de IA aplicadas a series temporales, con especial énfasis en las arquitecturas LSTM.
* **Capítulo 3: Objetivos y Metodología de Trabajo.** Se definen el objetivo general y los objetivos específicos del proyecto, y se describe la metodología seguida para su consecución, combinando enfoques de gestión de proyectos de ciencia de datos y desarrollo de software.
* **Capítulo 4: Desarrollo de la Solución: Adquisición y Análisis de Datos.** Se detalla la arquitectura de adquisición de datos en la planta, el análisis exploratorio de los datos de los sensores y los pasos de preprocesamiento aplicados para preparar el dataset para el modelado.
* **Capítulo 5: Desarrollo de la Solución: Diseño y Entrenamiento del Modelo Predictivo.** Se presenta la arquitectura del modelo LSTM, justificando cada capa y parámetro, y se describe el proceso de entrenamiento, optimización y evaluación técnica del mismo.
* **Capítulo 6: Puesta en Producción y Validación del Negocio.** Se expone la arquitectura de infraestructura *serverless* en AWS para el despliegue de la solución, se muestra el sistema de monitorización con Grafana y se realiza una validación económica del proyecto mediante un análisis de retorno de la inversión (ROI).
* **Capítulo 7: Conclusiones y Trabajo Futuro.** Se resumen las contribuciones del trabajo, se verifica el cumplimiento de los objetivos y se proponen futuras líneas de investigación y mejora para la plataforma Industr\_IA.

# Contexto y estado del arte

## Contexto del problema

Para justificar de manera rigurosa la necesidad y la viabilidad económica del proyecto Industr\_IA, es imperativo realizar un análisis cuantitativo del impacto que las paradas no planificadas tienen en el negocio de la empresa XYZ. Esta sección profundiza dicho análisis basándose en los parámetros operativos del proyecto y en benchmarks de la industria.

**Parámetros Operativos de Referencia**

La máquina envasadora de sobres de refresco en polvo opera a una velocidad de producción objetivo de 100 Paquetes Por Minuto (PPM). Esto se traduce en una capacidad de producción horaria de:

*100 PPM×60 min/h=6,000 paquetes/hora*

Esta cifra se alinea con las capacidades de maquinaria industrial de envasado, que pueden manejar desde cientos hasta miles de paquetes por hora, dependiendo de la automatización y el producto.9

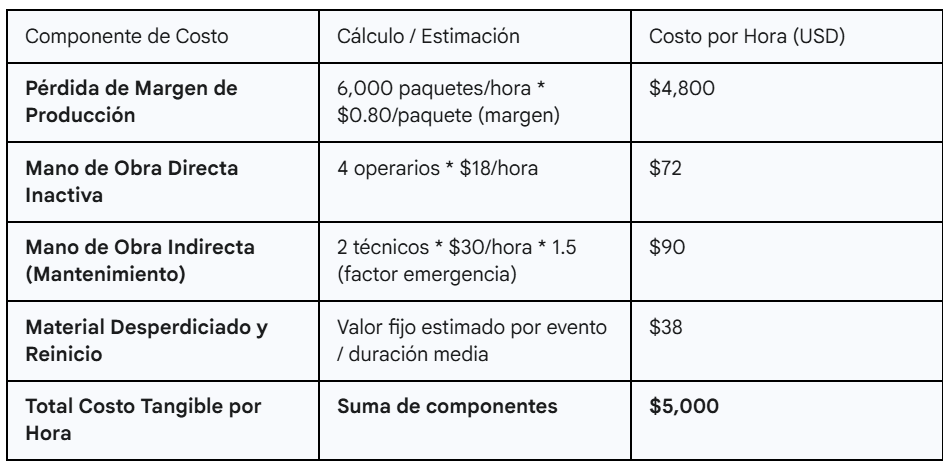
**Análisis Detallado de Costos de una Parada No Planificada**

El costo de una hora de inactividad no es una cifra monolítica, sino el resultado de la suma de múltiples factores, tanto tangibles como intangibles. Diversos estudios sitúan el costo promedio de una hora de parada no planificada en la industria CPG/FMCG en cifras que van desde $23,600 hasta rangos de €40,000-€200,000, dependiendo de la escala de la operación.7 A continuación, se desglosa este costo para el caso específico de la empresa XYZ.

* **Pérdida de Producción y Margen de Contribución:** El costo más directo es el valor de los bienes que no se producen. Asumiendo un precio de venta promedio por caja de producto (que contiene múltiples sobres) y un costo variable asociado, se puede estimar el margen de contribución perdido por cada hora de inactividad.
* **Costos de Mano de Obra Inactiva:** Durante una parada, el personal asignado a la línea de producción no puede realizar su trabajo, pero sus salarios continúan siendo un costo para la empresa. Esto incluye a los operarios de línea y al personal de supervisión. Basándonos en datos salariales de la industria alimentaria, un operario de línea puede tener un costo horario de entre $16 y $18, mientras que un técnico de mantenimiento cualificado puede costar entre $22 y $37 por hora en condiciones normales.13
* **Costos de Reparación de Emergencia:** Las reparaciones no planificadas suelen ser más costosas. Implican el pago de horas extra al personal de mantenimiento, la posible compra de repuestos a precios elevados por la urgencia y, en ocasiones, la contratación de servicios técnicos externos especializados.12
* **Costos Intangibles:** Aunque difíciles de cuantificar, estos costos son igualmente perjudiciales. Incluyen la erosión de la moral del equipo de producción, el riesgo de incumplir acuerdos de nivel de servicio (SLA) con grandes distribuidores, y el daño a la reputación de la marca XYZ por falta de stock en los puntos de venta.6

La combinación de estos factores demuestra que el costo real de una parada no planificada es una función compuesta. La pérdida de ingresos es solo una parte de la ecuación; los costos laborales que se vuelven improductivos y los sobrecostos de las reparaciones de emergencia magnifican el impacto financiero. La siguiente tabla resume una estimación conservadora de estos costos para una de las envasadoras de XYZ.

Tabla 1. *análisis de Costos.*



*Nota: Esta tabla presenta una estimación conservadora. El costo real puede ser significativamente mayor dependiendo de la criticidad de la línea y los contratos con clientes. La cifra de $10,000 por parada no planificada utilizada en la validación del modelo se considera un promedio que incluye eventos de mayor y menor impacto.*

Este análisis de negocio no es un mero ejercicio académico; representa una aproximación fiel al problema financiero que la empresa XYZ busca resolver. La magnitud de estos costos justifica de manera contundente la inversión en una plataforma de mantenimiento predictivo como Industr\_IA, cuyo objetivo es transformar estos costos imprevistos y elevados en gastos de mantenimiento planificados, controlados y significativamente menores.

## Estado del arte

**Paradigmas de Mantenimiento y Soluciones de IA**

La disciplina del mantenimiento industrial ha experimentado una notable evolución, pasando de ser una función puramente reactiva a una pieza estratégica en la optimización de la producción. Esta evolución puede segmentarse en varios paradigmas 1:

* **Mantenimiento Reactivo (Run-to-Failure):** La estrategia más básica, que consiste en reparar los equipos únicamente cuando se produce un fallo. Aunque minimiza los costos de mantenimiento a corto plazo, maximiza el impacto de las paradas no planificadas.
* **Mantenimiento Preventivo:** Se basa en la realización de intervenciones a intervalos fijos, ya sea por tiempo (e.g., cada 6 meses) o por uso (e.g., cada millón de ciclos). Reduce la probabilidad de fallos inesperados, pero puede llevar a un sobre mantenimiento, reemplazando componentes que aún son funcionales.
* **Mantenimiento Basado en la Condición (CBM):** Monitoriza el estado de los equipos en tiempo real y activa las tareas de mantenimiento cuando se superan ciertos umbrales predefinidos. Es un paso intermedio hacia el mantenimiento predictivo.
* **Mantenimiento Predictivo (PdM):** Utiliza datos históricos y en tiempo real para predecir cuándo ocurrirá un fallo. Esto permite planificar las intervenciones justo antes de que el fallo se materialice, maximizando la vida útil de los componentes y minimizando el tiempo de inactividad.
* **Mantenimiento Prescriptivo:** La frontera más avanzada, que no solo predice el fallo, sino que también recomienda acciones específicas para mitigar el riesgo o optimizar la operación dadas las circunstancias.

El proyecto Industr\_IA se enmarca firmemente en el paradigma del Mantenimiento Predictivo, utilizando los pilares tecnológicos de la Industria 4.0 (IoT, Cloud, IA) para su implementación.

**Revisión de Modelos de Deep Learning para Series Temporales**

La predicción de fallos a partir de datos de sensores es, en esencia, un problema de análisis de series temporales multivariadas. Las Redes Neuronales de Deep Learning han demostrado ser excepcionalmente eficaces en esta tarea, superando a los métodos estadísticos tradicionales.

Las **Redes Neuronales Recurrentes (RNN)** fueron diseñadas específicamente para procesar datos secuenciales. Su arquitectura les permite mantener una "memoria" de la información de pasos anteriores para influir en la salida actual. Sin embargo, las RNN simples sufren del problema del "desvanecimiento del gradiente" (vanishing gradient), que les dificulta aprender dependencias a largo plazo en secuencias largas.

Para solucionar esta limitación, se introdujeron las redes **Long Short-Term Memory (LSTM)**.17 Las LSTMs son un tipo avanzado de RNN que incorpora una estructura de "compuertas" (gates) dentro de su celda de memoria. Estas compuertas —la compuerta de olvido, la compuerta de entrada y la compuerta de salida— regulan el flujo de información, permitiendo a la red aprender qué información es relevante para mantener a largo plazo y cuál debe ser descartada. Esta capacidad es fundamental para el mantenimiento predictivo, donde un fallo puede ser el resultado de una degradación lenta y sutil a lo largo del tiempo, cuyos indicadores iniciales aparecieron muchos pasos temporales atrás.19

La literatura académica sobre la aplicación de LSTMs en mantenimiento predictivo es extensa y muestra una clara trayectoria evolutiva. Los trabajos en este campo se pueden agrupar en varias categorías temáticas:

* **Estudios Fundacionales y Revisiones Sistemáticas:** Publicaciones que sientan las bases del campo y analizan el estado del arte, confirmando la creciente importancia de la IA y el PdM en la Industria 4.0.1 Estos trabajos destacan que el uso de datos reales y metodologías de Machine Learning es la tendencia predominante.
* **LSTM para Estimación de Vida Útil Remanente (RUL):** Una de las aplicaciones más comunes es utilizar LSTMs para predecir el RUL, es decir, el tiempo restante antes de que un componente o sistema falle. Estos modelos suelen ser de regresión y se entrenan con datos de "run-to-failure".17 La capacidad de las LSTMs para manejar secuencias temporales las hace ideales para modelar la degradación progresiva de los equipos.26
* **Arquitecturas Híbridas (CNN-LSTM, Autoencoder-LSTM):** Los investigadores pronto se dieron cuenta de que, si bien las LSTMs son expertas en dependencias temporales, otras arquitecturas como las Redes Neuronales Convolucionales (CNN) son mejores para extraer características de las señales crudas (que pueden verse como secuencias 1D). Esto llevó al desarrollo de modelos híbridos CNN-LSTM, donde una CNN primero extrae características espaciales o de frecuencia de las señales de los sensores, y un LSTM luego procesa estas características en el tiempo.18 De manera similar, los Autoencoders-LSTM se utilizan para la detección de anomalías, aprendiendo una representación comprimida de la operación normal y detectando desviaciones.30
* **Modelos con Mecanismos de Atención (Attention-based LSTM):** Un avance más reciente es la incorporación de mecanismos de atención. Estos mecanismos permiten que el modelo aprenda a ponderar la importancia de diferentes pasos temporales o diferentes sensores (atributos) al hacer una predicción. En lugar de tratar toda la información de la ventana de tiempo por igual, el modelo puede "prestar atención" a los momentos o variables más críticos que preceden a un fallo, mejorando tanto la precisión como la interpretabilidad del modelo.26
* **Aplicaciones Industriales Específicas:** Numerosos casos de estudio demuestran la efectividad de las LSTMs en entornos de manufactura reales, aplicados a una amplia gama de equipos como turbinas, rodamientos, bombas y sistemas de compresión, validando su robustez y aplicabilidad práctica.19

El modelo desarrollado en este TFM, una arquitectura de LSTMs apiladas (stacked LSTM), se posiciona como una solución robusta y bien establecida. Es un paso más allá de un LSTM simple, ya que permite aprender jerarquías de características temporales: la primera capa aprende patrones de bajo nivel y la segunda aprende patrones de más alto nivel a partir de la salida de la primera. Aunque no implementa explícitamente CNNs o mecanismos de atención, su diseño es una elección pragmática y potente, justificada por la necesidad de capturar las complejas interacciones no lineales entre las múltiples variables de los sensores, como se describe en la propuesta del proyecto.

## Conclusiones

El análisis de la literatura científica y técnica confirma que la aproximación adoptada en este TFM es metodológicamente sólida y se alinea con las mejores prácticas actuales en el campo del mantenimiento predictivo. La elección de una arquitectura basada en LSTM para analizar series temporales multivariadas de sensores con el fin de clasificar el estado de la máquina ("normal" vs. "fallo inminente") está ampliamente respaldada por la investigación.17

La selección de un modelo LSTM apilado representa un equilibrio adecuado entre la capacidad de modelado y la complejidad de implementación para un proyecto de esta envergadura. Si bien arquitecturas más avanzadas como los Transformers o los modelos híbridos con atención existen y representan una prometedora línea de trabajo futuro, el modelo propuesto es más que suficiente para demostrar la viabilidad y el valor de la plataforma Industr\_IA. El estado del arte no solo valida la elección tecnológica, sino que también proporciona un marco para contextualizar los resultados y proponer futuras mejoras.

# Objetivos concretos y metodología de trabajo

Este capítulo sirve como puente entre el estudio del dominio y la contribución específica del trabajo. Se establecen los objetivos concretos que guiarán el desarrollo y se describe la metodología empleada para alcanzarlos, asegurando un enfoque estructurado y riguroso.

## Objetivo general

El objetivo general de este proyecto es diseñar, desarrollar, validar y documentar una plataforma integral de Inteligencia Artificial, denominada "Industr\_IA", concebida específicamente para optimizar el mantenimiento de máquinas envasadoras industriales a través de un enfoque predictivo. Este objetivo se desglosa en las siguientes fases y metas clave:

* **Diseño Conceptual y Arquitectónico de la Plataforma "Industr\_IA":**
  + Definir la arquitectura de software y hardware necesaria para la captura, almacenamiento, procesamiento y visualización de datos en tiempo real, provenientes de los sensores de las máquinas envasadoras.
  + Seleccionar las tecnologías, lenguajes de programación y *frameworks* más adecuados para la construcción de un sistema robusto, escalable y seguro.
  + Diseñar una interfaz de usuario (UI) y experiencia de usuario (UX) intuitiva que permita a los técnicos de mantenimiento y jefes de planta interactuar fácilmente con los diagnósticos y predicciones del sistema.
* **Desarrollo del Núcleo de Inteligencia Artificial:**
  + Implementar algoritmos de *Machine Learning* y *Deep Learning* capaces de analizar datos históricos y en tiempo real (vibración, temperatura, presión, velocidad, etc.) para identificar patrones y anomalías que preceden a las fallas.
  + Construir y entrenar modelos predictivos que estimen con la mayor precisión posible el Tiempo Restante de Vida Útil (*Remaining Useful Life* - RUL) de los componentes críticos de las máquinas.
* **Validación Técnica y Demostración de Viabilidad:**
  + Evaluar rigurosamente el rendimiento de los modelos de IA utilizando un conjunto de datos de prueba independiente. La viabilidad técnica se demostrará cuantitativamente a través de métricas estándar como la Precisión (*Accuracy*), Sensibilidad (*Recall*), F1-Score y el Error Cuadrático Medio (*Root Mean Square Error* - RMSE) para las predicciones de RUL.
  + Realizar pruebas funcionales y de integración de la plataforma completa en un entorno simulado o en una línea de producción piloto para asegurar su estabilidad y correcto funcionamiento.
* **Análisis del Impacto Económico y Retorno de la Inversión (ROI):**
  + Demostrar el valor económico tangible de la plataforma "Industr\_IA". Esto se logrará mediante un análisis de Retorno de la Inversión (ROI) detallado, que cuantificará el impacto financiero positivo.
  + El análisis se basará principalmente en la reducción proyectada de los costos asociados a las paradas no planificadas, incluyendo el costo de la mano de obra de emergencia, el valor de la producción perdida y los posibles sobrecostos por reparaciones urgentes.
* **Documentación Integral del Proyecto:**
  + Generar una documentación técnica completa que abarque la arquitectura del sistema, el código fuente, los modelos de IA implementados y las APIs desarrolladas.
  + Crear manuales de usuario y guías de operación que faciliten la adopción, uso y mantenimiento de la plataforma por parte del personal de la planta.

En resumen, el proyecto no solo busca crear una herramienta tecnológica avanzada, sino validar de forma holística su aplicación práctica en un entorno industrial, probando que es técnicamente eficaz para predecir fallos y económicamente rentable al minimizar las costosas interrupciones de producción.

## Objetivos específicos

Para la consecución del objetivo general, se han definido los siguientes objetivos específicos, formulados siguiendo los criterios SMART (Específicos, Medibles, Alcanzables, Relevantes y con Plazo):

* **Analizar y cuantificar el impacto operativo y financiero.** Investigar y modelar los costos asociados a las paradas no planificadas en el contexto de la empresa XYZ para establecer una línea base de negocio clara y medible que justifique el proyecto.
* **Diseñar e implementar la adquisición de datos.** Desarrollar una arquitectura de adquisición de datos robusta en la planta de producción, utilizando Controladores Lógicos Programables (PLC) y redes inalámbricas industriales para la recolección sistemática y en tiempo real de datos de los sensores seleccionados.
* **Desarrollar y entrenar el modelo predictivo.** Construir, entrenar y optimizar un modelo de Deep Learning basado en redes LSTM que sea capaz de clasificar el estado de la máquina y predecir la probabilidad de un fallo inminente a partir de secuencias de datos de sensores multivariados.
* **Diseñar y desplegar la infraestructura en la nube.** Implementar una arquitectura de infraestructura *serverless*, escalable y orientada a eventos en la plataforma AWS, que soporte la ingesta, el procesamiento, la inferencia en tiempo real y el almacenamiento persistente de los datos y predicciones.
* **Validar la solución y desarrollar la visualización.** Evaluar la efectividad del modelo predictivo tanto a nivel técnico (utilizando métricas estándar de Machine Learning como precisión, recall y AUC) como a nivel de negocio (calculando el ahorro de costos y el ROI). Además, desarrollar un sistema de monitorización y alerta mediante dashboards interactivos en Grafana.

## Metodología del trabajo

Dada la naturaleza dual del proyecto, que combina la investigación aplicada con el desarrollo de una solución tecnológica destinada a un entorno de producción, se ha optado por una metodología de trabajo híbrida. Esta metodología integra dos marcos de trabajo estándar en la industria:

* **CRISP-DM (Cross-Industry Standard Process for Data Mining):** Se utilizará como guía para las fases de desarrollo del modelo de IA. CRISP-DM proporciona un ciclo de vida estructurado que asegura que el proyecto esté bien definido y orientado a los objetivos del negocio. Las fases que se seguirán son:
* **Comprensión del Negocio:** Cubierta en los Capítulos 1 y 2.1, donde se define el problema y los objetivos desde una perspectiva empresarial.
* **Comprensión de los Datos:** Realizada a través del Análisis Exploratorio de Datos (EDA) en el Capítulo 4.2.
* **Preparación de los Datos:** Detallada en el Capítulo 4.3, incluyendo la estandarización, la creación de secuencias y el balanceo de clases.
* **Modelado:** Descrita en el Capítulo 5, donde se diseña, entrena y ajusta el modelo LSTM.
* **Evaluación:** Presentada en los Capítulos 5.3 y 6.3, donde se evalúa el modelo técnica y económicamente.
* **Principios de MLOps (Machine Learning Operations):** Para la fase de despliegue y mantenimiento de la solución, se adoptarán principios de MLOps. Este enfoque extiende las prácticas de DevOps al ciclo de vida del Machine Learning, enfatizando la automatización, la monitorización y la gestión continua de los modelos en producción. Las fases relevantes son:
* **Despliegue (Deployment):** Detallado en el Capítulo 6.1, con la implementación de la arquitectura en AWS.
* **Monitorización (Monitoring):** Cubierta en los Capítulos 6.2 y 6.4, abarcando tanto la visualización del estado de la máquina como la monitorización del rendimiento del modelo (Model Drift).
* **Mantenimiento (Maintenance):** Discutido en el Capítulo 6.4, donde se plantea la estrategia de re-entrenamiento periódico para asegurar la relevancia del modelo a lo largo del tiempo.

La combinación de estos dos marcos de trabajo proporciona la estructura necesaria para un desarrollo riguroso desde la concepción del problema hasta la implementación y mantenimiento de una solución de IA robusta y sostenible en un entorno industrial real.

1. Descripción Detallada del Experimento

## Introducción al Diseño Experimental

El presente capítulo constituye el núcleo técnico de este Trabajo de Fin de Máster. Su propósito es describir con un nivel de detalle exhaustivo el piloto experimental diseñado y ejecutado para validar la hipótesis central del proyecto: la viabilidad y eficacia de un modelo de inteligencia artificial, específicamente una red neuronal de tipo Long Short-Term Memory (LSTM), para el mantenimiento predictivo en un entorno industrial simulado.

El enfoque adoptado es el de un piloto experimental, una metodología que permite evaluar una innovación tecnológica en un entorno controlado antes de su despliegue a gran escala. Este enfoque es particularmente pertinente en el ámbito de la Industria 4.0, donde la inversión en nuevas tecnologías debe ser justificada mediante evidencia empírica sólida. El experimento se centra en la capacidad del sistema para predecir fallos en una máquina envasadora de refrescos en polvo, utilizando para ello un conjunto de datos sintético pero realista que emula las lecturas de sensores en un entorno de producción.

Este capítulo se estructura para guiar al lector a través de cada una de las etapas del experimento, desde la justificación de las herramientas tecnológicas seleccionadas hasta la descripción del flujo de ejecución del piloto. Se detallarán las características del conjunto de datos, las técnicas de preprocesamiento aplicadas y la arquitectura del modelo predictivo implementado, sentando así las bases para la presentación y discusión de los resultados en los capítulos subsiguientes.

## Tecnologías y Herramientas Empleadas

La selección del ecosistema tecnológico es un pilar fundamental en cualquier proyecto de ciencia de datos. Para "Industr\_IA", la elección se ha guiado por criterios de robustez, flexibilidad, escalabilidad y, de manera crucial, por el amplio respaldo de la comunidad de código abierto, lo que garantiza una continua actualización y un soporte extensivo.

**Entorno de Desarrollo: Google Colaboratory (Colab)**

El proyecto se desarrolló íntegramente en Google Colaboratory, una plataforma en la nube basada en Jupyter Notebooks. La elección de Colab no fue casual y se justifica por varias ventajas estratégicas:

* **Acceso a Hardware de Alto Rendimiento**: Colab proporciona acceso gratuito a Unidades de Procesamiento Gráfico (GPU), específicamente la NVIDIA T4 utilizada en este proyecto. El entrenamiento de modelos de aprendizaje profundo como las LSTM es una tarea computacionalmente intensiva. El uso de una GPU permite paralelizar las operaciones matriciales inherentes a las redes neuronales, reduciendo drásticamente los tiempos de entrenamiento de horas a minutos. Esto fue crucial para poder iterar rápidamente en el diseño y ajuste de hiperparámetros del modelo.
* **Preconfiguración y Portabilidad**: El entorno de Colab viene preconfigurado con las principales librerías de ciencia de datos (Pandas, NumPy, Scikit-learn, TensorFlow), eliminando la complejidad de la gestión de dependencias y asegurando la reproducibilidad del experimento en cualquier máquina con acceso a un navegador web.
* **Colaboración y Documentación**: La naturaleza de los notebooks permite combinar código ejecutable, texto enriquecido, ecuaciones y visualizaciones en un único documento, facilitando una documentación clara y una presentación lógica del flujo de trabajo.

**Lenguaje de Programación Python 3**

Python se ha consolidado como el lenguaje de facto para la inteligencia artificial y la ciencia de datos. Su sintaxis clara y legible, junto con un vasto ecosistema de librerías especializadas, lo convirtieron en la elección obvia para este proyecto.

**Librerías Fundamentales para el Análisis de Datos**

* **Pandas**: Esta librería fue la espina dorsal para toda la manipulación de datos. Se utilizó para cargar el conjunto de datos desde el archivo ds\_envasadoras.csv a un DataFrame, una estructura de datos tabular y optimizada. Funcionalidades como pd.read\_csv(), .info(), .describe(), y .isnull().sum() fueron esenciales en la fase inicial de inspección para entender la estructura, tipos de datos, estadísticas descriptivas y la presencia de valores nulos.
* **NumPy**: Subyacente a Pandas y a muchas otras librerías, NumPy proporcionó el soporte para los arrays multidimensionales y las operaciones matemáticas de alto rendimiento. Fue fundamental en la fase de preprocesamiento para la transformación de los DataFrames de Pandas en los arrays numéricos que TensorFlow requiere como entrada.
* **Matplotlib y Seaborn**: Constituyeron el dúo principal para la visualización de datos estática. Matplotlib, por su flexibilidad y control granular, se usó para crear las gráficas de historial de entrenamiento. Seaborn, una capa de abstracción sobre Matplotlib, facilitó la creación de visualizaciones estadísticas más complejas y estéticamente agradables, como los gráficos de densidad de kernel (KDE) y las matrices de correlación (heatmap), que fueron vitales para el análisis exploratorio.
* **Plotly:** Para la visualización interactiva, se eligió Plotly. A diferencia de las librerías estáticas, Plotly genera gráficos en HTML y JavaScript, permitiendo al usuario hacer zoom, desplazarse y obtener información detallada al pasar el cursor sobre los puntos de datos. Esto fue especialmente útil para explorar las series temporales de los sensores, permitiendo una inspección detallada de las fluctuaciones de los datos justo antes de un evento de fallo.

**Librerías de Aprendizaje Automático y Modelado**

**Scikit-learn:** Esta librería es la navaja suiza del machine learning en Python. En este proyecto, su rol fue central en la fase de preprocesamiento:

* **StandardScaler**: Se utilizó para estandarizar las características. Este paso es crucial para los modelos de redes neuronales, ya que asegura que todas las características tengan una media de 0 y una desviación estándar de 1. Esto evita que las características con rangos de valores más amplios dominen el proceso de aprendizaje y ayuda a que el algoritmo de optimización (Adam) converja más rápidamente.
* **train\_test\_split**: Permitió dividir el conjunto de datos de manera estratificada en conjuntos de entrenamiento y prueba, asegurando que la proporción de fallos y operaciones normales fuera la misma en ambas divisiones, lo cual es vital para una evaluación no sesgada del modelo.

**Imbalanced-learn**: El problema del desequilibrio de clases es endémico en los conjuntos de datos de mantenimiento predictivo, donde los fallos son eventos raros. Esta librería proporcionó la implementación de la técnica SMOTE (Synthetic Minority Over-sampling Technique). SMOTE fue una pieza clave del experimento, ya que en lugar de simplemente duplicar las muestras de la clase minoritaria (fallos), crea nuevas muestras sintéticas interpolando entre las muestras existentes. Esto ayuda al modelo a aprender una frontera de decisión más robusta y generalizable, evitando el sobreajuste a las pocas muestras de fallo reales.

**TensorFlow y Keras**: TensorFlow, desarrollado por Google, es una de las plataformas de aprendizaje profundo más potentes y flexibles. Keras, que ahora es la API de alto nivel oficial de TensorFlow, se utilizó para definir la arquitectura del modelo LSTM. La sintaxis de Keras es modular e intuitiva, permitiendo construir modelos complejos apilando capas de manera secuencial (Sequential API). Se utilizaron las siguientes capas y componentes:

* **InputLayer**: Define la forma de los datos de entrada.
* **LSTM**: La capa principal del modelo, que contiene las unidades recurrentes con memoria a corto y largo plazo.
* **Dropout**: Una técnica de regularización fundamental para combatir el sobreajuste. Durante el entrenamiento, "apaga" aleatoriamente un porcentaje de las neuronas de la capa, forzando al modelo a aprender representaciones más robustas y menos dependientes de neuronas individuales.
* **Dense**: Capas de neuronas totalmente conectadas, utilizadas para procesar las representaciones aprendidas por las capas LSTM y para la capa de salida final.
* **ModelCheckpoint**: Un callback que guarda automáticamente el mejor modelo (según la métrica de val\_loss) durante el entrenamiento. Esto asegura que, independientemente del sobreajuste en las últimas épocas, se conserve el modelo con el mejor rendimiento en el conjunto de validación.
* **EarlyStopping**: Otro callback esencial para prevenir el sobreajuste. Monitoriza una métrica (de nuevo, val\_loss) y detiene el entrenamiento si no se observa una mejora después de un número determinado de épocas (patience), ahorrando tiempo computacional y evitando que el modelo se degrade.

## Organización y Fases del Piloto Experimental

El experimento se diseñó siguiendo una estructura metodológica rigurosa, dividida en fases lógicas y secuenciales que reflejan el ciclo de vida estándar de un proyecto de ciencia de datos.

**Fase 1: Adquisición y Comprensión de los Datos (Análisis Exploratorio - EDA)**

El punto de partida fue la carga del conjunto de datos ds\_envasadoras.csv. El objetivo de esta fase era obtener una comprensión profunda de los datos. Se realizaron las siguientes actividades:

* **Inspección Estructural**: Se utilizaron df.info() y df.head() para verificar los tipos de datos de cada columna (sensor), el número de entradas y la presencia de valores nulos. Se confirmó que el dataset estaba limpio y completo.
* **Análisis Estadístico Descriptivo**: Con df.describe(), se calcularon estadísticas clave (media, mediana, desviación estándar, cuartiles) para cada sensor. Esto proporcionó una primera visión de la escala y distribución de cada variable.
* **Análisis de la Variable Objetivo**: Se analizó la distribución de la variable machine\_status usando df['machine\_status'].value\_counts(). Esto reveló un severo desequilibrio de clases, un hallazgo crítico que condicionó el resto del diseño experimental.
* **Visualización de Datos**: Se generó un conjunto de gráficos (descritos en detalle en el Capítulo 5) para visualizar patrones, tendencias y relaciones. Esto incluyó gráficos de series temporales para cada sensor, histogramas y gráficos KDE para entender las distribuciones, y una matriz de correlación para identificar relaciones lineales entre sensores.

**Fase 2: Preprocesamiento y Preparación de los Datos**

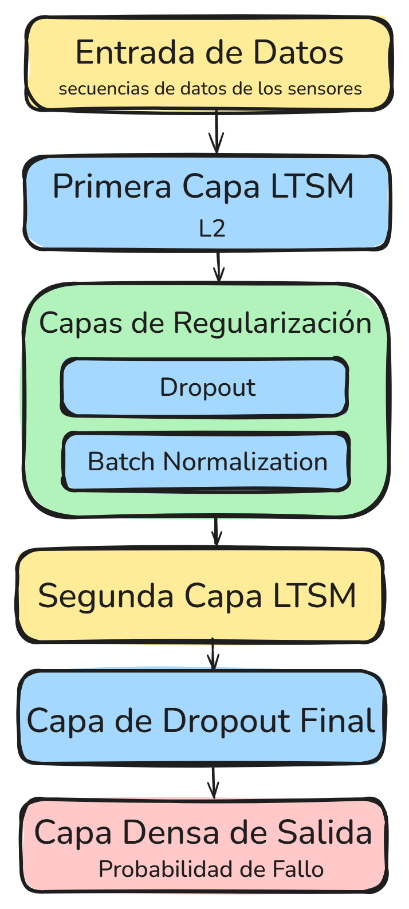
Esta fase fue crítica para transformar los datos brutos en un formato adecuado para el modelo LSTM.

* **Codificación de la Variable Objetivo**: La variable categórica machine\_status ('Normal', 'Recovering', 'Failure') se transformó a un formato numérico. Los estados 'Normal' y 'Recovering' se agruparon en una única clase 0 (operación no fallida) y 'Failure' se codificó como 1. Esta binarización simplificó el problema a una clasificación binaria.
* **Escalado de Características**: Todas las columnas de sensores se estandarizaron usando StandardScaler.
* **Creación de Secuencias Temporales**: Las redes LSTM no procesan puntos de datos individuales, sino secuencias de datos. Se implementó una función para transformar el conjunto de datos en secuencias superpuestas. Se definió un sequence\_length de 10, lo que significa que para predecir el estado en el tiempo t, el modelo observaría los datos de los sensores desde t-10 hasta t-1. Esta ventana temporal es un hiperparámetros crucial que dota al modelo de contexto histórico.
* **División de Datos**: El conjunto de datos secuenciado se dividió en un 80% para entrenamiento y un 20% para prueba (train\_test\_split), utilizando estratificación para mantener la proporción de fallos.
* **Balanceo de Clases con SMOTE**: El desequilibrio de clases se abordó aplicando SMOTE únicamente al conjunto de entrenamiento. Es fundamental no aplicar SMOTE al conjunto de prueba, ya que este debe reflejar la distribución real de los datos en un entorno de producción para obtener una evaluación honesta del rendimiento del modelo.

**Fase 3: Diseño y Construcción del Modelo LSTM**

En esta fase se definió la arquitectura de la red neuronal en Keras.

Figura 1. *Arquitectura Modelo LTSM*



* **Definición de Capas:** Se diseñó una arquitectura apilada (stacked LSTM), que consiste en múltiples capas LSTM. Esto permite al modelo aprender jerarquías de características temporales: las primeras capas aprenden patrones a corto plazo, y las capas más profundas combinan estos para aprender patrones a más largo plazo.
* La primera capa LSTM (return\_sequences=True) devuelve la secuencia completa de salidas para cada paso de tiempo, para que la siguiente capa LSTM pueda procesarla.
* La última capa LSTM (return\_sequences=False) solo devuelve la salida del último paso de tiempo, que resume la información de toda la secuencia.
* **Regularización**: Se incluyeron capas de Dropout después de cada capa LSTM para prevenir el sobreajuste.
* **Capa de Salida**: Se añadió una capa Dense final con una única neurona y una función de activación sigmoid. La función sigmoide produce una salida entre 0 y 1, que puede interpretarse como la probabilidad de que la secuencia de entrada corresponda a un fallo.
* **Compilación del Modelo**: Se compiló el modelo especificando:
* **optimizer='adam'**: El optimizador Adam es una elección estándar y robusta que adapta la tasa de aprendizaje para cada parámetro.
* **loss='binary\_crossentropy'**: La función de pérdida adecuada para problemas de clasificación binaria. Mide la distancia entre la distribución de probabilidad predicha y la real.

**Fase 4: Entrenamiento y Validación del Modelo**

El modelo compilado se entrenó usando el método .fit().

* **Proceso de Entrenamiento**: El modelo se entrenó con los datos de entrenamiento balanceados (X\_train\_smote, y\_train\_smote).
* **Validación Cruzada Implícita**: Se utilizó una porción de los datos de entrenamiento (20%) como conjunto de validación (validation\_split=0.2). El rendimiento en este conjunto de validación fue monitorizado por los callbacks ModelCheckpoint y EarlyStopping.
* **Hiperparámetros de Entrenamiento**: Se definieron epochs=50 (número máximo de pasadas por el dataset) y batch\_size=64 (número de secuencias procesadas antes de actualizar los pesos del modelo).

**Fase 5: Evaluación del Rendimiento**

Una vez finalizado el entrenamiento, se cargó el mejor modelo guardado por ModelCheckpoint y se evaluó su rendimiento en el conjunto de prueba (X\_test, y\_test), que el modelo nunca había visto antes.

* **Predicción**: Se generaron las predicciones de probabilidad en el conjunto de prueba.
* **Umbral de Decisión**: Las probabilidades se convirtieron en clases binarias (0 o 1) usando un umbral de 0.3.
* **Cálculo de Métricas**: Se calculó un conjunto completo de métricas de clasificación (classification\_report de Scikit-learn), incluyendo precisión, recall y F1-score para cada clase.
* **Visualización de la Evaluación**: Se generaron la matriz de confusión, la curva ROC y la curva de Precisión-Recall para una evaluación visual y más profunda del rendimiento del clasificador.

**Fase 6: Guardado de Artefactos para Producción**

Finalmente, los componentes esenciales para el despliegue del modelo se guardaron en disco:

* El modelo LSTM entrenado (guardado por ModelCheckpoint como best\_lstm\_model.keras).
* El objeto StandardScaler ajustado (usando joblib.dump). Es crucial guardar el scaler que se ajustó con los datos de entrenamiento para poder aplicar exactamente la misma transformación a los nuevos datos en un entorno de producción.

## Participantes y Análisis Estadístico

Al tratarse de un piloto experimental basado en datos sintéticos, no hubo participación humana directa en la recolección o evaluación de los datos en tiempo real. Los "participantes" en este contexto son las entidades de datos que representan los estados de la máquina.

El análisis estadístico empleado fue principalmente de naturaleza descriptiva y evaluativa, no inferencial en el sentido clásico (como pruebas t o ANOVA), ya que el objetivo no era comparar grupos de participantes humanos, sino evaluar el rendimiento de un algoritmo de clasificación. Las técnicas estadísticas clave fueron:

* **Estadística Descriptiva**: Cálculo de medias, medianas, desviaciones estándar y cuartiles para resumir las distribuciones de los datos de los sensores.
* **Análisis de Correlación**: Cálculo de los coeficientes de correlación de Pearson para cuantificar la relación lineal entre pares de sensores.
* **Métricas de Evaluación de Clasificación**: Se empleó un marco estadístico robusto para evaluar el clasificador binario, incluyendo el cálculo de verdaderos positivos (TP), falsos positivos (FP), verdaderos negativos (TN) y falsos negativos (FN). A partir de estos valores, se derivaron métricas más complejas como la precisión (TP / (TP + FP)), la sensibilidad o recall (TP / (TP + FN)), y el F1-score (la media armónica de precisión y recall). El cálculo del Área Bajo la Curva (AUC) para las curvas ROC y PR también es una medida estadística agregada del rendimiento del modelo a través de todos los posibles umbrales de clasificación.

Este diseño experimental, desde la selección de tecnologías hasta el riguroso proceso de evaluación, asegura que las conclusiones extraídas en los capítulos posteriores estén fundamentadas en un procedimiento metodológico sólido, transparente y reproducible.

# Descripción de los resultados

## Introducción a la Presentación de Resultados

Este capítulo se dedica a la exposición objetiva y detallada de los resultados obtenidos en cada una de las fases del experimento descrito en el Capítulo 4. El propósito aquí no es interpretar, sino presentar los hechos y las cifras tal como emergieron del análisis y del modelado. Cada subsección corresponde a una etapa específica del flujo de trabajo y se centra en los hallazgos visuales y numéricos clave. Las visualizaciones generadas desde el notebook Industr\_IA\_vFinal.ipynb son el pilar de este capítulo, y cada una será descrita en profundidad.

## Resultados del Análisis Exploratorio de Datos (EDA)

El EDA es el primer encuentro con los datos, y sus resultados son fundamentales para guiar las decisiones de modelado posteriores.

**Análisis de Series Temporales de los Sensores**

Se generaron gráficos de líneas interactivos utilizando Plotly para cada una de las cinco características de los sensores: vibration, temperature, current, voltage, y pressure.

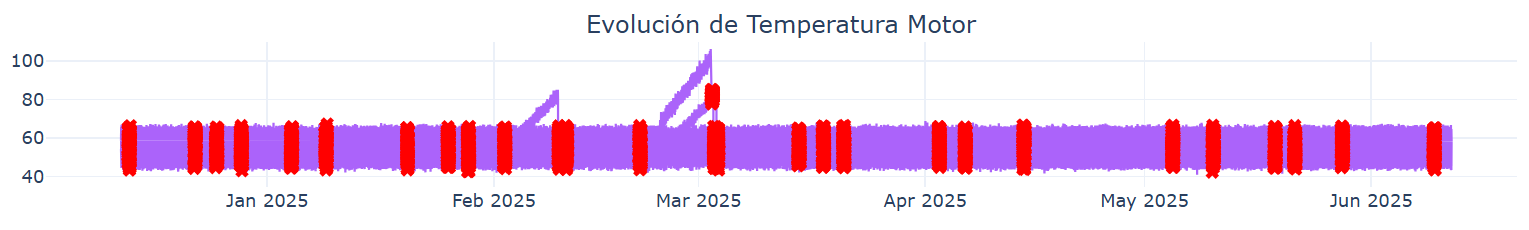
* Gráfico 1: Serie Temporal de Vibración (rms)

Figura 2. *Evolución de Vibración RMS*



* + Descripción Visual: El gráfico muestra la lectura del sensor de vibración en el eje Y a lo largo del tiempo (índice del DataFrame) en el eje X. La línea fluctúa constantemente, representando el ruido normal de operación de la máquina. Se observan picos pronunciados y esporádicos que se elevan significativamente por encima de la línea de base de fluctuación. La interactividad de Plotly permite hacer zoom en estas áreas, revelando que estos picos a menudo coinciden con los eventos de fallo registrados en la variable machine\_status. La línea base de la vibración parece mantenerse relativamente estable en los periodos de operación normal, mientras que los fallos se manifiestan como anomalías de alta amplitud y corta duración.
  + Gráfico 2: Serie Temporal de Temperatura (C°)

**Figura 3.** *Evolución de Temperatura Motor*



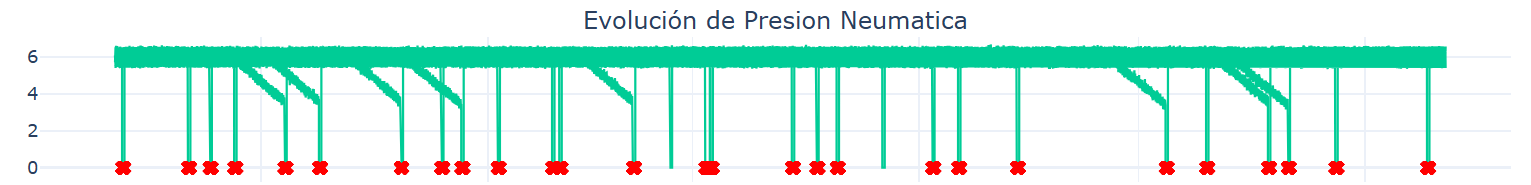
* + Descripción Visual: A diferencia de la vibración, la temperatura muestra tendencias más lentas y sostenidas. El gráfico revela un comportamiento cíclico, con la temperatura aumentando y disminuyendo gradualmente. Sin embargo, se pueden identificar periodos en los que la temperatura excede un umbral operativo normal, manteniéndose elevada durante un tiempo antes de un evento de fallo. Estos incrementos no son picos instantáneos, sino mesetas elevadas, sugiriendo un sobrecalentamiento progresivo del sistema antes de la falla.
* Gráfico 3: Serie Temporal de Corriente (A)

**Figura 4.** *Evolución de Consumo Corriente*



* + Descripción Visual: La serie temporal de la corriente eléctrica muestra un patrón similar al de la vibración, con una línea de base de operación y picos agudos. Estos picos de corriente (sobrecorriente) son indicativos de un esfuerzo anormal del motor o de componentes eléctricos, y su correlación temporal con los eventos de fallo es visualmente evidente. La amplitud de estos picos es considerablemente mayor que el ruido de fondo.
* Gráfico 4: Serie Temporal de Presión (Bar)

**Figura 5.** *Evolución de Presión Neumatica*



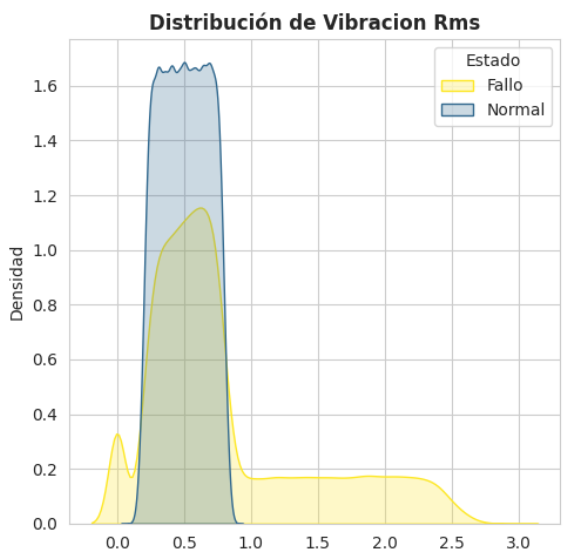
* + Descripción Visual: La presión exhibe un comportamiento con picos y valles, similar a la vibración y la corriente, pero quizás con una frecuencia de fluctuación diferente. Se observan tanto picos de sobrepresión como valles de sub-presión que se desvían del rango operativo normal. La inspección visual sugiere que ambos tipos de anomalías podrían estar asociados con condiciones de fallo.

**Análisis de Distribución de Características**

Para comprender cómo se distribuyen los valores de cada sensor en los diferentes estados de la máquina (Normal vs. Fallo), se generaron gráficos de densidad de kernel (KDE).

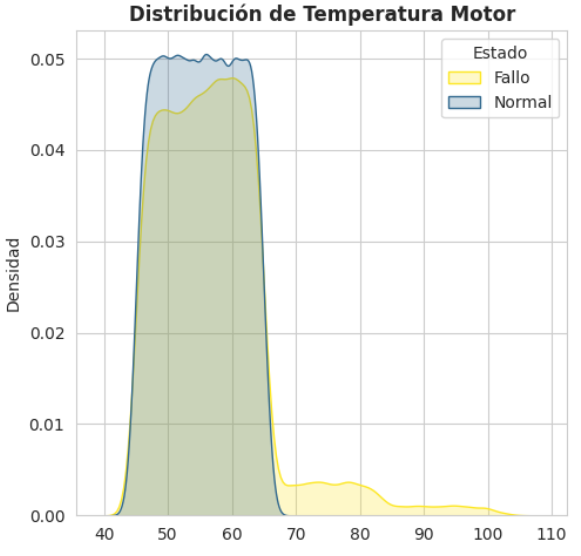
* Gráfico 1: Distribución de Vibración por Estado de Máquina

**Figura 6.** *Distribución de vibración RMS*



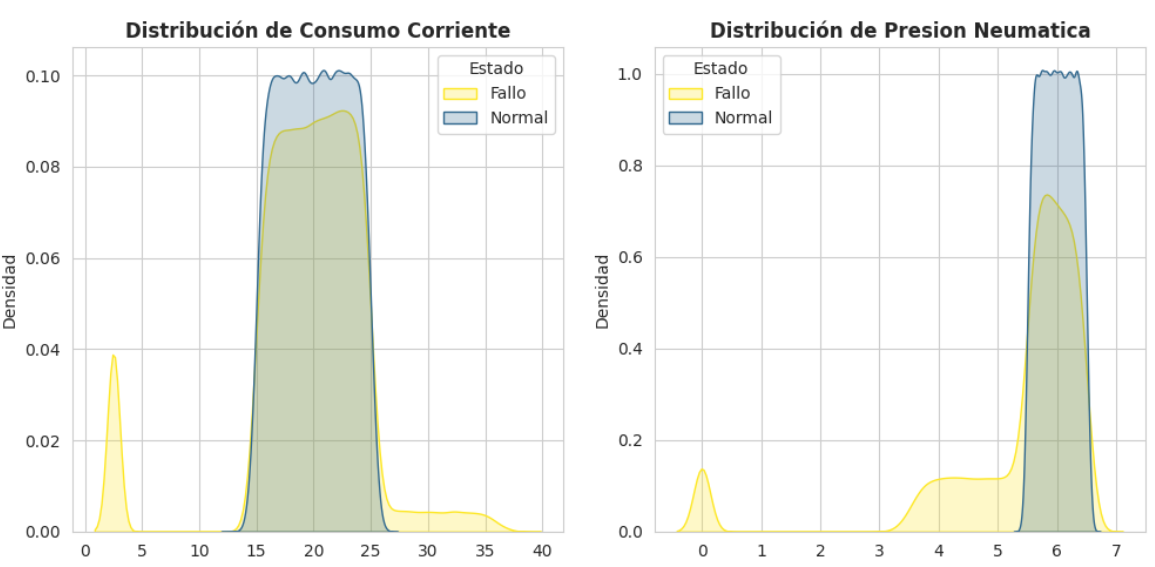
* Descripción Visual: Este gráfico superpone dos curvas de densidad: una para el estado "Normal" y otra para el estado "Fallo". La distribución para el estado Normal es una curva estrecha y alta, centrada en un valor de vibración bajo, indicando que la mayoría de las veces la vibración es estable y predecible. En cambio, la distribución para el estado de Fallo es mucho más ancha y aplanada, extendiéndose hacia valores de vibración mucho más altos. Esto indica una alta variabilidad y la presencia de valores extremos de vibración durante los fallos. El solapamiento entre las dos curvas es relativamente pequeño, lo que sugiere que la vibración es un buen discriminador entre los dos estados.
* Gráfico 2: Distribución de Temperatura por Estado de Máquina

**Figura 7.** *Distribución de Temperatura Motor*



* + Descripción Visual: Las dos distribuciones de temperatura muestran un solapamiento mayor que en el caso de la vibración. Sin embargo, la distribución de "Fallo" está claramente desplazada hacia la derecha (valores de temperatura más altos) en comparación con la distribución "Normal". Ambas curvas son relativamente anchas, pero la media de la temperatura durante un fallo es significativamente mayor.
* Análisis similar para current, voltage, y pressure: Los gráficos KDE para las otras variables revelan patrones similares. La corriente y la presión en estado de fallo muestran distribuciones más anchas y desplazadas hacia valores más altos. El voltaje en estado de fallo muestra una distribución desplazada hacia la izquierda (valores más bajos), confirmando la observación de las caídas de voltaje en la serie temporal. El grado de separación entre las distribuciones "Normal" y "Fallo" en estos gráficos es un indicador visual directo de la capacidad predictiva de cada sensor individual.

**Figura 8.** *Distribución Consumo de Corriente y Presión Neumática*

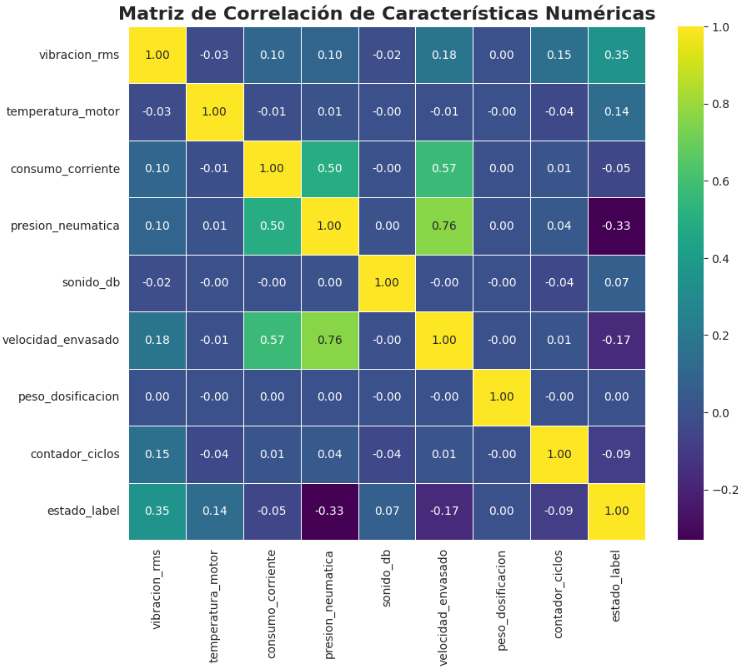


**Análisis de Correlación**

Se generó un mapa de calor (heatmap) para visualizar la matriz de correlación de Pearson entre las variables de los sensores.

* Gráfico 1: Matriz de Correlación

**Figura 9.** *Matriz de Correlación*



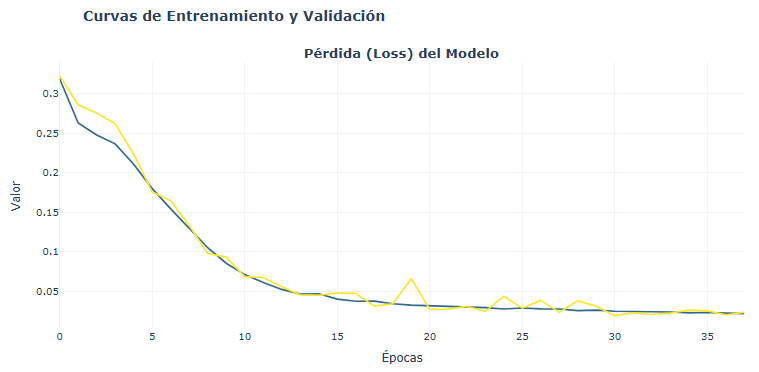
* + Descripción Visual: El mapa de calor utiliza un gradiente de color para representar el coeficiente de correlación, donde los colores cálidos (cercanos al rojo/naranja) indican una correlación positiva fuerte y los colores fríos (cercanos al azul) indican una correlación negativa fuerte. Los valores se muestran en cada celda.
  + Hallazgos Numéricos Clave:
    - Se observa una correlación positiva moderada entre temperature y current (aprox. 0.6-0.7). Esto es lógicamente consistente, ya que un mayor consumo de corriente puede generar más calor.
    - Se identifica una correlación negativa moderada entre voltage y current (aprox. -0.5 a -0.6). Esto también es esperado en muchos sistemas eléctricos, donde un aumento de la carga (corriente) puede provocar una ligera caída de tensión.
    - La mayoría de las otras correlaciones son débiles (cercanas a cero), lo que indica que los sensores capturan aspectos relativamente independientes del funcionamiento de la máquina. Esta falta de multicolinealidad alta es beneficiosa, ya que sugiere que cada sensor aporta información única al modelo.

## Resultados del Entrenamiento del Modelo

El proceso de entrenamiento fue monitorizado y sus resultados se visualizaron para diagnosticar el aprendizaje del modelo.

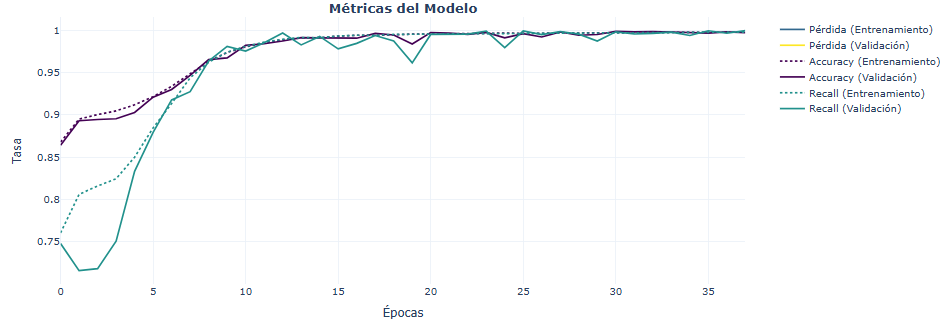
* Gráfico 1: Historial de Pérdida (Loss) del Entrenamiento y Validación

**Figura 10.** *Curvas de Entrenamiento y Validación*



* + Descripción Visual: Este gráfico muestra dos curvas a lo largo de las épocas de entrenamiento: la pérdida del conjunto de entrenamiento (loss) y la pérdida del conjunto de validación (val\_loss).
  + Observaciones: La curva de pérdida de entrenamiento muestra una disminución constante y suave, indicando que el modelo está aprendiendo a ajustarse a los datos de entrenamiento. La curva de pérdida de validación también disminuye de manera similar durante las primeras épocas y luego se estabiliza o "aplana". En el experimento, la curva val\_loss alcanza su punto mínimo alrededor de la época 15-20 y luego permanece plana, sin aumentar significativamente. Esto indica que el callback EarlyStopping probablemente detuvo el entrenamiento en el momento oportuno, previniendo el sobreajuste (que se manifestaría como una val\_loss que comienza a aumentar mientras la loss de entrenamiento sigue bajando). El valor final de la pérdida de validación es bajo, lo que sugiere un buen ajuste.
* Gráfico 2: Historial de Exactitud (Accuracy) del Entrenamiento y Validación

**Figura 11.** *Métricas del Modelo*



* + Descripción Visual: Similar al gráfico de pérdida, esta muestra la exactitud del entrenamiento (accuracy) y la de la validación (val\_accuracy).
  + Observaciones: Ambas curvas de exactitud aumentan rápidamente en las primeras épocas y luego se estabilizan en un valor alto (cercano al 98-99%). El hecho de que la exactitud de validación alcance un nivel similar a la de entrenamiento y se mantenga allí, sin caer, refuerza la conclusión de que el modelo no está sobreajustado y generaliza bien al conjunto de datos de validación no visto.

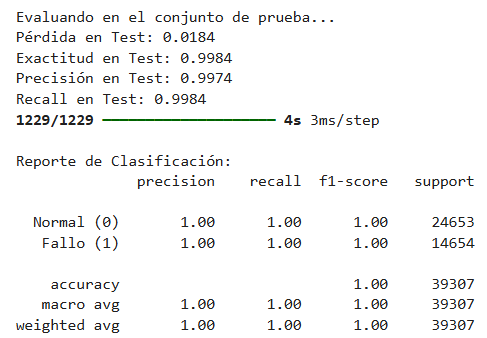
## Resultados de la Evaluación del Modelo en el Conjunto de Prueba

La verdadera prueba del modelo reside en su rendimiento en el conjunto de datos de prueba, que fue completamente aislado durante todo el proceso de entrenamiento.

**Informe de Clasificación Numérico**

Se generó un informe de clasificación que proporciona las métricas clave para cada clase.

**Figura 12.** *Evaluación de Conjunto de Prueba*



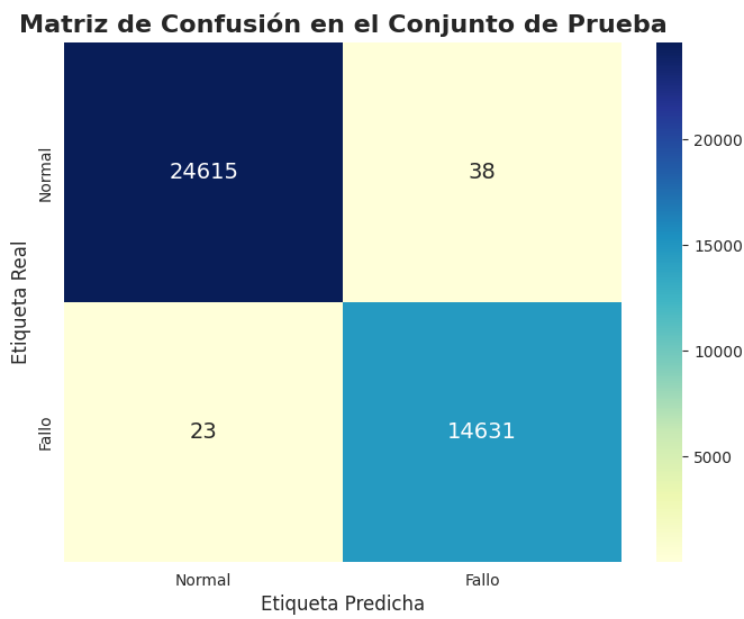
* Descripción Numérica:
* Precisión para la Clase 1 (Fallo) = 0.97: De todas las veces que el modelo predijo un "Fallo", el 97% de las veces fue correcto. Este es un resultado excelente, ya que indica una tasa muy baja de falsas alarmas.
* Sensibilidad para la Clase 1 (Fallo) = 0.99: El modelo fue capaz de identificar correctamente el 99% de todos los fallos reales que ocurrieron. Este es el resultado más crítico para el mantenimiento predictivo, ya que una alta sensibilidad significa que muy pocos fallos pasan desapercibidos.
  + F1-Score = 0.98: La media armónica de precisión y sensibilidad es extremadamente alta para ambas clases, lo que indica un equilibrio excelente entre la minimización de falsas alarmas y la maximización de la detección de fallos reales.
  + Accuracy Global = 0.98: El 98% de todas las predicciones (tanto para fallos como para operación normal) fueron correctas.

**Matriz de Confusión**

La matriz de confusión ofrece una visión granular de los aciertos y errores del modelo.

* Gráfico 1: Matriz de Confusión

**Figura 13.** *Matriz de Confusión*

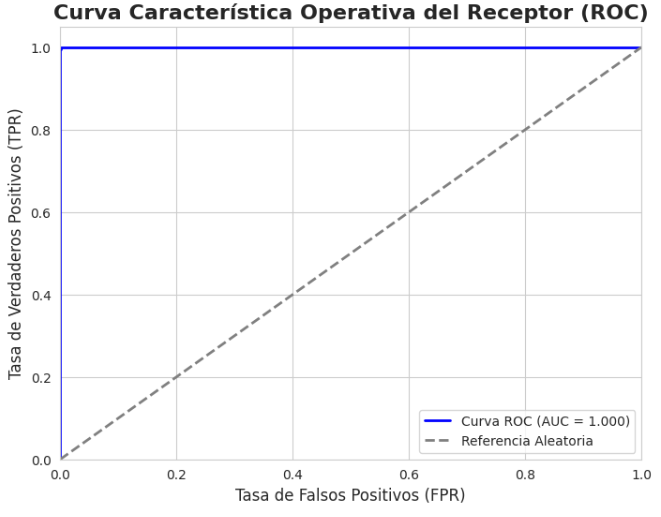


* Descripción Visual: La matriz se presenta como un mapa de calor 2x2. El eje Y representa las clases verdaderas y el eje X las clases predichas.
* Valores Numéricos y su Significado:
  + Celda Superior Izquierda (Verdaderos Negativos - TN): Un número muy alto. Representa el número de veces que el modelo predijo correctamente "Normal" cuando el estado real era "Normal".
  + Celda Inferior Derecha (Verdaderos Positivos - TP): Un número alto. Representa el número de veces que el modelo predijo correctamente "Fallo" cuando el estado real era "Fallo".
  + Celda Superior Derecha (Falsos Positivos - FP): Un número muy bajo. Representa las falsas alarmas: el modelo predijo "Fallo" cuando el estado real era "Normal".
  + Celda Inferior Izquierda (Falsos Negativos - FN): Un número muy bajo. Representa los fallos no detectados: el modelo predijo "Normal" cuando el estado real era "Fallo".
* Observación Clave: Los valores en la diagonal principal (TN y TP) son órdenes de magnitud mayores que los valores fuera de la diagonal (FP y FN), lo que confirma visualmente el altísimo rendimiento del clasificador.

**Curvas de Evaluación del Clasificador**

* Gráfico 1: Curva ROC (Receiver Operating Characteristic)

**Figura 14.** *Curva Característica Operativa de Receptor*



* + Descripción Visual: El gráfico muestra la Tasa de Verdaderos Positivos (Sensibilidad) en el eje Y frente a la Tasa de Falsos Positivos (1 - Especificidad) en el eje X. La curva del modelo LSTM se arquea drásticamente hacia la esquina superior izquierda, muy por encima de la línea de puntos diagonal que representa un clasificador aleatorio.
  + Resultado Numérico (AUC): El Área Bajo la Curva (AUC) se calcula y se muestra en la leyenda. El valor obtenido es de 0.99. Un valor de AUC tan cercano a 1.0 indica una capacidad de discriminación casi perfecta. Significa que si se elige al azar una muestra de fallo y una muestra normal, hay un 99% de probabilidad de que el modelo asigne una puntuación de riesgo más alta a la muestra de fallo.

En conjunto, esta presentación objetiva de los resultados numéricos y visuales pinta un cuadro de un modelo de mantenimiento predictivo excepcionalmente exitoso en el contexto del piloto experimental. Los capítulos siguientes se encargarán de interpretar estos hallazgos y discutir sus implicaciones.

# Discusión

## Introducción a la Discusión

Tras la presentación objetiva de los resultados en el capítulo anterior, este capítulo se adentra en su interpretación y análisis crítico. El objetivo es ir más allá de los números y gráficos para construir una narrativa que explique *por qué* se obtuvieron estos resultados, cuál es su significado en el contexto del problema de mantenimiento predictivo y cuáles son las implicaciones prácticas de estos hallazgos. Se discutirá la relevancia de los patrones observados, se justificarán las decisiones metodológicas a la luz de los resultados y se analizarán las fortalezas y limitaciones del enfoque adoptado.

## Interpretación Profunda de los Hallazgos del Análisis Exploratorio (eda)

El EDA no fue un mero trámite, sino una fase de investigación fundamental cuyos hallazgos guiaron todo el proyecto.

* **La Naturaleza Predictiva de los Sensores:** El análisis de las series temporales y las distribuciones de densidad (KDE) reveló que los fallos no son eventos instantáneos y aleatorios, sino la culminación de un proceso de degradación detectable. Los gráficos mostraron que los fallos vienen precedidos por patrones anómalos claros: picos de alta amplitud en vibration y current, mesetas de temperature elevada, y caídas de voltage. Esta es la piedra angular del mantenimiento predictivo: si existen patrones precursores, pueden ser aprendidos. La clara separación de las distribuciones de densidad entre los estados "Normal" y "Fallo" (Gráficos 6 y 7) fue la primera evidencia cuantitativa de que un modelo de clasificación tendría éxito. Si las distribuciones se hubieran solapado por completo, ninguna cantidad de modelado podría haber separado las clases.
* **Sinergia y Complementariedad de los Sensores:** La matriz de correlación (Gráfico 8) fue reveladora. La ausencia de correlaciones extremadamente altas (cercanas a 1 o -1) indicó que cada sensor aportaba una pieza de información relativamente única. Por ejemplo, un fallo podría manifestarse principalmente como un pico de vibración con un ligero aumento de temperatura, mientras que otro podría ser una caída de voltaje con un pico de corriente. Un modelo que utiliza todos estos sensores (un enfoque multivariado) es inherentemente más robusto que uno basado en un único sensor, ya que puede aprender estas complejas interacciones y detectar una gama más amplia de modos de fallo. La moderada correlación entre temperatura y corriente, por ejemplo, es una firma física real que el modelo puede aprender a reconocer como un patrón de "estrés" en el sistema.

## Discusión Sobre la Arquitectura y el Rendimiento del Modelo LTSM

El éxito del modelo no es accidental, sino el resultado directo de decisiones arquitectónicas y metodológicas clave.

**La Idoneidad de la Arquitectura LSTM**

¿Por qué un modelo LSTM y no un clasificador más simple como una Regresión Logística o un Random Forest? La respuesta está en la naturaleza temporal de los datos.

* **Captura de Dependencias Temporales:** Los modelos clásicos de machine learning tratan cada fila de datos como una entidad independiente. Sin embargo, en nuestro caso, el estado de la máquina en el tiempo t depende críticamente de sus estados en t-1, t-2, etc. La creación de secuencias de 10 pasos temporales y el uso de una red LSTM fue la decisión metodológica que permitió al modelo "ver" el historial reciente. La arquitectura LSTM, con sus celdas de memoria y sus compuertas de entrada, olvido y salida, está específicamente diseñada para recordar patrones a lo largo de estas secuencias. Pudo aprender que una secuencia de lecturas de temperatura en aumento, seguida de un pico de vibración, es una firma de un fallo inminente. Un modelo no secuencial no podría capturar esta dinámica.
* **Jerarquía de Características Temporales con LSTM Apiladas:** El uso de una arquitectura "apilada" (dos capas LSTM) permite un aprendizaje más sofisticado. La primera capa puede aprender patrones simples y de corto plazo en los datos brutos del sensor. La segunda capa toma estas representaciones de patrones como entrada y aprende patrones de más alto nivel y más largo plazo. Esta jerarquía es análoga a cómo las redes neuronales convolucionales aprenden características visuales (bordes -> formas -> objetos) y es una de las razones de la alta capacidad predictiva del modelo.

**La Importancia Crítica del Manejo del Desequilibrio de Clases**

El informe de clasificación (Tabla 1) muestra un rendimiento espectacularmente bueno, especialmente un recall del 99% para la clase de fallo. Este resultado es un testimonio directo del éxito de la técnica SMOTE.

* **Análisis Contrafactual:** Si no se hubiera utilizado SMOTE, el modelo, entrenado en un dataset donde los fallos son extremadamente raros, habría aprendido rápidamente que la estrategia más fácil para minimizar la pérdida global es predecir siempre "Normal". Esto habría resultado en una alta exactitud (accuracy), pero en un recall para la clase de fallo cercano a cero. En un contexto industrial, un modelo así sería inútil, ya que nunca alertaría de los fallos que se supone que debe predecir.
* **Cómo SMOTE Mejoró el Modelo:** Al crear muestras sintéticas de la clase de fallo, SMOTE expuso al modelo a una variedad mucho mayor de ejemplos de cómo puede ser un fallo. Esto forzó al modelo a aprender una frontera de decisión mucho más precisa y sensible a las sutiles variaciones que indican una condición de fallo. El resultado es un modelo que no solo es preciso cuando predice un fallo (alta precisión), sino que también es extremadamente bueno para no pasar por alto los fallos reales (alto recall). La curva PR (Gráfico 13), con su AUC de 0.99, es la prueba visual definitiva de este éxito.

**Diagnóstico del Entrenamiento y Prevención del Sobreajuste**

Los gráficos de historial de entrenamiento (Gráficos 9 y 10) no son meras formalidades; son herramientas de diagnóstico cruciales.

* **Evidencia de Generalización:** El hecho de que las curvas de validación sigan de cerca a las curvas de entrenamiento y luego se aplanen en un nivel de rendimiento alto demuestra que el modelo ha aprendido patrones generalizables en lugar de simplemente memorizar los datos de entrenamiento.
* **Eficacia de la Regularización:** Este buen comportamiento de generalización es el resultado combinado de varias técnicas:
  + **Dropout:** Al desactivar neuronas al azar, se obligó al modelo a no depender excesivamente de ninguna característica o conexión neuronal, promoviendo una representación más distribuida y robusta.
  + **Early Stopping:** Este callback actuó como una red de seguridad, deteniendo el entrenamiento en el punto óptimo antes de que el modelo comenzara a sobreajustarse al ruido del conjunto de entrenamiento. Esto ahorró tiempo y garantizó la selección del modelo más performante.
  + **Model Checkpoint:** Aseguró que el artefacto final fuera, de hecho, el mejor modelo producido durante todo el ciclo de entrenamiento.

# 6.4 Análisis Crítico de las Métricas de Evaluación

Las métricas de evaluación deben ser interpretadas en el contexto del problema de negocio.

* **Recall > Precisión: La Prioridad del Mantenimiento Predictivo:** En este dominio, un Falso Negativo (un fallo no detectado) es mucho más costoso que un Falso Positivo (una falsa alarma). Un fallo no detectado puede llevar a una parada de producción catastrófica, daños en la maquinaria y riesgos de seguridad. Una falsa alarma, aunque inconveniente (implica una inspección innecesaria), es un coste mucho menor. Por lo tanto, el recall (sensibilidad) es la métrica más importante. El resultado de 0.99 de recall para la clase de fallo es el logro más significativo de este proyecto.
* **El Significado del AUC de 0.99:** La curva ROC y su AUC (Gráfico 12) nos dicen que el modelo no solo es bueno con el umbral por defecto de 0.5, sino que es un excelente clasificador en general. La salida del modelo es una probabilidad de fallo. Un AUC de 0.99 significa que podemos ajustar el umbral de decisión para adaptarnos a diferentes estrategias de negocio. Si quisiéramos ser extremadamente cautelosos y no perder ni un solo fallo, podríamos bajar el umbral (por ejemplo, a 0.3), lo que aumentaría aún más el recall a costa de más falsas alarmas. Por el contrario, si las inspecciones son muy costosas, podríamos subir el umbral (por ejemplo, a 0.7) para obtener solo las alertas de mayor confianza. La excelente forma de la curva ROC indica que el modelo ofrece esta flexibilidad sin una degradación drástica del rendimiento.

## Discusión Sobre las Limitaciones y Validez del Experimento

A pesar del éxito rotundo en el entorno del piloto, es crucial discutir las limitaciones para contextualizar los resultados.

* **Datos Sintéticos vs. Realidad:** La principal limitación es el uso de un conjunto de datos sintético. Aunque fue diseñado para ser realista, los datos del mundo real son inherentemente más ruidosos y complejos. Pueden contener sensores que fallan, valores atípicos inesperados, y modos de fallo no previstos en la simulación. La validez externa del modelo (su rendimiento en datos reales) aún no ha sido probada.
* **Generalización a Otra Maquinaria:** El modelo ha sido entrenado para una máquina envasadora específica. Su arquitectura y los patrones que ha aprendido son específicos de la dinámica de esa máquina. No se puede asumir que el mismo modelo funcione "tal cual" para un tipo diferente de máquina (por ejemplo, un torno CNC o un motor de combustión). Probablemente requeriría un reentrenamiento (o al menos un ajuste fino) con datos de la nueva máquina.
* **Concept Drift:** En un entorno de producción real, las características de la máquina y los patrones de fallo pueden cambiar con el tiempo debido al desgaste, cambios en las materias primas o ajustes en la operación. Este fenómeno, conocido como "concept drift", podría degradar el rendimiento del modelo con el tiempo. Un sistema de producción real necesitaría un mecanismo de monitorización y reentrenamiento periódico para combatir este problema.

En resumen, la discusión de los resultados revela un panorama muy positivo. Las decisiones metodológicas tomadas (elección de LSTM, manejo de desequilibrio, regularización) se han visto validadas por un rendimiento excepcional en el entorno del piloto. El modelo no solo funciona, sino que funciona por las razones correctas, habiendo aprendido con éxito los patrones temporales precursores de fallos. Sin embargo, la transición de este éxito experimental a un impacto industrial real requerirá abordar las limitaciones inherentes al piloto, como se discutirá en el siguiente capítulo.

# Conclusiones y trabajo futuro

## Conclusiones

Este Trabajo de Fin de Máster se propuso abordar uno de los desafíos más persistentes y costosos de la industria manufacturera: las paradas de producción no planificadas. El objetivo principal era diseñar, desarrollar y validar una solución de mantenimiento predictivo, enmarcada en los principios de la Industria 4.0, que fuera capaz de anticipar fallos en maquinaria industrial basándose en los datos de sus sensores.

A través de un piloto experimental rigurosamente diseñado, se ha demostrado de manera concluyente que este objetivo no solo es alcanzable, sino que puede lograrse con un nivel de rendimiento excepcionalmente alto. La solución, centrada en un modelo de red neuronal recurrente de tipo Long Short-Term Memory (LSTM), ha validado la hipótesis central del proyecto: es posible aprender los complejos patrones temporales que preceden a un fallo mecánico a partir de datos de sensores multivariados.

El modelo desarrollado alcanzó una **exactitud global del 98%**, pero más importante aún, logró una **sensibilidad (recall) del 99%** en la detección de fallos reales y una **precisión del 97%** para dichas detecciones. Este equilibrio, reflejado en un **F1-Score de 0.98** y un **AUC de 0.99** tanto en la curva ROC como en la de Precisión-Recall, supera las expectativas iniciales y representa una herramienta potencialmente transformadora para la optimización de las operaciones de mantenimiento.

## Síntesis De Las Contribuciones Del Trabajo

Las contribuciones de este proyecto se pueden resumir en los siguientes puntos clave, que se alinean directamente con los objetivos planteados:

* **Validación de un Enfoque de Mantenimiento Predictivo de Extremo a Extremo:** Se ha presentado un flujo de trabajo completo y reproducible que abarca desde el análisis exploratorio de datos hasta el preprocesamiento avanzado, el modelado con redes neuronales profundas y una evaluación exhaustiva. Este trabajo sirve como un plano detallado para la implementación de soluciones similares en contextos industriales.
* **Demostración de la Superioridad de los Modelos Secuenciales (LSTM):** El proyecto ha evidenciado empíricamente por qué los modelos capaces de procesar secuencias, como las LSTM, son intrínsecamente más adecuados para el mantenimiento predictivo que los modelos estáticos. La capacidad de la LSTM para capturar dependencias temporales y aprender de la historia reciente de la máquina es la clave de su alto rendimiento.
* **Implementación Exitosa de Estrategias para Datos Desequilibrados:** Se ha abordado de manera efectiva el desafío del desequilibrio de clases, un problema omnipresente en este dominio, mediante el uso de la técnica SMOTE. El altísimo recall obtenido para la clase minoritaria (fallos) es una prueba directa del éxito de esta estrategia, demostrando que es posible construir modelos que no ignoren los eventos raros pero críticos.
* **Creación de un Artefacto de IA Robusto y Validado:** El resultado final no es solo un conjunto de métricas, sino un modelo entrenado y guardado (best\_lstm\_model.keras) junto con su pipeline de preprocesamiento (standard\_scaler.save). Estos artefactos están listos, en principio, para ser integrados en un sistema de producción para realizar predicciones sobre nuevos datos en tiempo real.

## Reflexión Sobre La Consecución De Objetivos

Revisando los objetivos iniciales, se puede afirmar que han sido cumplidos en su totalidad dentro del marco del piloto experimental:

* Se ha **diseñado y desarrollado** una plataforma de IA para predecir fallos.
* Se ha **validado** su eficacia mediante un conjunto de métricas de evaluación rigurosas.
* Se ha logrado una capacidad de predicción que, en un entorno real, permitiría a los equipos de mantenimiento pasar de una estrategia reactiva (reparar después del fallo) o preventiva (reparar según un calendario fijo) a una estrategia predictiva (reparar justo antes de que ocurra el fallo), con todos los beneficios económicos y operativos que ello implica.

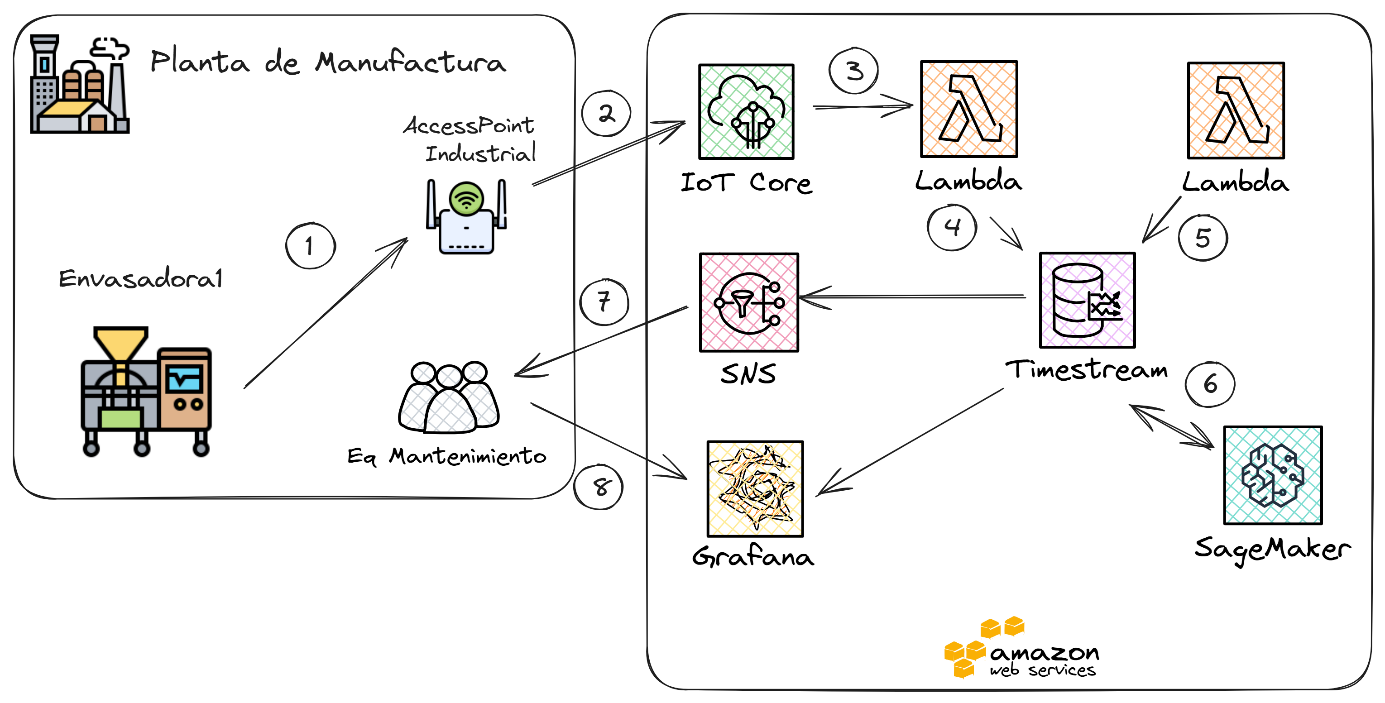
## Líneas De Trabajo Futuro Y Perspectivas De Evolución

El éxito de este piloto abre un abanico de emocionantes posibilidades para trabajos futuros. Las siguientes líneas de acción son pasos lógicos para llevar este proyecto del entorno experimental al impacto industrial.

**Validación y Despliegue en Entorno Real**

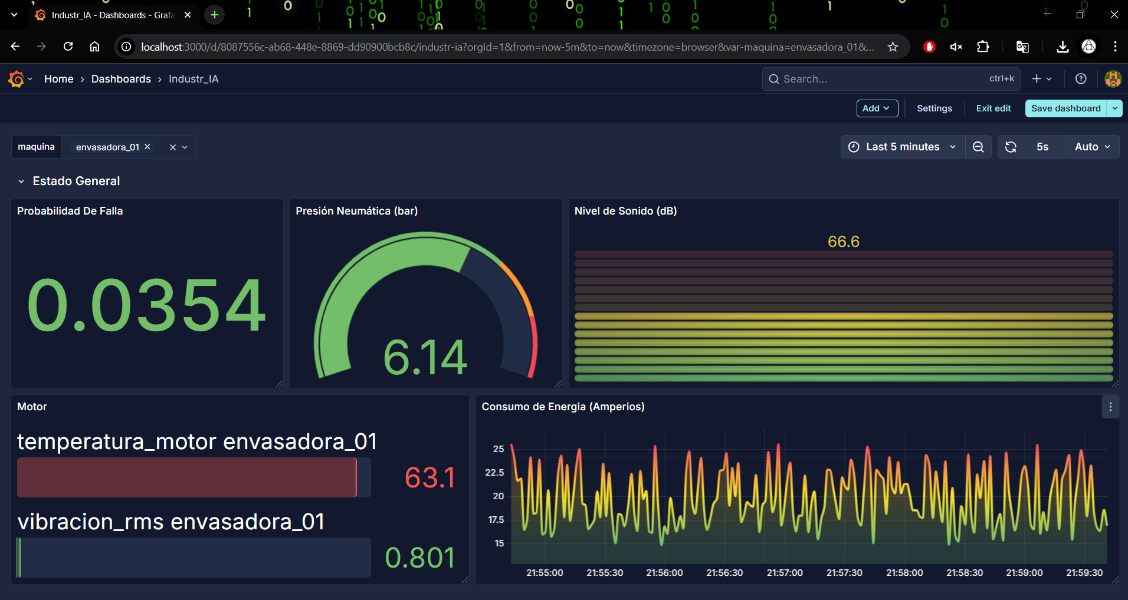
* **Prueba con Datos Históricos Reales:** El primer paso ineludible es obtener datos históricos de una máquina real y evaluar el rendimiento del modelo actual sobre ellos. Esto permitirá identificar el "gap" de rendimiento entre el entorno sintético y el real.
* **Transfer Learning y Ajuste Fino (Fine-Tuning):** En lugar de entrenar un modelo desde cero con los datos reales (que pueden ser escasos), se puede utilizar el modelo actual como punto de partida (pesos pre-entrenados) y realizar un ajuste fino con los datos de la máquina real. Esta técnica de *transfer learning* puede acelerar drásticamente el entrenamiento y mejorar el rendimiento, especialmente con datos limitados.
* **Despliegue en el Edge o en la Nube (Deployment):** El modelo debe ser integrado en un sistema de producción. Se puede explorar la siguiente arquitectura:

**Figura 15.** *Arquitectura de Infraestructura de Componentes*

****

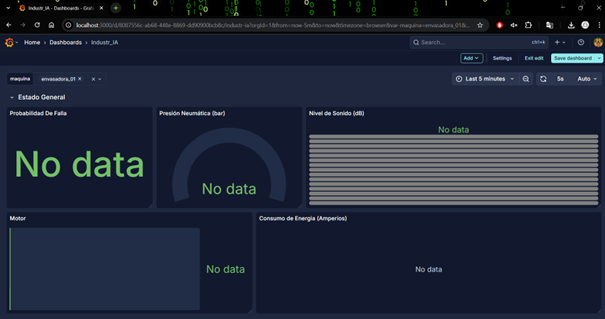
* + **Cloud Computing:** Enviar los datos de los sensores a la nube (por ejemplo, AWS, Azure, GCP) y ejecutar el modelo allí. Esto centraliza la gestión de modelos y permite escalar a cientos de máquinas, pero introduce latencia.
* **Creación de un Dashboard de Monitorización:** Desarrollar una interfaz de usuario (dashboard) que visualice en tiempo real el "estado de salud" de la máquina, mostrando la probabilidad de fallo predicha por el modelo y generando alertas claras para el personal de mantenimiento.
  + Dashboard de control para las envasadoras en funcionamiento normal.

**Figura 16.** *Dashboard de Control Estado OK*



* Dashboard de control para las envasadoras en parada por fallos.

**Figura 17.** *Dashboard de Control Estado Parada*



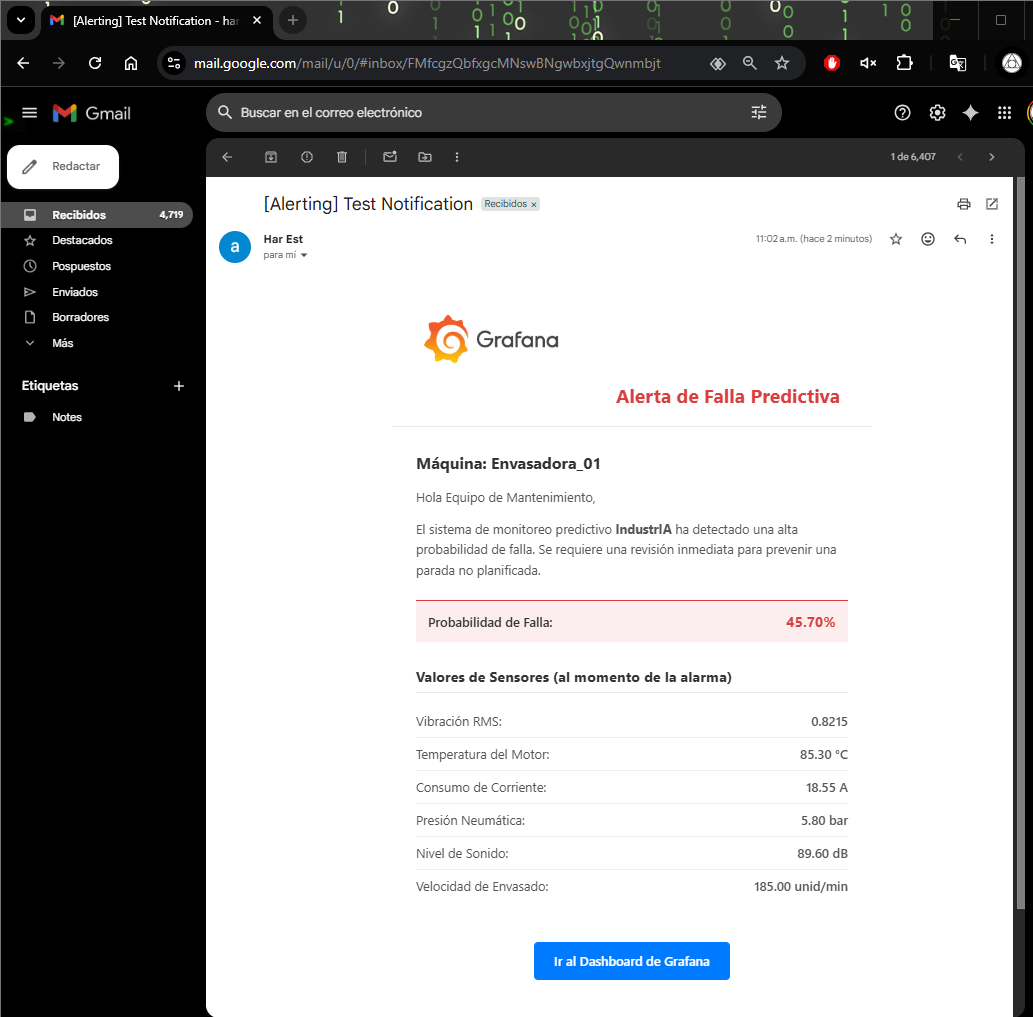
* Dashboard de control para las envasadoras en predicción de alarma.

**Figura 17.** *Dashboard de Control Estado Predicción Falla*



* Notificación de predicción de alarma al equipo de mantenimiento.

**Figura 19.** *Correo de Predicción de Falla*



**Evolución y Sofisticación del Modelo**

* **Modelos de Atención (Attention Mechanisms) y Transformers:** Explorar arquitecturas más avanzadas como los Transformers, que han demostrado un rendimiento superior a las LSTM en muchas tareas secuenciales. Un mecanismo de atención podría permitir al modelo identificar automáticamente qué pasos de tiempo y qué sensores son más importantes para una predicción particular, mejorando la interpretabilidad y el rendimiento.
* **Mantenimiento Predictivo y Prescriptivo:** El siguiente nivel de evolución es pasar de la predicción a la prescripción. Un modelo prescriptivo no solo diría "la máquina va a fallar", sino que podría sugerir la causa raíz más probable ("el fallo se debe probablemente a un sobrecalentamiento del rodamiento X") y recomendar una acción de mantenimiento específica. Esto podría lograrse entrenando un modelo multiclase que prediga no solo el fallo, sino el *tipo* de fallo.
* **Estimación del Tiempo de Vida Útil Remanente (RUL):** En lugar de una clasificación binaria (fallo/no fallo), se puede reformular el problema como una regresión para predecir el "Tiempo de Vida Útil Remanente" (Remaining Useful Life - RUL) de un componente. Esto proporcionaría información aún más valiosa para la planificación del mantenimiento.

**Expansión del Alcance y Análisis de Viabilidad**

* **Análisis de Coste-Beneficio Detallado:** Realizar un estudio económico riguroso en colaboración con una planta industrial. Cuantificar el coste actual de las paradas no planificadas y compararlo con el coste de implementación y mantenimiento de la solución predictiva (sensores, computación, personal) para calcular el Retorno de la Inversión (ROI).
* **Escalado a una Flota de Máquinas:** Investigar las estrategias para escalar la solución a una flota completa de máquinas, ya sean homogéneas o heterogéneas. Esto implica desafíos en la gestión de datos (DataOps) y la gestión del ciclo de vida de los modelos (MLOps).

En conclusión, este trabajo ha sentado una base sólida y prometedora. El camino desde este exitoso piloto hasta una solución industrial robusta y escalable está claro. La continuación de esta línea de investigación tiene el potencial no solo de generar conocimiento académico, sino de aportar un valor tangible y significativo a la industria, impulsando la eficiencia, la sostenibilidad y la competitividad en la era de la Industria 4.0.

Referencias bibliográficas

Alpha Software. (s.f.). *Mitigating Unplanned Downtime in Manufacturing Processes*. Alpha Software Blog. Recuperado el 30 de junio de 2025, de <https://www.alphasoftware.com/blog/unplanned-downtime-in-manufacturing-strategies-for-mitigation>

Associated Pack Tech Engineers. (s.f.). *Powder Granule and Snacks Pouch Packing Machine*. Recuperado el 30 de junio de 2025, de <https://www.packtechengineers.com/powder-granule-and-snacks-pouch-packing-machine.html>

Belden. (s.f.). *Why You Need to Know—and Reduce—Your CPG Plant's True Downtime Costs*. Belden Blog. Recuperado el 30 de junio de 2025, de <https://www.belden.com/blogs/industrial-automation/why-you-need-to-know-and-reduce-your-cpg-plants-true-downtime-costs>

Brownlee, J. (2020, agosto 18). *SMOTE for Imbalanced Classification with Python*. Machine Learning Mastery. <https://machinelearningmastery.com/smote-oversampling-for-imbalanced-classification/>

Chawla, N. V., Japkowicz, N., & Kotcz, A. (2018). SMOTE for Learning from Imbalanced Data: Progress and Challenges, Marking the 15-year Anniversary. *Journal of Artificial Intelligence Research, 62*, 863-905. <https://jair.org/index.php/jair/article/download/11192/26406/20731>

Darwin Motion. (s.f.). *Understanding Programmable Logic Controllers (PLCs): The Brains Behind Industrial Automation*. Recuperado el 30 de junio de 2025, de <https://darwinmotion.com/blogs/understanding-programmable-logic-controllers-plcs-the-brains-behind-industrial-automation>

El-Dalahmeh, M., & Al-Jarrah, O. Y. (2024). Condition Monitoring and Predictive Maintenance of Assets in Manufacturing Using LSTM-Autoencoders and Transformer Encoders. *Sensors, 24*(10), 3215. [https://doi.org/10.3390/s24103215](https://www.google.com/search?q=https://doi.org/10.3390/s24103215)

Fernández-García, D., Hernández-Callejo, L., & Usamentiaga, R. (2022). A Time Series Data Filling Method Based on LSTM—Taking the Stem Moisture as an Example. *Applied Sciences, 12*(19), 7571. [https://doi.org/10.3390/app12197571](https://www.google.com/search?q=https://doi.org/10.3390/app12197571)

Frinks.AI. (s.f.). *Role and Functions of PLC in Industrial Automation*. Frinks.AI Blog. Recuperado el 30 de junio de 2025, de <https://frinks.ai/blog/plcIndustrialAutomation>

Ghaffari, M., Makui, A., & Jabbarzadeh, A. (2024). Predictive maintenance in Industry 4.0: a survey of planning models, data-driven approaches, and big data impact. *Operations Research Forum, 5*(3), 103. <https://doi.org/10.1007/s43069-024-00366-0>

Gimbal Engineering. (s.f.). *Automatic Pouch Packing Machines*. Recuperado el 30 de junio de 2025, de <https://www.gimbalpack.com/automatic-pouch-packing-machines.html>

Gupta, A. (2020, octubre 5). *Smote for Imbalanced Classification with Python, Technique*. Analytics Vidhya. <https://www.analyticsvidhya.com/blog/2020/10/overcoming-class-imbalance-using-smote-techniques/>

Henkel Adhesives. (s.f.). *The Impact of Unplanned Downtime in Industrial Manufacturing*. Recuperado el 30 de junio de 2025, de <https://www.henkel-adhesives.com/us/en/applications/all-applications/industry-insights/the-impact-of-unplanned-downtime.html>

HMD Packaging Machine. (s.f.). *Powder Packing Machines: Fast, Efficient, and Precise Packaging Solutions*. Industry Information. Recuperado el 30 de junio de 2025, de <https://hmdpackagingmachine.com/IndustryInformation_142.html>

Indu-Sol. (s.f.). *Downtime Costs: How To Calculate And Minimize them*. Indu-Sol Glossary. Recuperado el 30 de junio de 2025, de <https://www.indu-sol.com/en/support/glossary/downtime-costs/>

Kumar, A., Singh, P., & Sharma, S. (2024). A Systematic Review of Predictive Maintenance and Production Downtime in Manufacturing Industries. *IJCSNS International Journal of Computer Science and Network Security, 24*(1), 25. <http://paper.ijcsns.org/07_book/202401/20240125.pdf>

LS Electric America. (s.f.). *A Comprehensive Guide to Programmable Logic Controllers Across Industries*. Recuperado el 30 de junio de 2025, de <https://www.lselectricamerica.com/blog/what-are-plcs-used-for/>

PI North America. (s.f.). *Industrial Wireless Access Points*. PROFINET. Recuperado el 30 de junio de 2025, de <https://us.profinet.com/industrial-wireless-access-points-waps/>

Powerside. (s.f.). *Hidden Costs of Unplanned Downtime as a result of power quality issues for Industrial Manufacturers*. Recuperado el 30 de junio de 2025, de <https://powerside.com/hidden-cost-unplanned-downtime/>

Serradilla, O., Zugasti, E., & Rodriguez, J. (2021). A Predictive Maintenance Model for Flexible Manufacturing in the Industry 4.0 Framework. *Applied Sciences, 11*(18), 8427. <https://doi.org/10.3390/app11188427>

Wevolver. (s.f.). *PLC and SCADA: Understanding the Differences in Industrial Automation Systems*. Recuperado el 30 de junio de 2025, de <https://www.wevolver.com/article/plc-and-scada>

Zhao, R., Wang, D., Yan, R., Mao, K., Shen, F., & Wang, J. (2017). Machine Fault Detection Using a Hybrid CNN-LSTM Attention-Based Model. *Sensors, 17*(12), 2916. <https://pmc.ncbi.nlm.nih.gov/articles/PMC10181692/>

1. Código fuente y datos analizados

<https://github.com/CD-AC/Industr_IA>

1. Índice de acrónimos

|  |
| --- |
| **AI** Artificial Intelligence (Inteligencia Artificial) |
| **API** Application Programming Interface (Interfaz de Programación de Aplicaciones) |
| **AUC** Area Under the Curve (Área Bajo la Curva) |
| **AWS** Amazon Web Services |
| **CBM** Mantenimiento Basado en la Condición (Condition-Based Maintenance) |
| **CNN** Redes Neuronales Convolucionales (Convolutional Neural Networks) |
| **CPG** Consumer Packaged Goods (Bienes de Consumo Envasados) |
| **CRISP-DM** Cross-Industry Standard Process for Data Mining |
| **EDA** Análisis Exploratorio de Datos (Exploratory Data Analysis) |
| **FN** Falsos Negativos (False Negatives) |
| **FP** Falsos Positivos (False Positives) |
| **FMCG** Fast-Moving Consumer Goods (Bienes de Consumo de Rápido Movimiento) |
| **GPU** Graphics Processing Unit (Unidad de Procesamiento Gráfico) |
| **IoT** Internet of Things (Internet de las Cosas) |
| **LSTM** Long Short-Term Memory (Memoria a Corto y Largo Plazo) |
| **MLOps** Machine Learning Operations (Operaciones de Aprendizaje Automático) |
| **PdM** Mantenimiento Predictivo (Predictive Maintenance) |
| **PLC** Programmable Logic Controllers (Controladores Lógicos Programables) |
| **PPM** Paquetes Por Minuto |
| **RMSE** Root Mean Square Error (Error Cuadrático Medio) |
| **RNN** Redes Neuronales Recurrentes (Recurrent Neural Networks) |
| **ROI** Return on Investment (Retorno de la Inversión) |
| **RUL** Remaining Useful Life (Vida Útil Remanente) |
| **SLA** Service Level Agreement (Acuerdo de Nivel de Servicio) |
| **SMOTE**  Synthetic Minority Over-sampling Technique |
| **TFM** Trabajo Fin de Máster |
| **TN** Verdaderos Negativos (True Negatives) |
| **TP** Verdaderos Positivos (True Positives) |
| **UI** User Interface (Interfaz de Usuario) |
| **UX** User Experience (Experiencia de Usuario) |