Cersofíos, Sofía

Morellato, Agostina

Modelo de Machine Learning para la predicción del riesgo de Cianobacterias en el Embalse San Roque para e Instituto Nacional del Agua - SCIRSA

Tesis para la obtención del título de grado de Ingenieras de Sistemas

Director: Porrini, Federico

Co-directora: Ruibal Conti, Ana Laura

UNIVERSIDAD CATÓLICA DE CÓRDOBA
JESUITAS

Informe de Trabajo Final

Modelo de Machine Learning para la predicción del riesgo de Cianobacterias en el Embalse San Roque para el Instituto Nacional del Agua - SCIRSA

Ingeniería de Sistemas

Cersofíos Sofía (2011293) - Morellato Agostina (2004827)

Director: Federico Porrini

Co-directora: Ana Laura Ruibal Conti

Facultad de Ingeniería - Córdoba, 2025

ÍNDICE

ABSTRACT

The San Roque reservoir, located in Córdoba, Argentina, is undergoing an advanced eutrophication process that has significantly deteriorated water quality. This situation threatens its recreational use, its role as a source of drinking water, and the biodiversity of the aquatic ecosystem.

The National Institute for Water (INA) through its regional center (SCIRSA) monitors the water quality of the reservoir through physical and biological data collection. However, data analysis remains a manual process, increasing researcher work, introducing errors, and delaying critical decision-making.

This study proposes developing a machine learning-based predictive model using INA-SCIRSA databases to detect patterns in cyanobacteria blooms and forecast water quality conditions. Algorithms such as multi-layer perceptron neural networks, Random Forest and Lineal Regression, will be evaluated to predict key variables, including cyanobacteria quantity, chlorophyll levels, and cyanobacteria dominance in the ecosystem.

Additionally, a graphical interface will be designed to visualize results intuitively, enhancing researchers' ability to interpret data efficiently. This system aims to optimize data analysis, improve event detection accuracy, and provide decision-support.

RESUMEN

El Embalse San Roque (ESR), en Córdoba, Argentina, enfrenta un avanzado proceso de eutrofización que ha deteriorado la calidad del agua, afectando su uso recreativo, su función como fuente de agua potable y la biodiversidad del ecosistema acuático.

El Instituto Nacional del Agua (INA), a través de su centro regional SCIRSA realiza un monitoreo continuo del embalse mediante la recolección de datos físicos, químicos y biológicos. Sin embargo, el análisis de esta información sigue siendo manual, lo que incrementa la carga de trabajo de los profesionales, introduce posibles errores y retrasa la toma de decisiones.

Este trabajo propone el desarrollo de un modelo predictivo basado en machine learning que utilice las bases de datos del INA-SCIRSA para identificar patrones en la proliferación de cianobacterias y predecir la calidad del agua. Se evaluarán algoritmos como Redes Neuronales Perceptrón Multicapa (MLP), Random Forest y Regresión logística, para estimar variables clave, como la cantidad de cianobacterias, los niveles de clorofila y la dominancia de cianobacterias en el ecosistema.

Además, se desarrollará una interfaz gráfica que facilite la visualización de los resultados, permitiendo a los profesionales acceder a la información de manera intuitiva. Este sistema busca optimizar el análisis de datos, mejorar la detección de eventos críticos y servir como una herramienta de apoyo en la toma de decisiones.

PRESENTACIÓN DEL TEMA

En las últimas décadas, la creciente preocupación por la calidad del agua en embalses y lagos ha impulsado el desarrollo de nuevas estrategias para su monitoreo y conservación.

En el caso del ESR, ubicado en la provincia de Córdoba, Argentina, el avance del proceso de eutrofización ha generado un deterioro progresivo en sus condiciones ambientales, afectando su uso recreativo, su función como fuente de agua potable y la biodiversidad del ecosistema acuático.

Desde hace más de 20 años, el Instituto Nacional del Agua (INA), a través del *Programa Permanente de Monitoreo del Embalse San Roque y su Cuenca*, ha recopilado datos físicos, químicos y biológicos para evaluar la calidad del agua y estudiar la evolución de este fenómeno. Sin embargo, el análisis y la gestión de esta información dependen de procesos manuales, que no solo aumenta la carga de trabajo de los profesionales, sino que aumenta el riesgo de introducir errores humanos y retrasos en la toma de decisiones.

Uno de los principales desafíos es la integración de datos provenientes de distintas fuentes. Actualmente, el monitoreo del embalse se basa en mediciones de calidad del agua, realizadas *in situ* y en análisis de laboratorio sobre las muestras tomadas en el mismo sitio de las mediciones, ambas se combinan con datos hidrometeorológicos obtenidos a través de sensores automáticos. La falta de un sistema automatizado de análisis y predicción, que analice en conjunto ambas bases de datos, dificulta la interpretación eficiente de esta información y limita la capacidad de respuesta ante episodios críticos, como la proliferación de cianobacterias potencialmente tóxicas.

Para abordar esta problemática, se propone el desarrollo de un modelo de predicción basado en machine learning, que permita analizar los datos históricos y en tiempo real, identificando patrones en la aparición de cianobacterias y así, según estos, poder predecir el estado futuro del ESR. Este sistema estará diseñado para:

- Integrar y procesar datos de calidad del agua e hidrometeorológicos de manera automatizada.
- Identificar tendencias y patrones que influyen en la proliferación de cianobacterias, entre las variables propuestas por el INA.
- Generar predicciones sobre las variables objetivo.

- Automatizar la detección de anomalías y eventos críticos mediante umbrales predefinidos.
- Visualizar los resultados a través de una interfaz intuitiva, accesible para los investigadores y gestores del INA.
- Resaltar predicciones críticas, según los umbrales, en la interfaz. Esto facilitará la interpretación de los resultados de la predicción y permitirá una respuesta más rápida ante situaciones de riesgo.

El uso de técnicas avanzadas de machine learning, como redes neuronales MLP, Random Forest y Regresión Lineal, junto con la información de las bases de datos relacionales y los procesos de ETL (Extracción, Transformación y Carga de datos) que transforman los datos que ingresaran, permitirá la construcción de un modelo robusto.

Esto, se espera, contribuirá a la prevención de crisis ambientales en el ESR, reemplazando procesos manuales por un modelo de predicción automático, mejorando la precisión en la detección de riesgos y facilitando la toma de decisiones basadas en datos confiables.

GLOSARIO

Hidrología y Calidad del Agua:

Embalse San Roque (ESR): Lago artificial ubicado en Córdoba, Argentina, construido en 1888. Su monitoreo ambiental se centra en el control de la calidad del agua debido a su avanzado estado de eutrofización.

Eutrofización: Proceso en el que un cuerpo de agua se enriquece excesivamente en nutrientes, como nitrógeno y fósforo, favoreciendo la proliferación de fitoplancton como las cianobacterias.

Cianobacterias: Microorganismos fotosintéticos que pueden formar floraciones masivas en ambientes acuáticos. Algunas especies producen toxinas peligrosas para la salud humana y la fauna acuática.

Floraciones algales: Crecimiento masivo y descontrolado de fitoplancton (como cianobacterias) en cuerpos de agua, impulsado por el exceso de nutrientes y condiciones ambientales favorables.

Fitoplancton: Microorganismos fotosintéticos suspendidos en el agua, fundamentales en la cadena trófica de los ecosistemas acuáticos.

Sitios de medición: Puntos estratégicos donde se realizan monitoreos en el Embalse San Roque.

Perfiles de agua: Niveles de profundidad en los que se toman muestras dentro del embalse.

Condición térmica: Estado de mezcla o estratificación del embalse en función de la temperatura en la columna de agua.

Cuenca hidrográfica: es un área geográfica donde toda el agua superficial (proveniente de lluvias, ríos, arroyos, etc.) drena hacia un mismo punto, generalmente un río principal, un lago o un embalse. En este caso representa en qué cuenca están ubicadas las estaciones de monitoreo, indicando el sistema fluvial en el que recoge datos

DIAGNÓSTICO

Actualmente, el análisis de la calidad del agua y la detección de floraciones algales en el Embalse San Roque son realizados manualmente por los profesionales del INA-SCIRSA. Sin embargo, la evaluación de esta información se realiza de manera manual, lo que implica varias dificultades:

- Carga de trabajo elevada: Los investigadores deben analizar grandes volúmenes de datos sin herramientas automatizadas.
- Mayor posibilidad de errores: La intervención manual en la interpretación de los datos puede generar imprecisiones.
- Retraso en la toma de decisiones: La detección de eventos críticos, como la proliferación de cianobacterias, puede no ser lo suficientemente rápida para prevenir impactos negativos.

La falta de un sistema automatizado de análisis y predicción reduce la eficiencia en la gestión del embalse y limita la capacidad de respuesta ante eventos críticos.

El INA-SCIRSA dispone de dos bases de datos que almacenan todas las variables necesarias para evaluar el estado actual del embalse y predecir su evolución. Estas bases contienen información clave que permite establecer patrones y tendencias en la calidad del agua, lo que abre la posibilidad de desarrollar un modelo de predicción que automatice el análisis y el procesamiento de estos datos. El modelo predictivo se nutrirá exclusivamente de los datos ya almacenados en las bases del INA-SCIRSA.

OBJETIVOS

GLOBAL

Contribuir a la protección de la salud pública y del ecosistema del Embalse San Roque mediante el desarrollo de un sistema automatizado de predicción de floraciones de cianobacterias, basado en técnicas de machine learning. Este modelo permitirá al Instituto Nacional del Agua (INA-SCIRSA) anticipar eventos críticos asociados a la eutrofización, optimizar la gestión de los recursos hídricos y facilitar decisiones estratégicas.

ESPECÍFICOS

- Integrar las bases de datos de calidad del agua e hidrometeorología y estructurarlas consumiendo solo los datos útiles para el modelo. Permitiendo un posterior análisis de la información disponible.
- Realizar un análisis exploratorio de datos (en inglés EDA) para evaluar la calidad de la información, detectando valores atípicos y datos faltantes.
- Desarrollar e implementar estrategias para el tratamiento de los datos en base al análisis exploratorio previo, como:
    - Imputación de valores nulos
    - Eliminación de valores o registros que ensucien el dataframe.
    - Transformación o generación de nuevas variables
- Seleccionar y testear algoritmos de machine learning adecuados para las estrategias de tratamientos de datos y para las predicciones de cianobacterias.
- Evaluar y comparar el desempeño de los modelos predictivos, utilizando métricas de error y validación cruzada, asegurando que el modelo final sea el más preciso posible.
- Automatizar la ejecución del modelo y su integración con las bases de datos, asegurando que las predicciones se actualicen en tiempo real con cada nuevo ingreso de datos, sin necesidad de intervención manual.

- Desarrollar una interfaz gráfica intuitiva que permita visualizar de manera clara y accesible los resultados de las predicciones, facilitando la interpretación de los datos a los investigadores y gestores del INA.
- Desarrollar documentación para el personal del INA-SCIRSA, sobre el modelo, para facilitar la comprensión sobre su uso.

MARCO TEÓRICO

A NIVEL DOMINIO

El Embalse San Roque y su Monitoreo Ambiental

El Embalse San Roque (ESR), ubicado en la provincia de Córdoba, es un lago artificial construido en 1888 con el propósito de abastecer de agua para consumo humano, control de crecidas, riego y generación de energía hidroeléctrica. En la actualidad, además de su función original, el embalse constituye uno de los principales centros turísticos del país, con un uso recreativo intensivo, especialmente en la localidad de Villa Carlos Paz.

Sin embargo, la calidad del agua del embalse se ha deteriorado progresivamente, presentando en la actualidad un estado avanzado de eutrofización *(Pussetto y col., 2018)*. Este fenómeno ha generado impactos ambientales significativos, como la mortandad de peces y la proliferación de malos olores, además de consecuencias económicas (mayores costos en infraestructura para la potabilización del agua, remediación del lago y reducción del turismo) y sociales (afectaciones a la salud pública y cierre de instituciones educativas).

Fig 1. Embalse San Roque

Eutrofización y Floraciones Algales

La eutrofización del ESR se ha intensificado en las últimas décadas debido al aumento en el aporte de nutrientes, principalmente fósforo y nitrógeno. Este proceso favorece la proliferación de algas, entre ellas las cianobacterias, generando floraciones algales visibles, especialmente en períodos de altas temperaturas y bajas precipitaciones.

Fig 2. Embalse San Roque en proceso avanzado de eutrofización

Programa Permanente de Monitoreo del Embalse San Roque

Desde hace más de 20 años, el INA, lleva adelante el Programa Permanente de Monitoreo del Embalse San Roque y su Cuenca, con el objetivo de comprender la evolución de la eutrofización, sus causas, manifestaciones e impactos.

Desde 1999 hasta la actualidad, se han llevado a cabo más de 300 monitoreos, evaluando diversas variables físicas, químicas y biológicas del embalse. Con una frecuencia mensual, se monitorean seis sitios a nivel de superficie y en profundidad. Además, se realiza el seguimiento de los cuatro ríos tributarios (San Antonio, Cosquín, Las Mojarras y Los Chorrillos) y del emisario del embalse, el río Suquía.

El monitoreo de la calidad del agua ha permitido identificar variaciones significativas en parámetros como clorofila-a, fósforo total y nitrógeno, indicadores clave del estado trófico del embalse.

INA – Subgerencia Centro de la Región Semiárida (SCIRSA)

El Instituto Nacional del Agua (INA), a través de su Subgerencia Centro de la Región Semiárida (SCIRSA) y del área de Limnología Aplicada y Calidad de Agua (LAyCA), lleva a cabo actividades de investigación, estudio y asesoramiento técnico en el ámbito de la calidad del agua y la gestión de los recursos hídricos. Dentro de sus principales líneas de trabajo, se destacan los estudios sobre la eutrofización de lagos y la proliferación de floraciones algales, con especial atención en las cianotoxinas y sus efectos en la salud. Asimismo, realiza evaluaciones sobre contaminación bacteriana en aguas recreativas, estudios de impacto ambiental y análisis de sedimentos en embalses.

El área de Hidrología de SCIRSA se encarga del procesamiento y análisis de datos hidrometeorológicos e hidrológicos, con el propósito de generar información relevante para

estudios científicos y atender requerimientos de organismos externos. Desde 1990, gestiona una red telemétrica para la medición en tiempo real de variables clave en las cuencas monitoreadas, permitiendo un seguimiento preciso de la precipitación y otros factores que afectan la disponibilidad y calidad del agua en la región.

Sitios de Medición en el Embalse San Roque

El monitoreo del Embalse San Roque se lleva a cabo mensualmente en seis puntos estratégicos (Tabla 1 y Fig 3), cuatro de ellos ubicados en las desembocaduras de los ríos tributarios y dos dentro del lago. Estos sitios han sido seleccionados debido a su importancia en la dinámica del embalse, permitiendo evaluar variaciones en la calidad del agua y detectar cambios en su estado ambiental.

Fig 3. Imágen de los puntos de muestreo del ESR

Código	Nombre	Latitud (S)	Longitud (O)
C	Centro del lago	31º 22' 36''	64º 27' 54.6''
TAC	Área de presa	31º 22' 24.27''	64º 25' 58.25''
DLM	Desembocadura Arroyo las Mojarras.	31º 20' 13.77''	64º 28' 8.38''
DCQ	Desembocadura del Río Cosquín próxima al puente	31º 19' 11.34''	64º 27' 21.88''
	Desembocadura del Río Cosquín en Plaza Federal	31º 20' 37.61''	64º 28' 9.48''
	Desembocadura del Río Cosquín conjunta al Arrollo las Mojarras	31º 20' 23.30''	64º 28' 78''
DLC	Desembocadura del Arroyo Los Chorrillos.	31º 24' 1''	64º 29' 52.34''
DSA	Desembocadura del Río San Antonio próxima al puente	31º 24' 56.06''	64º 29' 49.03''
	Desembocadura del Río San Antonio próxima al hotel Hippocampus	31º 24' 35.24''	64º 29'' 48.36''

Tabla 1. Tabla detalle de la denominación y ubicación de los puntos de muestreo en el embalse usados en la actualidad.

Los puntos de monitoreo en las desembocaduras de los ríos tributarios son:

- **DCQ**: Desembocadura del río Cosquín.
- **DLM**: Desembocadura del arroyo Las Mojarras.
- **DLC**: Desembocadura del arroyo Los Chorrillos.
- **DSA**: Desembocadura del río San Antonio.

Dentro del lago, los sitios de medición más importantes son:

- **C**: Ubicado en el centro del embalse.
- **TAC**: Punto cercano a la presa, donde se encuentra la toma de Aguas Cordobesas S.A.

El monitoreo de calidad del agua en el Embalse San Roque se lleva a cabo mediante el uso de diversos instrumentos especializados que permiten la recolección y análisis de variables físicas, químicas y biológicas. Algunas de estas variables se miden directamente en campo (parámetros *IN SITU*) mediante equipos específicos, mientras que otras requieren la toma de muestra y el posterior análisis de laboratorio.

Medición de Parámetros *In Situ*

Las mediciones en campo se realizan mediante una sonda multiparamétrica, un dispositivo compuesto por sensores sumergibles conectados a una pantalla de lectura, que permite registrar en tiempo real distintos parámetros de calidad del agua. Este instrumento se sumerge a diferentes profundidades en cada uno de los puntos de monitoreo, proporcionando información detallada sobre la variabilidad de las condiciones fisicoquímicas en la columna de agua.

La profundidad es un parámetro clave en este proceso, ya que proporciona el contexto necesario para interpretar las demás variables. La sonda calcula la profundidad en metros en función de la presión ejercida por la columna de agua, asegurando precisión en la ubicación de los registros. Las mediciones comienzan con una primera lectura a 0,2 metros de profundidad (nivel subsuperficial) y continúan con descensos metro a metro hasta alcanzar un metro por encima del fondo del embalse.

En cada sitio de monitoreo se realizan **perfiles verticales de las variables *in situ* y de laboratorio**, los cuales permiten analizar la variación de las mismas a diferentes profundidades. La selección de estas profundidades sigue un criterio conjunto basado en la disponibilidad de luz y la temperatura, el cual varía según la época del año.

Cuando la columna de agua se encuentra en condición de estratificación, se analizan los niveles subsuperficial, fótico, epilimnion, hipolimnion y fondo. En cambio, durante la condición de mezcla, se consideran los niveles subsuperficial, fótico, intermedio y fondo. Cabe aclarar que, en este informe, solo se presentan y analizan las mediciones realizadas en los perfiles **subsuperficial y toma** (denominado TAC4). A continuación en la Figura 4 se presenta un diagrama de los perfiles y la tabla 2 que resume los sitios, su nomenclatura y la descripción del perfil según su condición térmica.

Tabla 2. Detalle de los diferentes perfiles y la descripción correspondiente para la condición térmica

Sitio	Código del perfil	Mezcla		Estratificación	
		Criterio	Descripción del perfil	Criterio	Descripción del perfil
TAC	TAC1	0,20m	**Subsuperficial**	0,20m	**Subsuperficial**
	TAC2	2,5m*Secchi(m) (zona fótica)	**Fótica**	2,5m* Secchi(m) (zona fótica)	**Fótica**
	TAC3	NA		1m por encima de termoclina	**Epilimnio**
	TAC4	Altura de toma conductos turbina	**Toma**	Altura de toma conductos turbina	**Toma**
	TAC5	1m del fondo	**Fondo**	1 metro del fondo	**Fondo**
C	C1	0,20m	**Subsuperficial**	0,20m	**Subsuperficial**
	C2	2,5m* Secchi(m) (zona fótica)	**Fótica**	2,5m* Secchi(m) (zona fótica)	**Fótica**
	C3	NA		1m por encima de termoclina	**Epilimnio**
	C4	Entre 2 y 4m	**Intermedia**	1m por debajo de termoclina	**Hipolimnio**
	C5	1m del fondo	**Fondo**	1m del fondo	**Fondo**
DSA	DSA1	0,20m	**Subsuperficial**	0,20m	**Subsuperficial**
	DSA5	1m del fondo	Fondo	1m del fondo	Fondo
DCQ	DCQ1	0,20m	**Subsuperficial**	0,20m	**Subsuperficial**
	DCQ5	1m del fondo	Fondo	1m del fondo	Fondo
DLM	DLM1	0,20m	**Subsuperficial**	0,20m	**Subsuperficial**
DLC	DLC1	0,20m	**Subsuperficial**	0,20m	**Subsuperficial**

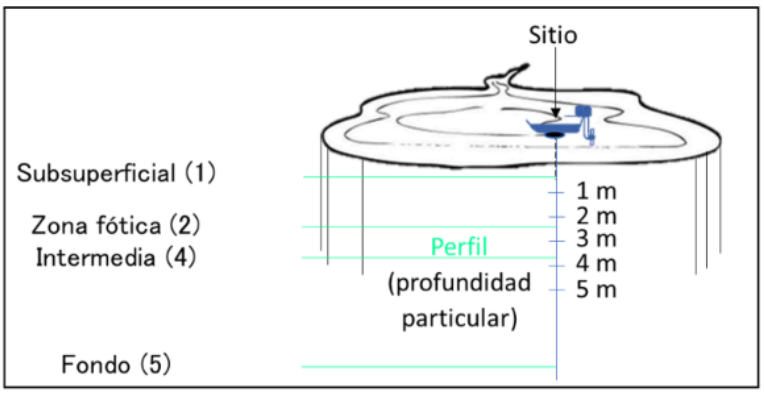


Fig 4. Diagrama de los diferentes perfiles

Monitoreo hidrológico y meteorológico

El monitoreo de las variables hidrológicas y meteorológicas en la cuenca del Embalse San Roque se realiza a través del Sistema Telemétrico de Transmisión de Datos Hidrológicos y Meteorológicos en Tiempo Real (STC), en funcionamiento desde 1986. Este sistema es operado por el INA-SCIRSA y permite la obtención continua de datos fundamentales para el análisis y la gestión del sistema hídrico *(Catalini y col., 2016).*

El STC cuenta con una estación central de recepción ubicada en la sede del INA-CIRSA en Villa Carlos Paz, desde donde se coordinan las tareas de monitoreo. La red de estaciones automáticas que lo componen está distribuida estratégicamente a lo largo de las subcuencas que alimentan el embalse, permitiendo el seguimiento en tiempo real de sus principales afluentes y del único efluente, el río Suquía. Las subcuencas más relevantes dentro de esta red incluyen las de los ríos San Antonio y Cosquín, que representan los tributarios más importantes del embalse, así como otras cuencas menores como las de los arroyos Las Mojarras y Los Chorrillos, que también influyen en la dinámica del sistema.

Siguiendo criterios geomorfológicos y las recomendaciones de la Organización Meteorológica Mundial (OMM), las estaciones están ubicadas en zonas clave de la cuenca para maximizar la representatividad de los datos. El objetivo principal es monitorear fenómenos críticos como las crecidas súbitas causadas por lluvias intensas, que representan una amenaza significativa para la seguridad de las poblaciones.

Las estaciones automáticas están equipadas con sensores especializados que registran de forma continua variables ambientales clave, que serán utilizadas como insumos para el entrenamiento del modelo de predicción. Entre estas variables se encuentra la **temperatura del aire**, que se mide mediante sensores del tipo termistor, capaces de detectar variaciones térmicas con alta precisión. Por otro lado, la **precipitación** se registra a través de pluviómetros de doble cubeta basculante, los cuales contabilizan la cantidad de lluvia caída con una resolución mínima de 1 milímetro (mm). Toda la información recolectada se transmite en tiempo real mediante radiofrecuencia en la banda VHF, ya sea de manera directa o a través de repetidoras, hacia la central de recepción y procesamiento ubicada en Villa Carlos Paz.

La frecuencia de medición varía según el sensor, la calibración y la estación, pero en general, para las variables de medición programada como la temperatura, los registros se toman en intervalos que oscilan entre 18 y 62 minutos. En estos casos, cada valor corresponde a una medición instantánea realizada en un momento específico.

En el caso de la precipitación, la medición es no programada y ocurre cada vez que se acumula 1 mm de lluvia. Este tipo de registro es acumulativo y diferencial, es decir, para

conocer la precipitación total de un día, se calcula la diferencia entre el primer y el último valor registrado durante ese período. Si no hay precipitaciones, el sensor emite dos señales diarias como “latido” para indicar que se encuentra operativo.

La continuidad y calidad de los datos recolectados por el sistema de monitoreo pueden verse afectadas por diversos factores. Entre los principales se encuentran fallas en el funcionamiento de los sensores, interrupciones en la transmisión de datos, roturas del equipamiento, así como también la imposibilidad de realizar tareas de mantenimiento o calibración periódica. Estas situaciones pueden generar vacíos en las series temporales de datos, que varían en duración: desde intervalos de algunas horas hasta períodos más extensos, como meses o incluso años completos sin registros disponibles.

En lo que respecta al sensor de precipitación, es fundamental comprender su lógica de funcionamiento para interpretar correctamente los datos registrados. Este instrumento utiliza un sistema de doble cubeta basculante, donde cada cubeta actúa como un pequeño recipiente que se llena con el agua de lluvia caída. Cuando una de ellas alcanza el peso correspondiente a 1 milímetro de precipitación, se vuelca automáticamente. En ese momento, la otra cubeta entra en acción, iniciando un nuevo ciclo de medición. Este mecanismo genera un reinicio del acumulador, por lo que el primer registro posterior al volcado es igual a cero, y el siguiente marca uno, señalando que se ha contabilizado el primer milímetro tras el reinicio. Esta dinámica debe ser tenida en cuenta al momento de analizar los datos, ya que afecta directamente la interpretación de la cantidad y continuidad de los eventos de precipitación.

Conceptos de Limnología

En este contexto, el análisis de diversas variables limnológicas permite comprender el estado y la dinámica del embalse, facilitando la interpretación de los procesos ecológicos que en ellos ocurren.

El objetivo de esta sección es proporcionar un marco conceptual para la posterior interpretación de los resultados, destacando las principales variables que determinan la calidad del agua desde el punto de vista biológico, físico y químico. A continuación, se detallan las más relevantes y resaltando en negrita aquellas que tienen un rol principal en este estudio.

La acidez o alcalinidad del agua está determinada por el pH, que representa la concentración de iones hidrógeno (H^+) en la solución. Este parámetro indica el carácter

ácido o básico del agua y puede afectar la disponibilidad de nutrientes y la toxicidad de ciertos compuestos *(Roldán Pérez & Ramírez Restrepo, 2008).*

El oxígeno disuelto (OD) es la cantidad de oxígeno disponible en el agua para los organismos acuáticos. Su concentración depende de la temperatura, la presión y los procesos biológicos, como la fotosíntesis y la respiración.

El porcentaje de saturación de oxígeno disuelto expresa la proporción de oxígeno en relación con el máximo que el agua puede contener en condiciones específicas de temperatura y presión. Es un indicador clave de la calidad del agua y de los procesos biogeoquímicos que ocurren en ella.

La conductividad eléctrica mide la capacidad del agua para conducir corriente eléctrica, lo que está directamente relacionado con la concentración de iones disueltos en la solución. Su valor depende de la presencia de sales y otros compuestos iónicos en el agua.

Los sólidos disueltos totales (SDT) representan la concentración total de sustancias minerales y orgánicas disueltas en el agua. Este parámetro es un indicador de la mineralización del agua y puede estar influenciado por aportes naturales o antropogénicos.

La **temperatura del agua** es un factor determinante en la dinámica de los ecosistemas acuáticos, ya que regula procesos metabólicos, la solubilidad de gases y la estratificación térmica de los cuerpos de agua.

El potencial óxido-reducción (ORP) está asociado con las reacciones redox, es decir, aquellos procesos en los que ocurre transferencia de electrones. Este parámetro es clave en la transformación de compuestos químicos y en la disponibilidad de ciertos nutrientes.

La turbidez se refiere al grado en que las partículas suspendidas en el agua dificultan la transmisión de la luz. Puede ser generada por materiales externos (turbidez alóctona) o por procesos internos del ecosistema acuático (turbidez autóctona). La turbidez afecta la fotosíntesis y la producción primaria, modificando el equilibrio ecológico del cuerpo de agua.

El color del agua depende de la luz reflejada y absorbida por las sustancias presentes en ella. Puede ser un indicador de la presencia de materia orgánica, fitoplancton o compuestos disueltos.

La **condición de estratificación** se refiere a la formación de capas en la columna de agua con propiedades físicas y químicas diferenciadas. Este fenómeno ocurre en ciertas épocas del año y está determinado principalmente por la temperatura y la densidad del agua.

Se distinguen tres capas principales:

- Epilimnion: Capa superficial cálida, de menor densidad y con mayor concentración de oxígeno debido al contacto con el aire y la acción del viento.
- Metalimnion (o Termoclina): Capa intermedia donde la temperatura desciende abruptamente con la profundidad, actuando como una barrera para el intercambio de oxígeno y nutrientes entre las capas superior e inferior.
- Hipolimnion: Capa profunda, más fría y densa, con bajos niveles de oxígeno. En esta zona se acumulan nutrientes y materia orgánica en descomposición debido a la falta de mezcla con las capas superiores.

Las floraciones algales, especialmente las de cianobacterias, están estrechamente relacionadas con la estratificación del agua, ya que durante este proceso los nutrientes como el fósforo y el nitrógeno pueden acumularse en el hipolimnion. Si la estratificación se rompe, estos nutrientes se liberan a la superficie, favoreciendo un crecimiento acelerado de algas. Además, el epilimnion, al ser más cálido y recibir mayor luz solar, crea condiciones óptimas para su proliferación, mientras que la reducción de oxígeno en el hipolimnion puede facilitar la liberación de nutrientes desde el sedimento, favoreciendo el desarrollo de ciertas algas en condiciones de hipoxia.

Cuando la estratificación desaparece, el cuerpo de agua entra en condición de mezcla, permitiendo la redistribución de nutrientes y oxígeno a lo largo de la columna de agua, lo que puede potenciar la aparición de floraciones algales. Se considera que un cuerpo de agua está estratificado cuando la temperatura varía más de 1°C entre un metro y otro en profundidad, condición que se evalúa mediante mediciones en distintos puntos de la columna de agua.

El **nitrógeno y sus variantes** son esenciales para la síntesis de proteínas y otros compuestos biológicos. Se encuentra en el agua en diversas formas, incluyendo iones nitrato (NO_3^-), iones amonio (NH_4^+) y nitrógeno molecular (N_2), y su disponibilidad puede influir en la productividad primaria del ecosistema.

El **fósforo y sus variantes** es otro nutriente clave en los ecosistemas acuáticos. Desempeña un papel fundamental en el metabolismo biológico y suele ser el factor limitante de la producción primaria en estos ambientes. Se encuentra en diferentes formas, como fósforo reactivo soluble y fósforo total, cuya determinación varía según el tratamiento de la muestra.

La **clorofila-a** es un pigmento fotosintético presente en algas y cianobacterias. Su concentración en el agua es un indicador de la biomasa fitoplanctónica y, por lo tanto, de la productividad del sistema acuático.

El carbono orgánico total (COT) representa la cantidad total de materia orgánica disuelta y particulada en el agua. Su origen puede ser natural o antropogénico, y su presencia influye en la calidad del agua y en los procesos biogeoquímicos.

El **fitoplancton** está compuesto por organismos fotosintéticos microscópicos, como ciertas bacterias y algas, que forman la base de la cadena trófica en los ecosistemas acuáticos. Su abundancia y composición pueden reflejar cambios en la calidad del agua y en la disponibilidad de nutrientes.

Estas variables limnológicas se encuentran interconectadas y determinan el funcionamiento del ecosistema acuático. En la siguiente figura (Fig 5), se presenta un esquema que resume algunas de sus principales interacciones y cómo influyen en la calidad del agua del ESR.

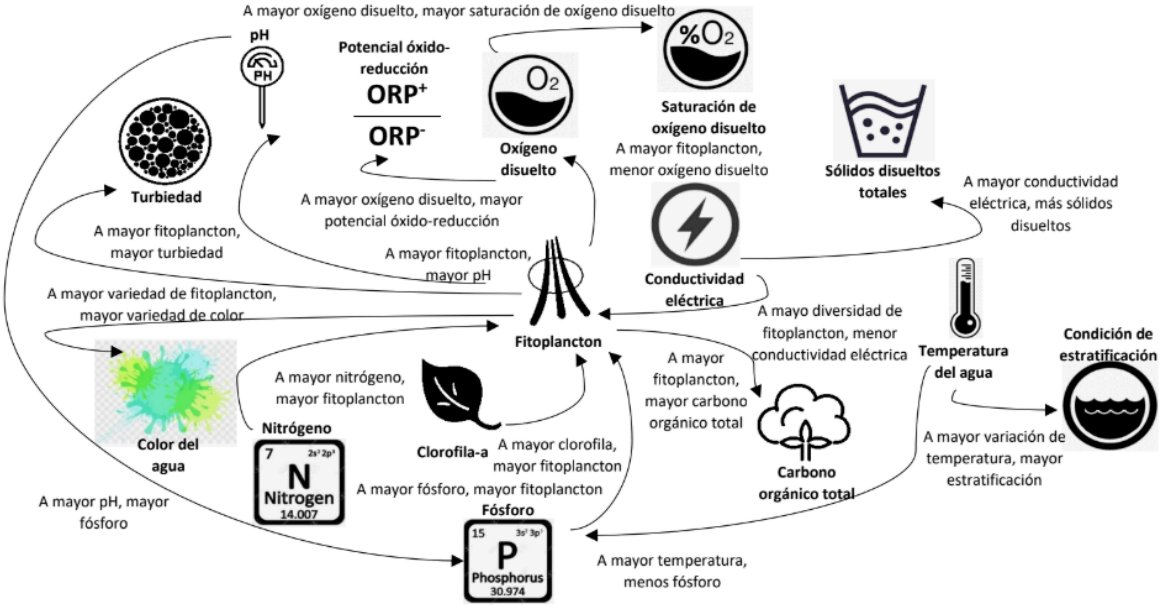


Fig. 5. Diagrama de algunas de las principales relaciones entre las variables limnológicas. Fuente: Pablo Andreoni (2020).

Conceptos de Meteorología

Las variables meteorológicas juegan un rol fundamental en la caracterización del clima y en la dinámica de los ecosistemas acuáticos. A continuación, se presentan algunas de las variables más relevantes. Se destaca en negrita aquellas que tienen un papel principal en este estudio.

La **temperatura** se define como una medida de la velocidad promedio del movimiento de los átomos y moléculas, donde valores elevados de temperatura corresponden a mayores velocidades promedio de las partículas y viceversa. En este estudio, se analizará específicamente la temperatura del aire, la cual refleja el grado de calor o frío de los gases que lo componen.

La radiación solar es la energía transferida desde el sol a un objeto. Se puede conceptualizar como un flujo continuo de partículas o fotones, los cuales son paquetes discretos de energía.

La humedad representa la cantidad de vapor de agua presente en el aire. En particular, este trabajo considera la humedad relativa, la cual se define como el cociente entre la cantidad real de vapor de agua en el aire y la máxima cantidad posible de vapor de agua que puede contener a una temperatura y presión específicas.

La evaporación es el proceso de cambio de estado del agua, de líquido a gaseoso, influenciado por factores como la temperatura, la humedad y la radiación solar.

La **precipitación** se refiere a cualquier cantidad de agua, ya sea en estado líquido o sólido, que cae desde una nube y alcanza la superficie terrestre.

La presión atmosférica es la fuerza ejercida por la masa de aire sobre una región determinada. Se calcula como el producto de la temperatura, la densidad del aire y una constante. Su variación con la altitud es inversamente proporcional, es decir, a mayor altitud, menor presión atmosférica.

El viento es el movimiento en masa del aire generado por diferencias en la presión atmosférica. Se caracteriza por dos magnitudes: la velocidad, que expresa la distancia recorrida por unidad de tiempo, y la dirección, determinada por el punto de origen y el destino del flujo de aire.

Estas variables meteorológicas están estrechamente interrelacionadas y su interacción tiene un impacto significativo en los procesos ambientales. En la siguiente figura, se presenta un diagrama simplificado que ilustra algunas de las principales relaciones entre las variables mencionadas.

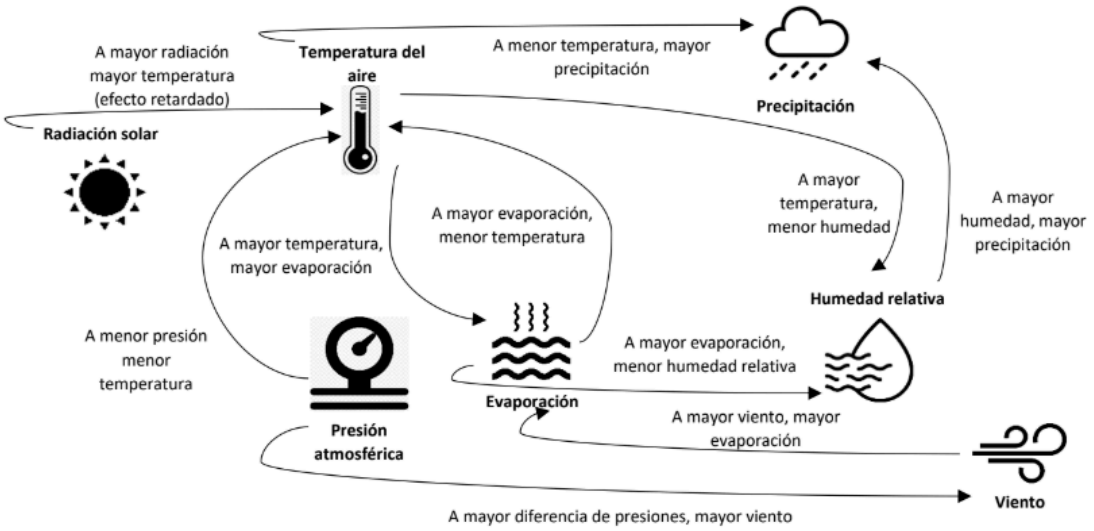


Fig. 6. Diagrama de algunas de las principales relaciones entre las variables meteorológicas. Fuente: Pablo Andreoni (2020).

A NIVEL TÉCNICO DE LA SOLUCIÓN

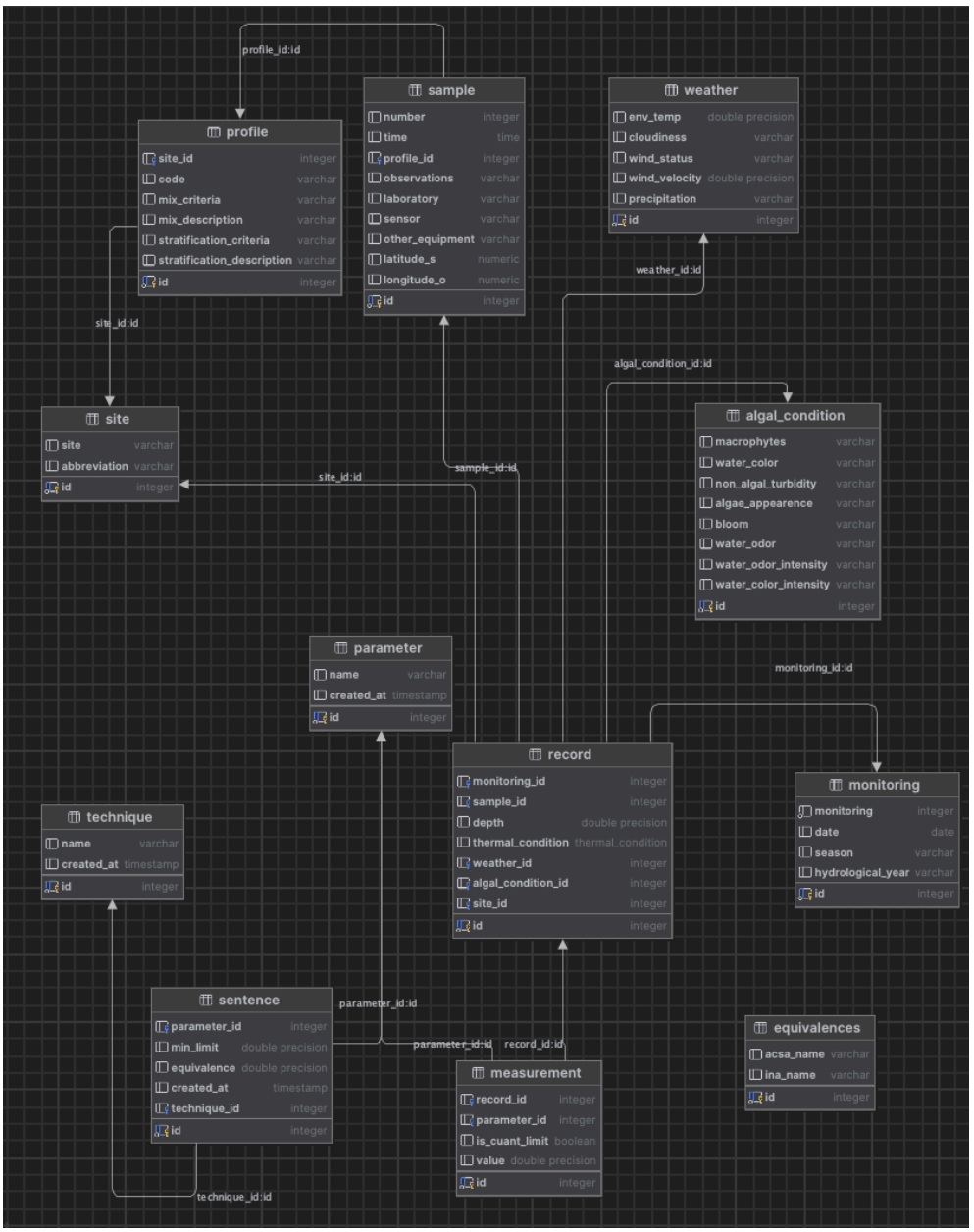
Previo al desarrollo del modelo predictivo, se llevó a cabo una etapa de análisis e investigación técnica. Esta etapa resultó fundamental para comprender el contexto tecnológico existente, evaluar herramientas y metodologías alternativas, y sentar las bases de una solución integral que responda a los desafíos del monitoreo y predicción de floraciones algales en el Embalse San Roque.

El punto de partida conceptual y técnico del presente proyecto se encuentra en el trabajo final desarrollado por Barafani y Dubowez (2023), titulado *"Solución tecnológica para el aprovechamiento de datos del Instituto Nacional del Agua"*. Este proyecto sentó las bases del sistema actual de gestión de datos del INA, implementando una arquitectura ETL (Extracción, Transformación y Carga) que automatiza el flujo de información desde la toma de mediciones manuales *in situ* y los resultados de laboratorio, hasta su consolidación en la base de datos de calidad de agua. Asimismo, extendió este proceso a los datos provenientes de sensores automáticos, integrando sus lecturas en tiempo real dentro de la base de datos hidrometeorológica. De esta forma, el proyecto estableció un esquema para centralizar la información del monitoreo ambiental.

A continuación, se describe la estructura, contenido y funcionamiento de ambas bases de datos con el fin de entender su rol central en la arquitectura del sistema de predicción.

1. Calidad de Agua ("Water-Quality"):

Esta Base de Datos (BD) se denomina “water quality” y almacena los datos de la calidad del agua. Posee información actualmente desde 12-1996 hasta 12-2023. Tiene un total de 328 registros de mediciones, registradas con una frecuencia mensual, entre las fechas especificadas.



Se describen las tablas que contiene:

1. **site (Sitios de muestreo):** Almacena los diferentes sitios donde se toman muestras de agua.

2. **profile (Perfiles de muestreo):** Representa diferentes perfiles de estratificación del agua, se relaciona con site porque indica a qué sitio pertenece el perfil.

3. **monitoring (Eventos de monitoreo):** Registra cada evento de monitoreo realizado, registrando fecha, estación del año y año hidrológico.

4. **record (Registros de monitoreo):** Contiene información detallada de cada monitoreo realizado.

    - Relaciones principales con :

        - monitoring, indica el evento al que pertenece.

        - sample, indica la muestra asociada.

        - site, indica el sitio del monitoreo.

5. **sample (Muestras de agua):** Describe cada muestra recolectada en los eventos de monitoreo. Indica a qué perfil de muestreo pertenece.

6. **weather (Condiciones climáticas):** Registra los datos climáticos en el momento del muestreo.

7. **algal_condition (Condiciones algales):** Almacena características del agua en cuanto a algas y turbidez.

8. **parameter (Parámetros medidos):** Define los parámetros que se miden en la calidad del agua.

9. **measurement (Mediciones):** Contiene los valores obtenidos en cada monitoreo para los distintos parámetros.

    - Detalla sobre el parámetro:

        - value: Valor medido.

        - is_over_limit: Indica si el valor excede el límite permitido.

10. **sentence (Normativas y técnicas):** Contiene los límites permitidos para cada parámetro según una técnica determinada.

    - Relación con:

        - parameter, indica qué parámetro aplica.

        - technique, indica la técnica usada.

- Campos principales:
    - Valores mínimos y máximos permitidos.

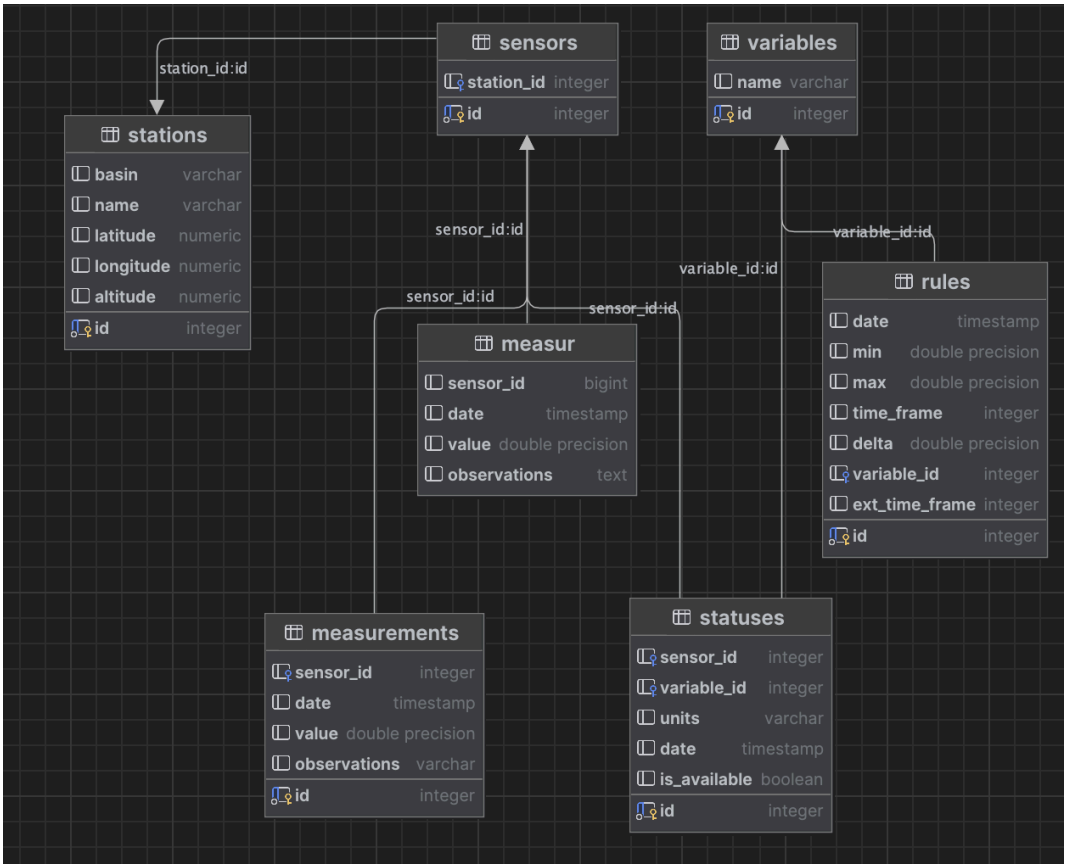
11. **technique (Técnicas de medición):** Define las técnicas utilizadas para la medición de parámetros.
12. **equivalences (Equivalencias de nombres):** Relaciona las nomenclaturas utilizadas por el INA con las utilizadas por ACSA (Aguas Cordobesas S.A.) tanto para parámetros como para sitios de medición.

Relaciones clave entre tablas

- **site** → **profile**: Cada sitio tiene uno o más perfiles.
- **profile** → **sample**: Cada perfil puede estar asociado con varias muestras.
- **monitoring** → **record**: Cada evento de monitoreo genera múltiples registros.
- **record** → **measurement**: Cada registro tiene mediciones de diferentes parámetros.
- **record** → **weather**: Cada monitoreo se asocia con una condición climática específica.
- **record** → **algal_condition**: Los registros incluyen información sobre condiciones algales.
- **measurement** → **parameter**: Cada medición está vinculada a un parámetro específico.
- **sentence** → **parameter & technique**: Define los límites permitidos para cada parámetro según una técnica específica.

2. Hidrometeorológica (“Alerts”)

Denominada “Alerts”, almacena datos hidrometeorológicos, posee información desde 01-1986 hasta el 02-2019. Tiene un total de 10.258.348 registros tomados con frecuencia diaria o menor, dependiendo de la variable monitoreada.



Se describen las tablas que contiene:

1. stations (Estaciones de monitoreo): Contiene información sobre las estaciones hidrometeorológicas, describiendo la cuenca hidrográfica a la que pertenece (San Antonio, Cosquín, Anisacate, entre otros), nombre y ubicación geográfica.

2. sensors (Sensores): Representa los sensores instalados en cada estación.

3. variables (Variables medidas): Lista las variables hidrometeorológicas medidas por los sensores (ej. temperatura, precipitación).

4. statuses (Estados de sensores y variables): Registra la disponibilidad de un sensor y la unidad de la variable medida y la fecha de registro del estado.

5. measurements (Mediciones): Contiene los valores registrados por cada uno de los sensores, junto con la fecha, hora y observaciones de la medición.

6. rules (Reglas para control de calidad de mediciones): Define límites y condiciones para validar las mediciones.

- Relación con:

    - variables → indica a qué variable se aplica la regla.

- Campos principales:
    - date → Fecha de creación de la regla.
    - min, max → Valores mínimos y máximos permitidos.
    - time_frame, ext_time_frame → Intervalos de tiempo para validar datos.
    - delta → Diferencia permitida entre mediciones consecutivas.

7. measur (Posible duplicado de measurements): Parece ser una versión incompleta de la tabla measurements.

Relaciones clave entre tablas

- **stations** → **sensors**: Cada estación puede tener múltiples sensores.
- **sensors** → **statuses**: Un sensor puede estar asociado a más de una variable.
- **variables** → **statuses**: Cada variable se puede medir en más de un sensor.
- **sensors** → **measurements**: Los sensores registran mediciones periódicamente.
- **variables** → **rules**: Se definen reglas de validación para cada variable.

Flujo de Datos

Para comprender el proceso de recopilación de datos actual, se detalla el flujo que recorren los mismos, desde su origen hasta su incorporación en las bases, identificando las dos fuentes de información:

1. **Mediciones manuales *IN SITU* y toma de muestras:**
    - Se realizan mediciones mensuales en distintos puntos del embalse y sus afluentes.
    - Se registran datos *in situ* como temperatura del agua, oxígeno disuelto y turbidez.
    - Se toman muestras de agua para su análisis en laboratorio, donde se determinan variables físico-químicas y biológicas.

- Los resultados del laboratorio más los datos de la planilla registrados *IN SITU*, se transcriben en planillas de Excel y posteriormente se cargan en un sistema desarrollado por Barafani y Dubowez (2023).

- Este sistema ejecuta un proceso ETL (Extracción, Transformación y Carga). El ETL toma los datos crudos provenientes de las planillas de Excel y se encarga de procesarlos y normalizarlos, asegurando su calidad y consistencia antes de almacenarlos en la base de datos.

2. **Sensores hidrometeorológicos automáticos:**

   - Se cuenta con sensores distribuidos en distintas estaciones de monitoreo dentro de la cuenca del ESR.

   - Estos sensores registran variables meteorológicas como temperatura del aire, humedad relativa, precipitación, velocidad y dirección del viento, presión atmosférica y nivel del embalse.

   - La información es enviada a través de sistemas de transmisión automática, vía **radio frecuencia VHF**, hacia la central de procesamiento del INA–SCIRSA en Villa Carlos Paz. Aquí son capturados por un software que los transforma en archivo plano, para que luego sean revisados, comentados y subidos a la **Base de Datos Hidrometeorológica**, por un experto (Fig 7).

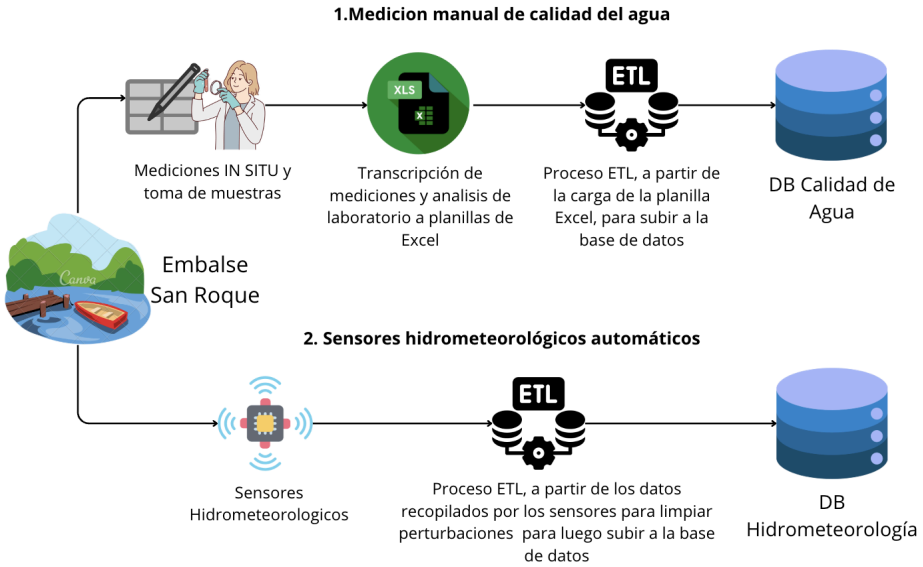


Fig 7. Elaboración propia. Flujo de datos hasta el almacenamiento en las bases de datos.

Análisis de herramientas tecnológicas para el desarrollo

Frontend

Se analizan distintos frameworks para el desarrollo del frontend, principalmente React, Vue.js y Angular:

- React: Biblioteca JavaScript basada en componentes. Ventajas: comunidad activa, gran cantidad de recursos, buena integración con bibliotecas de visualización como D3.js o Plotly.
- Vue.js: Framework progresivo y más accesible para principiantes. Ventajas: curva de aprendizaje baja, sintaxis clara. Desventajas: menor soporte para bibliotecas de ciencia de datos y menor adopción en entornos científicos.
- Angular: Framework completo orientado a grandes aplicaciones. Ventajas: ofrece una arquitectura integral, ideal para equipos grandes. Desventajas: mayor complejidad, rigidez en su estructura y menor agilidad para prototipos rápidos.

Backend

Se investigan opciones como Flask, FastAPI y Django:

- Flask: Microframework en Python. Ventajas: liviano, flexible, ideal para proyectos con lógica personalizada y APIs REST simples. Desventajas: requiere más configuración manual para tareas como autenticación, administración o documentación.
- FastAPI: Framework moderno con tipado fuerte y documentación automática. Ventajas: alta velocidad, integración con OpenAPI, ideal para APIs complejas. Desventajas: menor madurez y comunidad aún en expansión comparado con Flask.
- Django: Framework completo orientado a aplicaciones grandes. Ventajas: incluye ORM, autenticación, administración. Desventajas: estructura más rígida, innecesaria para sistemas enfocados en predicción y servicios REST.

Modelos: Clasificación vs. Regresión

Se evalúa la conveniencia de utilizar modelos de clasificación o regresión para las variables objetivo:

- Regresión (salida continua): útil para predecir cantidades exactas como cel/l o µg/l. Ventajas: mayor detalle, útil para análisis científicos y simulaciones. Desventajas: requiere definir umbrales para generar alertas categóricas, y es más sensible a outliers.

  Modelos considerados:

    - Regresión lineal: Modelo estadístico que ajusta una línea recta para predecir una variable continua a partir de una o más variables independientes.
    - Random Forest Regressor: Ensamble de múltiples árboles de decisión entrenados sobre subconjuntos aleatorios de datos y características, que promedian sus predicciones para estimar una variable continua.
    - XG Boost (Extreme Gradient Boosting): Algoritmo de ensamble basado en boosting de árboles que optimiza la predicción de variables continuas minimizando el error de forma iterativa y eficiente.
    - Red neuronal (MLP Regressor): Modelo de aprendizaje profundo compuesto por capas densamente conectadas que captura relaciones complejas y no lineales para predecir variables continuas.
    - LSTM (Long Short-Term Memory): Variante de red neuronal recurrente (RNN) diseñada para aprender dependencias a largo plazo en datos secuenciales, ideal para modelar series temporales como mediciones ambientales en el tiempo, gracias a su capacidad de conservar información relevante durante múltiples pasos temporales.

- Clasificación (salida categórica): útil para etiquetar situaciones como bajo, medio o alto riesgo. Ventajas: fácil interpretación, útil para sistemas de alerta operativa. Desventajas: pierde precisión cuantitativa.

  Modelos considerados:
    - Regresión logística multiclase: Extensión de la regresión logística que modela probabilidades para múltiples clases mediante funciones logísticas generalizadas.
    - Random Forest Classifier: Ensamble de árboles de decisión que votan de forma agregada para clasificar observaciones en categorías discretas.
    - MLP Classifier: Red neuronal multicapa que utiliza retropropagación para aprender a clasificar instancias en distintas clases a partir de patrones no lineales.
    - SVM (Support Vector Machine): Algoritmo que encuentra el hiperplano óptimo que separa clases maximizando el margen entre ellas, eficaz para problemas lineales y no lineales mediante kernels.

Métricas de Evaluación

Se analizan métricas según el tipo de modelo:

- Para regresión:

MAE (Mean Absolute Error): fácil de interpretar, menos sensible a outliers. Desventaja: puede subestimar errores grandes.

MSE (Mean Squared Error): penaliza errores grandes, útil para detectar desviaciones significativas. Desventaja: difícil de interpretar en unidades reales, especialmente cuando los valores están muy dispersos.

R^2 (Coeficiente de Determinación): mide qué proporción de la variabilidad total de la variable objetivo es explicada por el modelo. Ventaja: intuitiva para evaluar el poder explicativo global.

- Para clasificación:

  F1 Macro: útil para clases desbalanceadas, ya que considera tanto precisión como recall en cada clase y promedia los resultados. Desventaja: puede penalizar fuertemente un mal desempeño en una sola clase.

  ROC-AUC Macro: mide la capacidad del modelo para distinguir entre clases.

  Desventaja: requiere probabilidades bien calibradas y puede no ser tan útil en presencia de muchas clases sin orden natural.

  Accuracy (Precisión global): indica el porcentaje total de predicciones correctas. Ventaja: fácil de interpretar y comunicar. Desventaja: engañosa cuando hay desbalance de clases, ya que un modelo que predice siempre la clase mayoritaria puede tener alta accuracy sin ser útil.

Imputación de Datos Faltantes

El tratamiento de datos faltantes es crítico debido a la naturaleza incompleta de muchos registros ambientales. Se investigan varias técnicas:

- Promedio por sitio y estación: fácil de implementar. Desventajas: ignora relaciones multivariadas.
- KNN Imputer: considera similitud con otras observaciones. Ventajas: capta patrones locales. Desventajas: sensible a la escala de los datos.
- Regresión lineal: permite imputar en base a relaciones estadísticas. Desventajas: asume relaciones lineales.
- Random Forest y XGBoost Imputer: detectan relaciones no lineales. Ventajas: alta precisión. Desventajas: mayor complejidad computacional, requiere ajuste de hiperparámetros.

Infraestructura: Contenerización e Integración Continua

Se analiza el uso de tecnologías de despliegue moderno:

- Docker: permite encapsular el entorno completo de ejecución (código, bibliotecas, variables de entorno, dependencias) en un contenedor portátil. Ventajas: asegura reproducibilidad entre entornos, facilita la escalabilidad en servidores o la nube, permite pruebas en entornos aislados.
- GitHub Actions: plataforma de CI/CD integrada a GitHub, que permite ejecutar scripts automáticos ante eventos como push, pull request o cron. Ventajas: ideal para flujos automáticos como tests, entrenamiento de modelos, generación de imágenes Docker, y despliegue automático.
- GitLab CI/CD: alternativa completa que permite configurar pipelines directamente desde archivos .gitlab-ci.yml. Ventajas: ofrece runners propios o personalizados, mejor integración para entornos privados, visualización detallada de pipelines. Desventajas: menor integración directa con repositorios GitHub; curva de aprendizaje más empinada

Modelos precedentes de predicción

En los últimos años, la aplicación de modelos de machine learning (ML) en la predicción de floraciones algales ha mostrado avances significativos. Diversos estudios han explorado enfoques basados tanto en datos ambientales como en imágenes satelitales o microscópicas, proponiendo soluciones adaptables a distintas escalas y fuentes de información.

Uno de los trabajos más relevantes en este campo fue desarrollado por investigadores de la Universidad Complutense y Autónoma de Madrid *(Fournier y col., 2024)*, quienes diseñaron un sistema de alerta temprana para la predicción de cianobacterias en el embalse Cuerda del Pozo, utilizando redes neuronales LSTM (Fig 8, 9 y 10). Este modelo, entrenado con variables clave como temperatura del agua, clorofila-a y ficocianina, alcanzó una precisión del 90% incluso en predicciones con horizontes de 28 días, demostrando la eficacia de modelos secuenciales en contextos ambientales con alta variabilidad temporal.

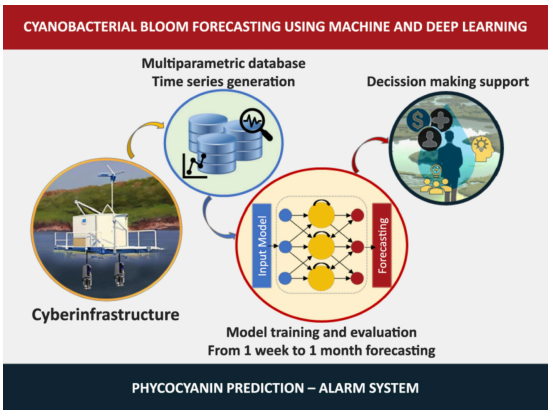


Fig. 8. Diagrama sobre el proceso del sistema

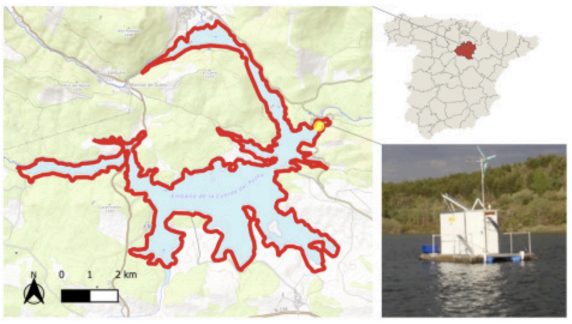


Fig. 9. Imagen del embalse Cuerda del Pozo, España

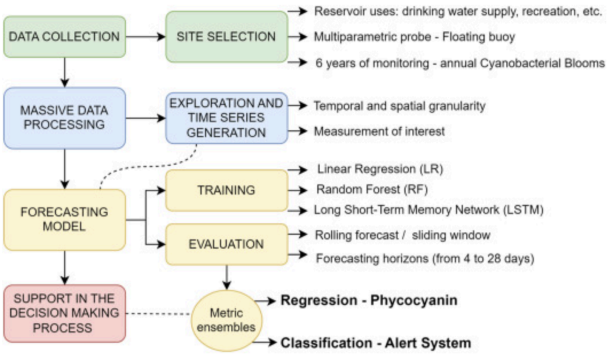


Fig. 10. Flujo del modelo, desde la recolección de datos hasta la salida del modelo

En Argentina, un proyecto presentado en las Jornadas Argentinas de Informática *(Rosa y col., 2024)* abordó la clasificación automática de imágenes de cianobacterias utilizando redes convolucionales (CNNs) como ResNet y GoogleNet. Esta investigación validó la posibilidad de desarrollar sistemas de diagnóstico automático con datasets locales, lo cual resulta especialmente relevante para regiones con escasez de datos estandarizados y recursos humanos especializados.

Por su parte, un estudio publicado por *Hwang y col.* (2023) exploró el uso combinado de XGBoost y redes neuronales para predecir la abundancia de clorofila-a y cianobacterias en lagos, confirmando la relevancia de variables como la temperatura, el fósforo y el nitrógeno. Además, se destacaron los modelos basados en árboles como herramientas robustas y fácilmente interpretables para tareas de regresión en ambientes acuáticos.

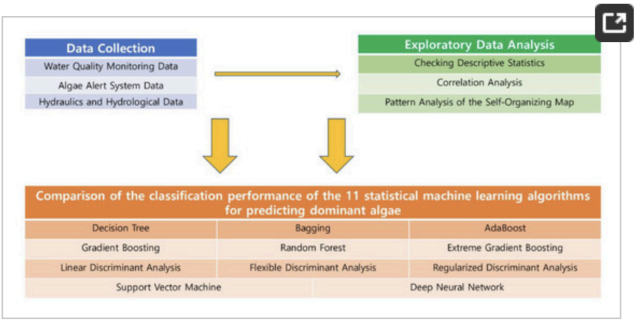


Figure 1. Methodological flowchart used in this study.

Fig. 11. Tabla de comparación de performance de algoritmos de ML

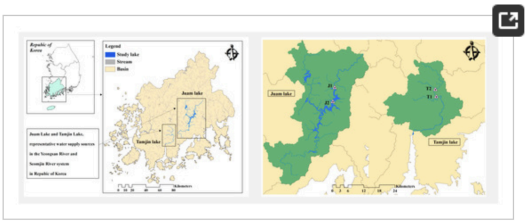


Figure 2. Sampling sites at Juam Lake and Tamjin Lake.

Fig. 12. Mapa de sitios en el Lago Juam y Tamjin, República de Corea (Corea del Sur)

Estos antecedentes evidencian el **alto interés científico y tecnológico** que genera la predicción de floraciones algales a nivel global, y aportan **referencias concretas** que ayudan a orientar el diseño de este proyecto. Estos desarrollos previos sirven como punto de partida para identificar **algoritmos con potencial para probar**, como XGBoost, LSTM, Random Forest, entre otros. Además, ayuda a analizar estrategias para el sistema, que incluyen el preprocesamiento de datos, la selección de variables críticas (como nutrientes, temperatura y clorofila), y la necesidad de incorporar herramientas de interpretabilidad. En conjunto, ofrecen una base sólida de proyectos similares, para entender desde dónde iniciar.

PROPUESTA DE SOLUCIÓN

ALCANCE FUNCIONAL

La presente investigación parte de la hipótesis de que la proliferación de cianobacterias en el Embalse San Roque está influenciada significativamente por la calidad del agua y por las variables hidrometeorológicas. En este contexto, se busca responder a la pregunta: **¿Cómo influyen la calidad del agua y las variables hidrometeorológicas en la aparición de cianobacterias?**

Se plantea que, mediante la aplicación de un modelo de machine learning entrenado con datos históricos y en tiempo real, será posible identificar patrones relevantes y generar predicciones precisas sobre la ocurrencia de estos eventos. Esta aproximación permitiría optimizar la gestión ambiental del embalse, anticipar situaciones críticas y mejorar la toma de decisiones mediante el uso de herramientas automatizadas basadas en inteligencia artificial.

El modelo analizará una serie de **variables de entrada**, agrupadas en tres categorías:

- Hidrológicas: Precipitación acumulada (3 días), nivel del lago (cota), temperatura máxima y mínima del aire.

- Fisico-químicas: Temperatura del agua, del aire, fósforo hidrolizable total (PHT), fósforo reactivo soluble (PRS), condición térmica y nitrógeno inorgánico total (NIT - calculado como la sumatoria de nitrito, nitrato y amonio).

- Biológicas: Clorofila a (µg/l), Total de algas (abundancia total), total de cianobacterias (sumatoria de todas los géneros de cianobacterias registrados), dominancia de cianobacterias % de cianobacterias sobre la abundancia total).

Estas variables se miden tanto en la superficie como en diferentes profundidades de la columna de agua. Sin embargo, en todos los sitios de monitoreo se consideran únicamente los valores correspondientes a mediciones superficiales. En el caso particular del sitio TAC, se utilizan tanto los datos superficiales como los obtenidos en la profundidad correspondiente a la toma de agua para potabilización.

Las **variables objetivo** que el modelo buscará predecir son:

- Cantidad de células de cianobacterias (cel/L)

- Concentración de clorofila (µg/L)

- Dominancia de cianobacterias (%)

Las predicciones no serán valores numéricos continuos, sino clases categóricas predefinidas que representan rangos de riesgo (por ejemplo: bajo, medio y alto). Esto convierte al modelo en uno de clasificación, lo que facilita la interpretación de los resultados para la gestión ambiental del embalse. Los umbrales de clasificación serán definidos en base a criterios técnicos establecidos por el INA.

Dado que las cianobacterias dependen de la disponibilidad de nutrientes (fósforo y nitrógeno inorgánico total), la temperatura del agua, la estratificación térmica y las precipitaciones acumuladas, se espera que la interacción de estas variables tenga un alto grado de correlación con las variables objetivo. En particular:

- Un aumento en nitrógeno y fósforo promoverá el crecimiento del fitoplancton y, en particular, de las cianobacterias.

- La temperatura del agua y la estratificación térmica influirán en la disponibilidad de nutrientes, afectando la proliferación de algas.

- La precipitación acumulada y el nivel del lago modificarán la concentración de nutrientes y la estabilidad térmica del embalse.

Si el modelo logra establecer relaciones significativas entre estas variables de entrada y las variables de salida, entonces será posible anticipar eventos de floración algal, mejorando la gestión del embalse y optimizando los recursos destinados a su monitoreo y control.

Requerimientos del Sistema

Los requerimientos del sistema se dividen en **requerimientos funcionales**, que describen las funcionalidades esenciales del modelo de predicción, y **requerimientos no funcionales**, que establecen criterios de calidad y desempeño del sistema.

Requerimientos Funcionales:

1. Entrada de Datos:

   RF1.1: El modelo debe aceptar datos históricos y en tiempo real desde dos bases de datos PostgreSQL:

   - Base de datos de Calidad de Agua (monitoreo mensual).
   - Base de datos de Hidrometeorología (datos en tiempo real de sensores).

   RF1.2: Los datos de entrada deben incluir variables que tienen influencia en la floración de algas: temperatura del agua, del aire, condición térmica, nitrito, nitrato, amonio, fósforo soluble, fósforo total, niveles de agua del embalse y precipitaciones. Así como de las variables objetivo.

   RF1.3: El modelo debe calcular variables adicionales basadas en los datos de las bases de datos:

   - Cianobacterias Total = Sumatoria de todos los géneros presentes de cianobacterias.
   - Nitrógeno Inorgánico Total = Suma de nitrito, nitrato y amonio.
   - Dominancia de cianobacterias (%) = (Cianobacterias Total (cel/L) * 100) / Total de Algas (Sumatoria de todos los grupos de algas expresados en cel/L).

   RF1.4 Si una base de datos no está disponible, el modelo debe notificar al usuario con una alerta.

2. Predicción de Variables de Salida

RF2.1: El sistema debe clasificar los resultados de las siguientes variables de salida en categorías de riesgo:

- Cantidad de células de cianobacterias (Cianobacterias Total), (Vigilancia, Alerta, Emergencia)
- Concentración de clorofila (µg/L), (Vigilancia, Alerta, Emergencia).
- Dominancia de cianobacterias (%), (No dominante, Dominante)

RF2.2: Las predicciones generadas deben ser a un intervalo de tiempo mensual.

3. Automatización del Proceso

RF3.1: El sistema debe correr un proceso de actualización, para el reentrenamiento de los modelos, que debe ejecutarse automáticamente al recibir una notificación de actualización o eliminación o ingreso de datos, mediante triggers en las bases de datos.

RF3.2: Las predicciones generadas deben almacenarse automáticamente en la base de datos.

4. Interfaz de Usuario

RF4.1: El sistema debe proporcionar una interfaz gráfica interactiva que permita elegir para qué sitio/s se realizará la predicción y visualizar los resultados de la misma.

RF4.2: Las etiquetas clasificatorias, deben resaltarse en la interfaz, con colores para una rápida interpretación.

5. Mantenimiento del Modelo

RF5.1: El sistema debe permitir la actualización y reentrenamiento del modelo con nuevos datos para mejorar su precisión.

RF5.2: Debe incluir herramientas para evaluar el rendimiento del modelo, para conocer el grado de fiabilidad de la predicción.

Requerimientos No Funcionales

1. Precisión y Fiabilidad:

RNF1.1: El modelo debe mantener un rendimiento mínimo medido por:

- Curva ROC-AUC ≥ 0,50.
- F1-Score ≥ 0,30.

Y las métricas de rendimiento se deben poder revisar periódicamente.

RNF1.2: El sistema debe ser confiable, garantizando una disponibilidad mínima del 99,5% mensual (uptime) y capaz de manejar grandes volúmenes de datos sin pérdida de información, en conjunto ambas bases de datos, contienen 10 millones de registros.

2. Escalabilidad:

   RNF2: El sistema debe ser escalable para manejar un creciente volumen de datos, proyectado de 26.000 nuevos registros por mes, entre los registros de calidad de agua y de hidrometeorología, sin que el tiempo de predicción supere en más de un 20% el tiempo base (≤3 s).

3. Desempeño:

   RNF3: El tiempo de respuesta del modelo para generar una predicción, no debe superar los 3 segundos desde la solicitud del usuario, garantizando una experiencia fluida y compatible con la toma de decisiones operativas.

4. Seguridad:

   RNF4: El sistema debe garantizar la confidencialidad y protección de la información, asegurando su acceso únicamente al personal autorizado. La aplicación debe estar alojada en un servidor interno del INA y el acceso se realizará exclusivamente mediante la red privada virtual (VPN) institucional.

5. Usabilidad:

   RNF5.1: La interfaz debe ser intuitiva y accesible para usuarios con diferentes niveles de experiencia técnica.

6. Compatibilidad:

   RNF6: Debe integrarse con los sistemas existentes en el INA.

Historias de usuario

HU1	Integración de datos en tiempo real y históricos
Requerimiento Relacionado	RF1.1, RF1.2, RNF2
Historia de Usuario	Como administrador del sistema, quiero que el modelo acepte datos históricos y en tiempo real desde las bases de datos PostgreSQL, para asegurar que las predicciones estén basadas en información actualizada y reflejen las condiciones más recientes del embalse.
Criterios de Aceptación	1. El sistema se conecta automáticamente a ambas bases de datos PostgreSQL. 2. Los datos incluyen todas las variables clave para la predicción. 3. Si una base de datos no está disponible, el sistema debe registrar el error y reintentar la conexión.

HU2	Generación de variables calculadas
Requerimiento Relacionado	RF1.3, RNF1.2
Historia de Usuario	Como investigador, quiero que el sistema calcule automáticamente variables como Cianobacterias Total, Nitrógeno Inorgánico Total y Dominancia de Cianobacterias (%), para que estos datos estén listos para ser utilizados en las predicciones.
Criterios de Aceptación	1. El sistema calcula Cianobacterias Total sumando todos los géneros de cianobacterias registradas. 2. Nitrógeno Inorgánico Total se calcula sumando nitrito, nitrato y amonio.

	3. Dominancia de cianobacterias (%) se calcula correctamente y se almacena en la base de datos.

HU3	Generación automática de predicciones mensuales
Requerimiento Relacionado	RF2.1, RF2.2, RNF1.1, RNF1.2
Historia de Usuario	Como investigador, quiero que el sistema prediga la clasificación de células de cianobacterias, concentración de clorofila y dominancia de cianobacterias con intervalo mensualmente, para realizar análisis y planificar acciones.
Criterios de Aceptación	1. Las predicciones se generan cuando un usuario selecciona un sitio para predecir y oprime el botón “Predecir” y la fecha es con un mes de diferencia de los últimos datos. 2. Los resultados de las predicciones son almacenados en la base de datos. 3. Se permite visualizar en la interfaz, las predicciones históricas. 3. Se valida la precisión de los modelos y se registran métricas de desempeño.

HU4	Ejecución automática del modelo
Requerimiento Relacionado	RF3.1, RF3.2, RNF1.1, RNF1.2, RNF2, RNF3
Historia de Usuario	Como investigador, quiero que el modelo de predicción se reentrene automáticamente al recibir nuevos datos o realizar cambios en la

	base de datos, asegurando que siempre trabaje con la información actualizada.
Criterios de Aceptación	1. El sistema utiliza triggers en las bases de datos para detectar cambios. 2. El backend está escuchando constantemente una función de notificación de la base de datos y dispara el reentrenamiento del modelo al recibir una notificación.

HU5	Interfaz de usuario para visualización de predicciones
Requerimiento Relacionado	RF4.1, RF4.2, RNF4, RNF5.1, RNF6
Historia de Usuario	Como investigador, quiero que la interfaz muestre los resultados de la predicción y destaque las clases para facilitar la interpretación de los datos.
Criterios de Aceptación	1. La interfaz permite visualizar predicciones de manera clara. 2. Los valores críticos se destacan en rojo según umbrales preconfigurados, amarillo los valores medios y verdes los que no presentan peligro.

HU6	Reentrenamiento y actualización del modelo
Requerimiento Relacionado	RF6.1, RNF4, RNF1.1, RNF2, RNF6

Historia de Usuario	Como administrador del sistema, quiero que el modelo pueda actualizarse y reentrenarse con nuevos datos, asegurando que siga mejorando su precisión con el tiempo.
Criterios de Aceptación	1. Se permite cargar nuevos datos y con esto disparar el reentrenamiento del modelo. 2. Se registra cada actualización del modelo y sus métricas de desempeño.

HU7	Seguimiento del Desempeño del Modelo
Requerimiento Relacionado	RNF3, RF5.2
Historia de Usuario	Como administrador del sistema, quiero contar con métricas de desempeño del modelo en la interfaz, para evaluar su precisión y mejorar su funcionamiento.
Criterios de Aceptación	1. Se muestra en la interfaz la tasa de error de las predicciones, el mejor modelo elegido. 2. Se muestra un gráfico con los valores de las métricas de los modelos, para cada sitio y variable, para poder monitorear el rendimiento del modelo.

DISEÑO

PANTALLAS PROTOTIPO
Se definieron las siguientes pantallas clave:

Fig 13. Mock up, interfaz de selección de sitio.

1. Pantalla de Selección de Perfil (Sitio de Monitoreo)

Esta vista permite al usuario seleccionar un sitio de monitoreo entre los puntos estratégicos del embalse (p. ej., “C – Centro”, “DCQ – Cosquín”, “TAC1 – Toma”, entre otros). Los botones se presentan junto a un mapa satelital de referencia que indica la ubicación geográfica de los sitios, facilitando la asociación espacial de los datos.

Componentes:

- Lista de sitios con etiquetas y ubicación.
- Imagen satelital con puntos de medición georreferenciados.
- Botón de acción: “Seleccione el perfil”.

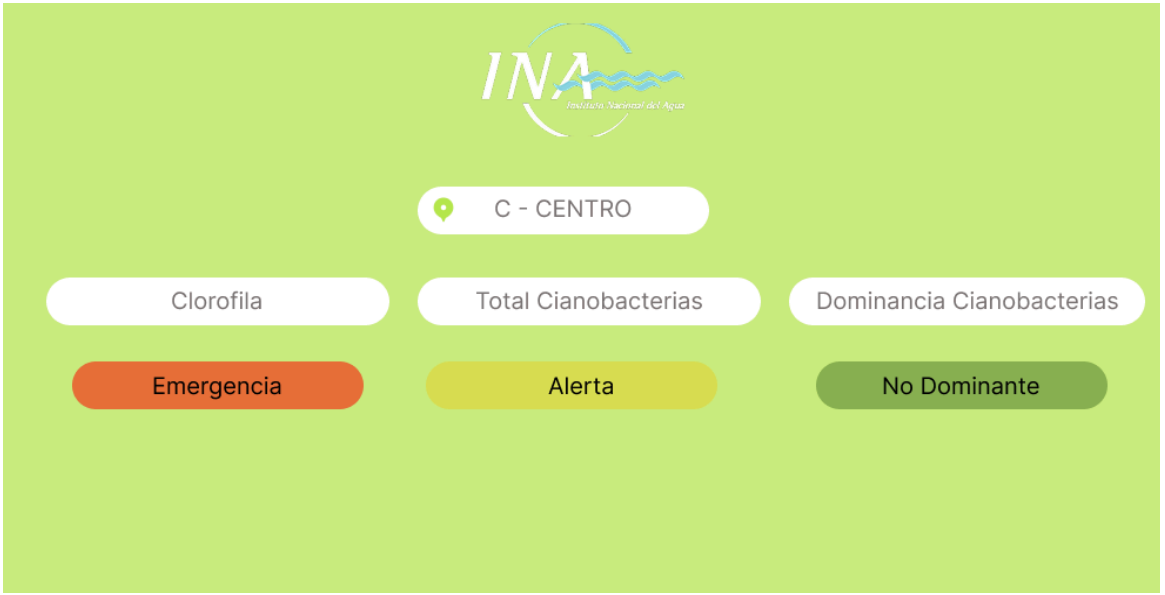


Fig 14. Mock up, interfaz de salida de predicción por sitio.

2. Pantalla de Visualización de Resultados

Una vez seleccionado el sitio, esta pantalla muestra los valores actuales o más recientes de las tres variables objetivo del modelo predictivo: **Concentración de clorofila (µg/L), Cantidad total de cianobacterias (cel/L), Dominancia de cianobacterias (%).**

Además, el sistema clasifica automáticamente el nivel de riesgo asociado a los valores predichos, mostrando un indicador visual (botones codificados por color) con las siguientes categorías: **Emergencia (rojo), Alerta (amarillo), Vigilancia (verde),** para Clorofila y Total de Cianobacterias) y **Dominante (rojo), No dominante (verde)** para Dominancia de Cianobacterias.

Componentes:

- Selector del sitio activo.
- Visualización de variables clave.
- Indicadores de nivel de riesgo basados en umbrales predefinidos.

Fig 15 .Mock up, interfaz de salida de predicción por sitio.

3. Pantalla de Predicciones Históricas

Esta pantalla permite consultar los resultados generados previamente por el modelo predictivo, organizados en una tabla que resume las predicciones históricas por sitio, fecha y variable objetivo. Para cada registro se muestra la variable predicha (Clorofila, Cianobacterias o Dominancia de Cianobacterias), la fecha y la clasificación asignada según los niveles de riesgo definidos.

Esta funcionalidad permite hacer un seguimiento temporal del comportamiento del embalse, identificar patrones estacionales o eventos críticos pasados, y evaluar la efectividad de las decisiones tomadas en base a estas predicciones.

Componentes:

- Tabla histórica con columnas: fecha, sitio, variable, predicción.
- Clasificación basada en criterios técnicos del INA (Vigilancia, Alerta, Emergencia; Dominante/No dominante).
- Acceso cronológico y organizado para análisis.

ARQUITECTURA

El sistema está diseñado en una arquitectura modular que permite integrar y procesar datos de la calidad del agua e hidrometeorología para predecir la proliferación de cianobacterias en el Embalse San Roque. La estructura se basa en tres capas principales:

- Capa de Datos (Bases de Datos y Vistas en PostgreSQL)

- Capa de Procesamiento (Backend en Python + Machine Learning)
- Capa de Presentación (Frontend en React.js)

Esta arquitectura permite la recolección, procesamiento, modelado y visualización de datos en una plataforma eficiente y automatizada.

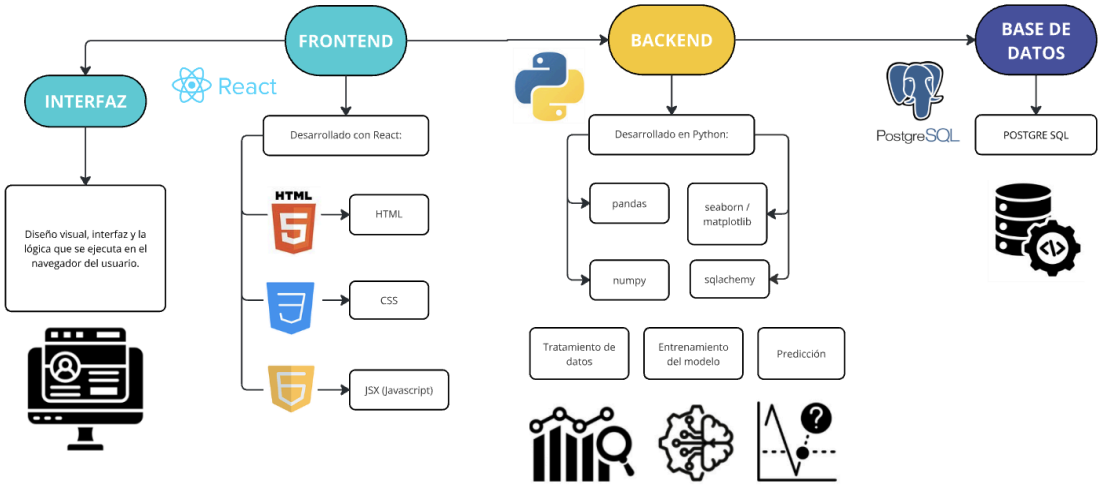


Fig 16. Diagrama de despliegue del modelo. Elaboración propia

Con este diseño, se busca cumplir con cuatro objetivos clave:

- **Modularidad:** cada proceso (ingestión, preprocesado, entrenamiento, inferencia, persistencia histórica y presentación) debe vivir en su propio componente, para facilitar pruebas, mantenimiento y despliegues independientes.
- **Escalabilidad y resiliencia:** los servicios puedan replicarse horizontalmente, y la gestión de reintentos garantice que el sistema soporte fallos parciales sin interrumpir el servicio.
- **Desacoplamiento mediante eventos:** La comunicación entre capas se sustenta en un mecanismo de notificaciones, de modo que quien produce un dato no necesita saber quién lo consumirá, y viceversa. Con esto, cada componente puede escalar o reconfigurar de forma independiente, sin impactar al resto del sistema.
- **Trazabilidad:** todas las ejecuciones de entrenamiento y predicciones deben quedar registradas (métricas de modelos y predicciones históricas), para poder encontrar y analizar cualquier ejecución, así como monitorear rendimientos.

Componentes principales de cada capa.

1. **Bases de datos**

- water_quality: Almacena los registros de calidad del agua.
- alerts: Contiene datos hidrometeorológicos.
- model_data: Almacena el dataframe que usarán de entrada los modelos, las predicciones generadas y las métricas de modelos luego del entrenamiento.

2. **Backend**
   - **Ingestión:** carga con SQLAlchemy los datos de calidad de agua e hidrometeorología.
   - **Tratamiento:** limpia registros, filtra sitios, genera lags y variables derivadas.
   - **Persistencia histórica:** almacena el DataFrame final y registra cada predicción.
   - **Entrenamiento:** script independiente que ajusta pipelines de ML (Machine Learning), compara modelos y almacena el mejor modelo junto a sus métricas.
   - **Inferencia (API):** servicio REST que expone endpoints para predecir (por perfil o “Todos”), consultar el data frame final y obtener el historial de predicciones y gráfico de métricas de los modelos.
   - **Notificar:** un hilo daemon suscrito a una función de notificación, y un worker que reconstruye el data frame, con los nuevos datos y dispara el reentrenamiento.
3. **Frontend**
   - Interfaz web que hace solicitudes http al backend, solicita predicciones, muestra predicciones históricas, dataframe final, gráficos de métricas e información sobre el modelo y las métricas usadas para elegir el mejor modelo, para ayudar a la interpretación de las predicciones.

Patrones y decisiones clave

- **Layered:** separación clara entre capa de datos, lógica de negocio, aplicación y presentación. Facilita el mantenimiento y la evolución independiente de cada módulo. De este modo, podemos actualizar o escalar una capa sin afectar al resto.
- **Event-Driven/Observer:** triggers y LISTEN/NOTIFY desacoplan productor (BD) y consumidor (backend). Esto nos permite añadir nuevos consumidores sin modificar al productor y viceversa, logrando un sistema más flexible a cambios.
- **Pipeline:** el flujo ETL/ML se articula como etapas definidas, una serie de pasos encadenados, cada uno con una responsabilidad concreta y contratos claros de entrada/salida. Garantiza reproducibilidad y facilita la prueba de cada bloque por

separado. Además, permite intercambiar, modificar o reordenar pasos sin reescribir el flujo completo.

- **Repository:** SQLAlchemy abstrae los accesos a datos, aislando la lógica de negocio. Así, la lógica de negocio manipula objetos de alto nivel en lugar de cadenas de SQL.
- **Singleton:** los modelos y escaladores se cargan una vez en memoria para todas las peticiones de predicción. Esto evita la sobrecarga de leer ficheros (archivos en disco), en cada petición y minimiza la latencia. Este patrón garantiza además que todas las peticiones compartan exactamente los mismos artefactos, conservando la consistencia de las predicciones.
- **Producer-Consumer:** listener (producer de eventos de cambio) y worker (consumer y orquestador) coordinados por debounce (agrupar múltiples eventos ocurridos en rápida sucesión y ejecutar la acción asociada sólo una vez, tras un periodo de inactividad), por si hay más de un cambio en la base de datos.

Stack tecnológico

Para cada capa se eligió una tecnología acorde a sus necesidades:

- **PostgreSQL,** proporcionada por las bases del INA, se suma a la creación y manejo de vistas y triggers para transformación inicial y notificaciones.
- **SQLAlchemy** (o psycopg2) como ORM (Object-Relational Mapping) y para LISTEN/NOTIFY.
- **Python 3.9+** en entorno virtual de python (pandas, scikit-learn, TensorFlow/Keras, KerasTuner).
- **Flask o FastAPI** para exponer la API RESTful de inferencia.
- **Docker** y docker-compose para orquestar base de datos, backend y frontend.
- **Threading y select.select()** para implementar listener y worker como hilos daemon.

Flujo de Datos

El flujo de datos sigue un esquema estructurado desde la captación de datos hasta la visualización de predicciones en la interfaz gráfica. Se compone de los siguientes pasos:

Captura de Datos:

- Mediciones manuales de calidad del agua: Se registran en hojas de cálculo y se almacenan en una base de datos PostgreSQL mediante un proceso ETL (Extracción, Transformación y Carga).

- Sensores hidrometeorológicos: Capturan en tiempo real datos de temperatura, precipitación y nivel del embalse. Previo al almacenamiento en la base de datos, estos registros pasan por un proceso ETL para detectar aquellos registros con errores en las mediciones.

Consolidación de Datos mediante Vistas SQL:

- Se unifican los datos de ambas bases de datos, en vistas optimizadas para consultas eficientes, en cada una. Estas filtran y organizan la información relevante, evitando múltiples consultas pesadas en el backend.

Procesamiento de datos:

- Se extraen los datos de las vistas y se construye el dataframe, que será la entrada del modelo.

Entrenamiento de los modelos:

- Se entrenan modelos de Random Forest, Regresión Logística y MLP para predecir la clasificación de alertas de cianobacterias, clorofila y dominancia de cianobacterias.

Almacenamiento y Visualización de Resultados:

- Las métricas de resultado de entrenamiento se almacenan en la base de datos model_data, para seguir el rendimiento de los modelos.
- Desde el frontend se hacen solicitudes http para acceder y hacer solicitudes a cada uno de los endpoints expuestos por el backend.
    - Al hacer una predicción los datos se presentan en la interfaz web en React.js
    - Se permite acceder al dataframe, al historial de predicciones y al monitoreo de las métricas del modelo.
- Las predicciones se guardan en la base model_data.
- Ante una actualización de las tablas en la base de datos de water_quality se dispara una notificación al backend, que se encarga de correr el proceso de actualización del dataframe, reentrenar los modelos y recargarlos sin necesidad de reiniciar ningún servicio.

PROCESAMIENTO Y EXPLORACIÓN DE LOS DATOS

Análisis de datos

Para comprender la estructura y calidad del conjunto de datos utilizado en este proyecto, se realizaron diversas tareas de análisis exploratorio de datos. Estas permitieron identificar características clave de las variables, relaciones entre ellas, y posibles problemas de calidad de datos que podrían afectar el rendimiento de los modelos.

Se analizaron tanto variables numéricas como categóricas. Entre las técnicas utilizadas se destacan:

- Visualización de valores faltantes mediante mapas de calor y gráficos de dispersión, para obtener un panorama general de la completitud de los datos.
- Histogramas y curvas de densidad (KDE) para observar la distribución de las variables numéricas, identificar valores extremos o atípicos, y comprender el rango y comportamiento general de cada variable.
- Gráficos de barras y conteo de valores únicos para variables categóricas, lo cual permitió detectar posibles errores de carga o inconsistencias en la nomenclatura.
- Matrices de correlación para variables numéricas, con el objetivo de identificar patrones de dependencia, redundancias entre columnas y posibles relaciones útiles para tareas de predicción.
- Gráficos de evolución temporal para visualizar tendencias a lo largo del tiempo.
- Análisis estacional mediante gráficos de barras de promedios por estación del año (verano, otoño, invierno, primavera).
- Análisis espacial mediante promedios por código de perfil.
- Análisis de anomalías en series temporales analizando valores máximos y mínimos dentro de la base de precipitaciones.

Se calcularon además medidas estadísticas como la media, mediana, desviación estándar y los percentiles 25%, 50% y 75%, lo cual permitió describir estadísticamente la dispersión y tendencia central de los datos.

Se adjunta documentos enviados al INA-SCIRSA para verificación de la coherencia y definición de tratamientos de los datos en el **Anexo I**.

Detección de tipos de datos

Se verificó el tipo de dato de cada columna del conjunto, distinguiendo entre:

- Columnas numéricas (por ejemplo: "Clorofila (µg/l)", "T° (°C)", "Total de Cianobacterias (Cel/L)", entre otras), sobre las que se aplicaron análisis estadísticos y gráficos.
- Columnas categóricas (por ejemplo: "Condición Térmica", "Descripción Estratificación" y "Código Perfil"), para las cuales se validaron los dominios posibles y se analizaron los valores únicos presentes.

Este proceso fue clave para asegurar que cada variable se interpretara correctamente en el contexto del análisis y del posterior modelado.

Identificación de datos nulos o faltantes

Se contabilizó la cantidad de datos nulos por columna, identificando aquellas variables con mayor proporción de valores faltantes.

Este análisis permitió determinar la magnitud del problema de completitud y fundamentar la necesidad de aplicar estrategias específicas de imputación o descarte.

A continuación, se presenta un mapa de calor que visualiza los valores faltantes en el conjunto de datos, donde las celdas en color amarillo indican la ausencia de datos por variable.

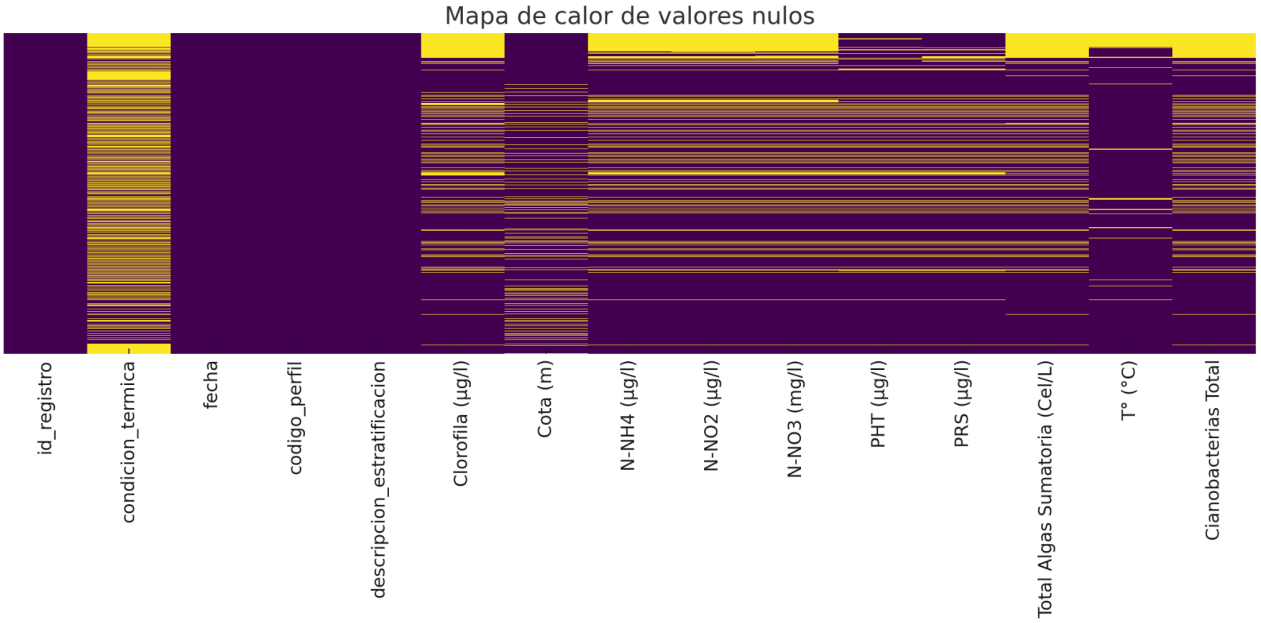


Fig 17. Mapa de calor de valores faltantes, por variable, distribuido en el tiempo (eje y).

Cantidad de valores nulos por variable

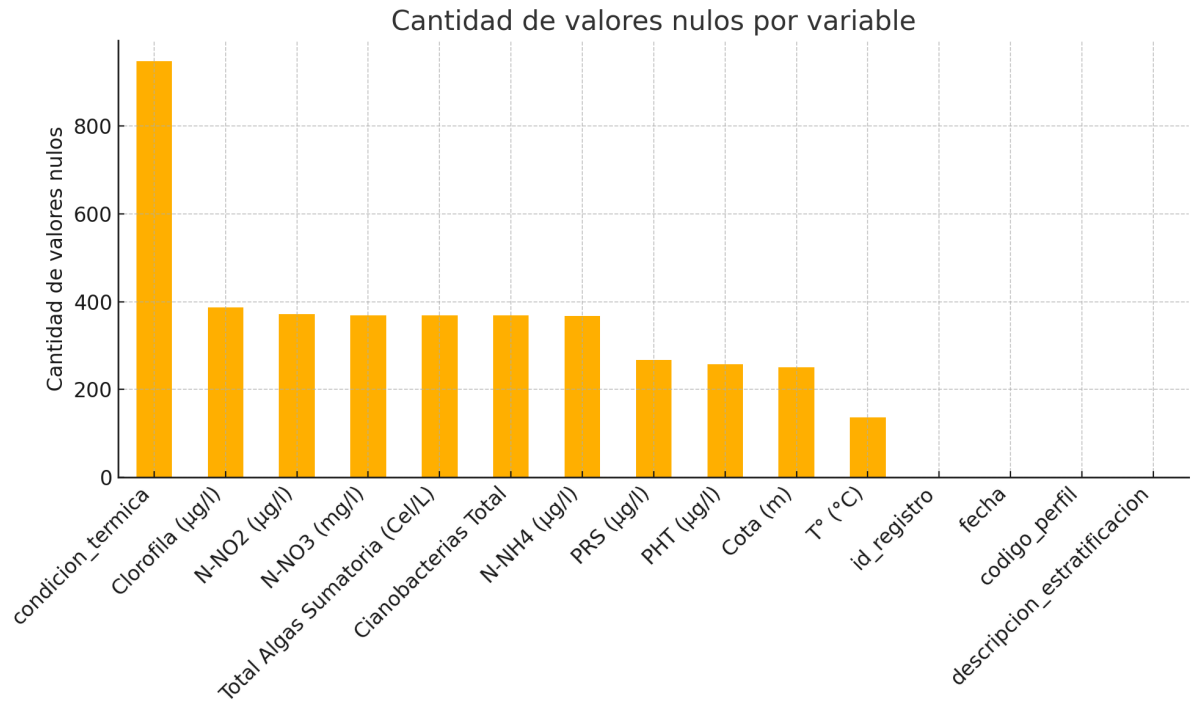


Fig 18. Gráfico de barras de cantidad de valores faltantes por variable.

Además del análisis global, se realizaron evaluaciones más específicas para conocer la distribución de los faltantes por cada variable en el tiempo y el espacio:

- Porcentaje de faltantes por año y sitio de muestreo: Este análisis permitió identificar períodos históricos con vacíos importantes de información en determinadas variables. Por ejemplo, para la variable Clorofila (µg/l) en el perfil C1, los años 1996, 1997 y 1998 presentan un 100% de datos faltantes, mientras que en el año 2000 no se registraron ausencias.

  Se adjunta ejemplo de la variable Clorofila en los años 1996-2001 en el sitio C1, el documento completo se detalla en el Anexo I.

codigo_perfil	año	variable_entrada	% Valores faltantes
C1	1996	Clorofila (µg/l)	100,00
C1	1997	Clorofila (µg/l)	100,00
C1	1998	Clorofila (µg/l)	100,00
C1	1999	Clorofila (µg/l)	66,67
C1	2000	Clorofila (µg/l)	0,00

- Porcentaje de faltantes por estación del año y por perfil: Se analizó también la distribución estacional de los datos faltantes para las variables objetivo, observando diferencias importantes entre sitios. En invierno, perfiles como DLC1 y DLM1 presentaron porcentajes de ausencias superiores al 85%, mientras que otros perfiles, como TAC1 o DCQ1, tuvieron tasas inferiores al 5%.

  Se adjunta tabla con ejemplo de la variable Clorofila en la estación Invierno representativo de estos análisis. Los demás análisis se detallan en el Anexo I.

Estación	Código Perfil	% Faltantes
Invierno	C1	3.33%
Invierno	DCQ1	3.51%
Invierno	DLC1	91.18%
Invierno	DLM1	88.46%
Invierno	DSA1	4.55%
Invierno	TAC1	3.13%
Invierno	TAC4	5.00%

Verificación de la coherencia y consistencia de los datos

Se realizó un análisis orientado a detectar posibles errores de carga o inconsistencias entre columnas relacionadas. Para ello se aplicaron los siguientes procedimientos:

- Análisis de valores únicos por columna, para detectar inconsistencias en los valores de categorías esperadas.
- Cruce de variables relacionadas, como el caso de "Total de Algas Sumatoria" y "Total de Cianobacterias", donde se verificó la coherencia entre el valor total y los desgloses por género.
- Comparación de registros duplicados o similares en cuanto a fecha y sitio de muestreo, validando que los valores asociados fueran consistentes.
- Identificación de valores extremos o fuera de rango mediante el análisis de máximos/mínimos.

Estos controles ayudaron a validar la calidad de la base de datos y a sentar las bases para la definición de estrategias de tratamientos de datos previa al modelado.

Principales conclusiones del análisis en la base de datos de Calidad de Agua

- Las variables **Clorofila, Cianobacterias y Total de Algas** presentan distribuciones altamente asimétricas, con valores extremos que elevan la media; se utiliza la mediana y percentiles para representar su tendencia central.

- Se observó una alta correlación positiva entre: Clorofila y Cianobacterias; Clorofila y PHT (Fósforo total); Cianobacterias y Total de Algas.

- La temperatura del agua, del aire y las precipitaciones mostraron un patrón estacional muy claro, con máximos en verano y mínimos en invierno.

- Los nutrientes (NH_4^+, NO_2^-, NO_3^-, PHT, PRS) presentan alta variabilidad, con picos abruptos en algunos años y estaciones, y valores extremos que indican posibles eventos puntuales de aporte externo.

- Cota (nivel del lago) fue una variable estable, sin variaciones significativas a lo largo del tiempo.

- Análisis estacional: primavera y verano muestran mayores concentraciones de clorofila y cianobacterias; el invierno presenta valores mínimos.

- Análisis temporal anual: se detecta una posible tendencia creciente en los niveles de clorofila y cianobacterias en los últimos años del registro.

- Análisis espacial: existen perfiles con valores más altos, lo que podría indicar zonas de mayor vulnerabilidad.

Principales hallazgos del análisis en la base de datos de Calidad de Agua

Datos atípicos

Como parte del proceso de limpieza de datos, se identificaron y corrigieron valores atípicos que fueron producto de errores de carga, duplicación o inconsistencias lógicas. Estas correcciones se realizaron siguiendo criterios objetivos basados en el conocimiento del dominio, así como una validación cruzada con los registros históricos disponibles en las planillas oficiales del INA-SCIRSA.

Se establecieron dos premisas clave para la **detección y corrección de anomalías**, junto con una tercera **decisión metodológica** orientada a garantizar la consistencia temporal del conjunto de datos:

Premisa 1: Único valor de cota por fecha

La variable Cota (m) representa una medida general del estado del lago y, por convención, debe tener un único valor por día. Se identificaron casos donde, para una misma fecha, existían registros con diferentes valores de cota. Estos fueron considerados errores de carga y se corrigieron de la siguiente manera:

Caso 25/04/2023: Se detectaron dos valores diferentes de cota para la misma fecha. El valor correcto (35.36 m) se validó con la base de datos del INA y se aplicó a todos los registros de esa fecha.

2023-04-25	1726	2023-04-25	35.37	C1
	1727	2023-04-25	35.36	TAC1
	1728	2023-04-25	35.36	TAC4
	1729	2023-04-25	35.36	DCQ1
	1730	2023-04-25	35.36	DSA1

Caso 22/12/2003: De seis registros con la misma fecha, cinco coincidían en el valor de cota y uno difería. Se adoptó el valor mayoritario como referencia, considerando el caso como un error humano aislado.

2003-12-22	584	2003-12-22	34.73	C1
	585	2003-12-22	31.90	DCQ1
	586	2003-12-22	31.90	DLM1
	587	2003-12-22	31.90	DSA1
	588	2003-12-22	31.90	TAC1
	589	2003-12-22	31.90	TAC4

Premisa 2: No debe haber más de una medición por sitio en la misma fecha

Cada registro del conjunto de datos está asociado a una fecha y a un código de perfil, que representa un sitio específico de muestreo. Según la metodología del monitoreo, no debería existir más de una medición por sitio en una misma fecha. Sin embargo, durante el análisis se detectaron múltiples casos en los que esta regla no se cumplía.

La mayoría de estas duplicaciones se debieron a errores de carga: los valores correspondían en realidad a diferentes perfiles, pero fueron asignados erróneamente al mismo código. También se identificaron casos en los que registros pertenecientes a perfiles que no forman parte del análisis fueron cargados bajo códigos de perfiles incluidos en esta tesis. En estos casos, se descartaron los registros mal asignados para preservar la coherencia del conjunto de datos con el diseño del estudio.

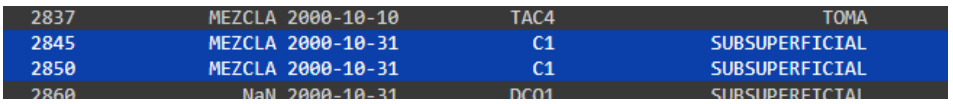
Este tipo de errores generaba inconsistencias lógicas y podía distorsionar las relaciones entre variables. Por ello, se aplicaron criterios estrictos de verificación cruzada con las

planillas originales y, según correspondiera, se procedió a la corrección o eliminación de los registros.

Se adjuntan dos ejemplos:

Ejemplo 1 – Perfil C1, fecha 2000-10-31:

Se encontraron dos registros con código de perfil C1 para esa fecha. Tras revisar el historial de mediciones en los registros del INA, se comprobó que uno de los registros había sido mal rotulado y correspondía al perfil Ce (no utilizado en este proyecto). Se corrigió el código de perfil para reflejar el valor correcto.



2837	MEZCLA 2000-10-10	TAC4	TOMA
2845	MEZCLA 2000-10-31	C1	SUBSUPERFICIAL
2850	MEZCLA 2000-10-31	C1	SUBSUPERFICIAL
2860	NaN 2000-10-31	DC01	SUBSUPERFICIAL

Ejemplo 2 – Perfil DSA1, fecha 2017-03-28:

En esa fecha se realizaron, de manera excepcional, cuatro mediciones en el sitio DSA: una medición principal identificada como DSAa y tres adicionales (DSAb, DSAc y DSAd). Dado que la medición DSAa corresponde al registro principal y representa la información prioritaria del sitio, se optó por conservar únicamente esa y descartar las demás.

12685	NaN	2017-03-28	DSA1	SUBSUPERFICIAL
12689	NaN	2017-03-28	DSA1	SUBSUPERFICIAL
12690	NaN	2017-03-28	DSA1	SUBSUPERFICIAL
12691	NaN	2017-03-28	DSA1	SUBSUPERFICIAL

Premisa 3: Solo debe existir una medición mensual por sitio

Históricamente, en algunos períodos se realizaron más de una medición por sitio en el mismo mes. Sin embargo, en años recientes la metodología se estandarizó a una única medición mensual por sitio. Para mantener la coherencia temporal de los datos y evitar sesgos por sobremuestreo en ciertas épocas, se adoptó la premisa de conservar sólo una medición por mes y por sitio. En los casos en que existían múltiples registros mensuales para un mismo perfil, se aplicó un criterio de selección para conservar el más representativo.

Estrategias de datos faltantes y tratamientos de datos:

1. Conversión de variable fecha:

   La variable fecha es fundamental para todo nuestro modelo, se necesita que sea precisa para:

   - El análisis mensual
   - La agrupación por período
   - La selección de la mejor medición del mes

   Es decir, el comportamiento temporal de las floraciones de cianobacterias dependen directamente de una fecha válida y en el formato correcto.

   Por lo que asegurarnos que todos los registros tengan una fecha válida y consistente, permite organizar, agrupar o seleccionar datos mensuales de forma confiable.

   Resultado

   No fue necesario eliminar registros debido a errores en el formato de la fecha. Todos los valores presentes en la columna correspondiente pudieron ser convertidos correctamente al tipo datetime, lo cual garantiza que las fechas están correctamente formateadas y listas para ser utilizadas en procesos de ordenamiento, agrupamiento y filtrado temporal. Esto permitió conservar la totalidad de los registros y asegurar la calidad del dataset sin pérdida de información.

   ```
   Se eliminaron 0 registros por tener fechas inválidas.
   ```

2. Filtrado de sitios:

   Durante el proceso de análisis para desarrollar estrategias para subsanar los datos faltantes, el equipo del INA, solicitó el análisis de los datos faltantes por sitio, para verificar que la cantidad de datos en cada uno fuera suficiente para el modelo.

   ```
                  Promedio % Faltantes
   codigo_perfil
   DLC1                      65.754190
   DLM1                      62.230216
   TAC4                      11.074380
   DSA1                       8.645161
   C1                         8.557377
   DCQ1                       8.000000
   TAC1                       2.624113
   ```

En este proceso se detectó, como se ve en la imagen, que en los sitios DLC1 (Desembocadura Los Chorrillos) y DLM1 (Desembocadura Las Mojarras) se encontraba un faltante de más del 60% por lo que se decidió quitarlos del data frame.

Resultado

Inicialmente, se contaba con 1.747 registros y 15 columnas, lo que representaba un total de 26.405 datos posibles. De ellos, 3.093 eran valores faltantes, lo que equivale a un **11,8 % de datos faltantes.**

Luego de aplicar el filtro para excluir los sitios DLC y DLM, el conjunto de datos se redujo a 1.429 registros, manteniendo las 15 variables, lo que da un total esperado de 21.435 datos, y se redujo a 1.783 valores nulos (reduciendo en un 57,6% los datos faltantes), lo que representa un **8,32 %.** de datos faltantes total.

3. Filtrado de registros por fechas:

Como se mencionó anteriormente, desde el primer registro en el 18-12-1996 y 24-07-1999 el porcentaje de datos faltantes es significativo, sobre todo en una de las variables esenciales del modelo, Clorofila, que posee un 100% de datos faltantes en el periodo mencionado. Y por no tener registros previos a estos, no se puede lograr una imputación en esas fechas.

Por lo que, en conjunto con el INA se decidió eliminar los registros comprendidos en el período mencionado. Esto mejora la calidad del dataset, reduciendo el ruido de los datos faltantes.

Resultado

En nuestro conjunto de datos contamos con 1.429 registros (filas) y 15 variables (columnas), lo que equivale a un total esperado de 21.435 valores (1.429 × 15).

Antes de aplicar la estrategia de imputación, se detectaron 1783 valores nulos, lo que representa un 8,32 % de datos faltantes sobre el total.

Luego del tratamiento aplicado, el total esperado de valores es 20.130 y el total de datos faltantes se redujo a 1.117 valores nulos, es decir, un 5.55% de datos faltantes.

Esto refleja una mejora significativa en la completitud del dataset, disminuyendo la proporción de datos faltantes en casi 3 puntos porcentuales, lo cual es clave para mejorar la calidad del entrenamiento del modelo.

condicion_termica	631
fecha	0
codigo_perfil	0
descripcion_estratificacion	0
Clorofila (µg/l)	125
Cota (m)	249
N-NH4 (µg/l)	109
N-NO2 (µg/l)	112
N-NO3 (mg/l)	111
PHT (µg/l)	43
PRS (µg/l)	46
Total Algas Sumatoria (Cel/L)	106
T° (°C)	96
Cianobacterias Total	106

Antes del tratamiento

condicion_termica	564
fecha	0
codigo_perfil	0
descripcion_estratificacion	0
Clorofila (µg/l)	38
Cota (m)	249
N-NH4 (µg/l)	40
N-NO2 (µg/l)	41
N-NO3 (mg/l)	40
PHT (µg/l)	29
PRS (µg/l)	32
Total Algas Sumatoria (Cel/L)	19
T° (°C)	46
Cianobacterias Total	19

Después del tratamiento

4. Imputación de valores en la columna `Cota (m)`

La variable Cota (m), que representa el nivel del lago, tiene el mismo valor para todos los sitios del embalse en una misma fecha. Por eso, cuando se detectan valores faltantes en esta columna, en uno o varios sitios para un día determinado, pero el dato sí está presente en otros sitios ese mismo día, se decidió imputar ese valor faltante utilizando el valor existente del mismo día. Esta estrategia permite no sólo completar los datos ausentes, sino también mantener la coherencia temporal en todos los registros correspondientes a esa fecha.

Resultado: Pasamos de 249 valores faltantes para Cota (m) a 0, logrando un porcentaje de datos faltantes para esta variable del **0%**. Y para el total del dataset se redujo el porcentaje de 5.5% a **4.31%** ya que los valores faltantes pasaron de 1117 a 868, sobre un total de 20.130 datos.

Cantidad datos nulos pre tratamiento de variable Cota (m)	
id_registro	0
condicion_termica	564
fecha	0
codigo_perfil	0
descripcion_estratificacion	0
Clorofila (µg/l)	38
Cota (m)	249
N-NH4 (µg/l)	40

```
Cantidad datos nulos post tratamiento de variable Cota (m)
id_registro                          0
condicion_termica                  564
fecha                                0
codigo_perfil                        0
descripcion_estratificacion          0
Clorofila (µg/l)                    38
Cota (m)                             0
N-NH4 (µg/l)                        40
```

5. Selección de fecha representativa por mes

   Dado que solo debe existir una medición mensual por sitio, se identificaron casos en los que, dentro de un mismo mes, había más de una medición para un mismo sitio. Para resolver esta duplicación, se estableció un criterio de selección: para cada mes y cada sitio, se eligió la medición correspondiente al sitio que presentaba la menor proporción de datos faltantes y, en última instancia, la fecha más cercana al final del mes, ya que en ese periodo suelen realizarse los monitoreos. Esta estrategia garantiza una mayor representatividad y calidad del conjunto de datos mensual utilizado por el modelo.

   Resultado:

   Gracias a la estrategia aplicada, se redujeron los valores nulos de 868 a 720. El porcentaje de datos faltantes bajó de 4.31% a **3.86%.** Teniendo en cuenta como mencionamos antes que el total de datos es 18.630.

id_registro	0
condicion_termica	564
fecha	0
codigo_perfil	0
descripcion_estratificacion	0
Clorofila (µg/l)	38
Cota (m)	0
N-NH4 (µg/l)	40
N-NO2 (µg/l)	41
N-NO3 (mg/l)	40
PHT (µg/l)	29
PRS (µg/l)	32
Total Algas Sumatoria (Cel/L)	19
T° (°C)	46
Cianobacterias Total	19

Antes del tratamiento

id_registro	0
condicion_termica	519
fecha	0
codigo_perfil	0
descripcion_estratificacion	0
Clorofila (µg/l)	29
Cota (m)	0
N-NH4 (µg/l)	18
N-NO2 (µg/l)	18
N-NO3 (mg/l)	18
PHT (µg/l)	17
PRS (µg/l)	17
Total Algas Sumatoria (Cel/L)	11
T° (°C)	41
Cianobacterias Total	11

Después del tratamiento

6. Imputación de la variable condición térmica

   La variable *condición térmica* representa el estado de **mezcla o estratificación** térmica en la columna de agua y se calcula a partir de mediciones de temperatura y

profundidad. Al tratarse de una variable definida a nivel de sitio y fecha, todos los registros correspondientes a un mismo sitio y día comparten el mismo valor.

Para mitigar los datos faltantes, se definió una estrategia de imputación basada en la propuesta metodológica de Andreoni (2020), la cual consiste en dos pasos principales:

1. Propagación horizontal del valor existente: en primer lugar, si al menos uno de los registros del sitio y misma fecha ya posee un valor asignado de *condición térmica*, dicho valor se replica a los demás registros del mismo conjunto.

2. Cálculo a partir de gradientes térmicos verticales: cuando no es posible propagar el valor, se calcula una nueva variable de *condición térmica* a partir de las diferencias de temperatura (ΔT) y profundidad (ΔZ) entre puntos consecutivos, siguiendo el siguiente criterio jerárquico (respetando el orden de evaluación):

    - Si la diferencia de temperatura (redondeada a un decimal) entre dos puntos con diferencia de profundidad igual o menor a 1 metro es **mayor o igual a 1 °C**, se asigna el valor **ESTRATIFICADA**, ya que se considera un cambio brusco de temperatura en poca profundidad.

    - Si la diferencia de temperatura es exactamente **0,9 °C** con $\Delta Z \leq 1$ m, se considera un caso **ambiguo**. El valor se deja como **INDETERMINACIÓN** por requerir revisión, etiquetado como caso “T” (incumplimiento de la condición térmica).

    - Si $\Delta T \geq 1$ °C pero la diferencia de profundidad supera 1 metro, también se considera **indeterminado** y se deja como **INDETERMINACIÓN**, identificado como “Z” (incumplimiento de la condición de profundidad).

    - En todos los demás casos, donde no se detecta un gradiente térmico significativo, se asigna el valor **MEZCLA**, ya que no hay evidencia de estratificación térmica.

Esta lógica se ilustra en la tabla resumen de decisiones incluida en la siguiente figura, donde se detalla la combinación de condiciones evaluadas y su correspondiente clasificación.

Tabla resumen de decisiones

Profundidad (z)	Diferencia de temperatura (Δt)	Resultado	Justificación
≤ 1	≥ 1.0	ESTRATIFICADA	Cambio brusco de T° en poca profundidad
≤ 1	$= 0.9$	INDETERMINACIÓN	Cambio cercano al umbral, se revisa
> 1	≥ 1.0	INDETERMINACIÓN	Cambio ocurre a mucha profundidad
Otro caso	–	MEZCLA	No hay evidencia de estratificación

Dado que en nuestro análisis principal solo se emplearon datos del nivel superficial, fue necesario generar una vista especial del conjunto de datos que incluyera perfiles verticales completos por sitio y fecha, permitiendo así calcular las diferencias térmicas entre distintas profundidades. Este procesamiento se realizó de manera independiente, utilizando un subconjunto más amplio de registros. Posteriormente, los valores de condición térmica obtenidos se integraron al conjunto final mediante el campo *id_registro*, completando únicamente los casos que originalmente presentaban valores faltantes.

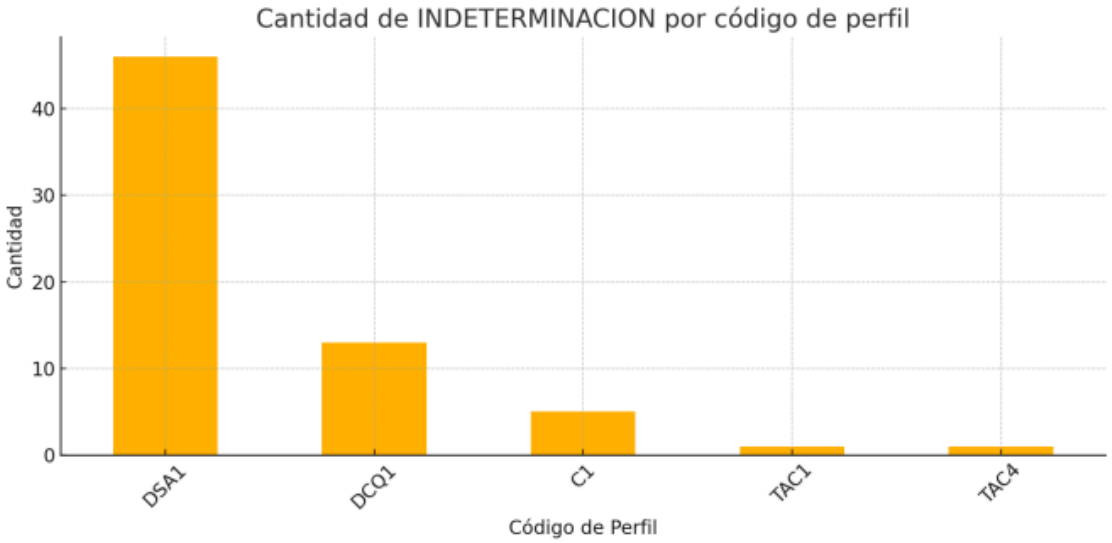
Luego de aplicar el procedimiento de imputación explicado anteriormente, se completaron 188 registros con valores de condición térmica, clasificados como MEZCLA (113), INDETERMINACIÓN (66) o ESTRATIFICADA (9), según los criterios definidos a partir de los gradientes de temperatura y profundidad. Sin embargo, permanecieron 331 registros sin dato, ya que no se disponía de la información mínima necesaria para aplicar ninguna de las reglas de clasificación (por ejemplo, ausencia de mediciones en más de un nivel o falta de temperatura). Estas situaciones fueron identificadas expresamente como “SD” (sin datos) en el conjunto final.

Como este número representa una cantidad considerable, se hizo un análisis adicional para entender mejor de dónde provenían estas indeterminaciones y faltantes. En particular, se revisaron los sitios de medición para ver si estaban concentradas en algunos puntos específicos.

El análisis mostró que los sitios ubicados en las desembocaduras de los ríos San Antonio (DSA1) y Cosquín (DCQ1) son los que más aportan a estas indeterminaciones. Esto probablemente se deba a que en esos lugares las

mediciones suelen hacerse sólo en superficie o en el fondo, sin muchos datos intermedios, lo que dificulta evaluar los cambios verticales de temperatura.

Por esta razón, se decidió no incluir la variable condición térmica como entrada del modelo para los sitios DSA1 y DCQ1. Esta decisión se tomó para asegurar que las variables utilizadas en el modelo aporten información confiable y consistente, evitando sesgos o ruido derivados de datos ambiguos.



Por otro lado, se revisaron manualmente algunos registros que inicialmente figuraban como “INDETERMINACIÓN” y que, tras una evaluación más cuidadosa realizada por las especialistas del INA-CIRSA, pudieron ser reclasificados como “MEZCLA” o “ESTRATIFICADA”. Este ajuste se aplicó a siete casos, en los cuales, si bien la información disponible no cumplía plenamente con los criterios automáticos de clasificación, ofrecía

indicios suficientes para una interpretación fundamentada. Estas correcciones se realizaron exclusivamente en los sitios donde la variable condición térmica se considera como entrada del modelo.

7. Imputación de datos faltantes (estrategia por variable):

Para las variables para las cuales no hay una estrategia o un patrón claro para seguir, se decidió imputar valores faltantes, usando diferentes métodos y elegir el mejor para cada variable con base en métricas de error.

Tratamiento de Temperatura del agua (T°)

La temperatura es un parámetro ambiental clave en el desarrollo de cianobacterias y otros grupos de algas, y muestra una fuerte estacionalidad y variabilidad por sitio de monitoreo. Por este motivo, se optó por una imputación conservadora que respeta estas características.

Se imputaron los valores faltantes de Temperatura (T°) calculando la media por combinación de estación del año y sitio (codigo_perfil).

Tratamiento de Temperatura del aire

La temperatura del aire, al igual que la del agua muestra una fuerte estacionalidad y variabilidad por sitio de monitoreo. Por este motivo, se optó por imputar los valores faltantes calculando la media por combinación de estación del año, mes y sitio (codigo_perfil).

Total Algas Sumatoria (Cel/L) y Cianobacterias Total

Se decidió eliminar los 11 registros que presentaban simultáneamente valores faltantes en las variables Total Algas Sumatoria (Cel/L) y Cianobacterias Total, ambas utilizadas como variables objetivo en los modelos de predicción de floraciones algales.

Esta decisión se fundamenta en dos criterios principales:

- Volumen de datos disponible: El conjunto de datos contiene registros mensuales durante un período de aproximadamente 20 años, lo que garantiza una amplia cobertura temporal y una representación adecuada de la variabilidad estacional y espacial. La eliminación de 11 observaciones no compromete la representatividad del conjunto de datos.
- Calidad del modelo: Dado que estas variables son objetivo del modelo, imputarlas podría introducir ruido o sesgos no deseados. Al eliminarlas, se prioriza la integridad del proceso de entrenamiento y evaluación de los modelos predictivos.

Se considera que esta eliminación mejora la robustez de los análisis posteriores sin afectar la solidez del dataset.

Tratamiento de la variable Clorofila (µg/l)

Se detectaron 20 registros con valores faltantes en la columna Clorofila (µg/l), sobre un total de 1231 registros, correspondiente a un 1,62%. Estos registros fueron imputados aplicando una estrategia híbrida que prioriza la precisión y consistencia de los valores imputados. Se utilizó una estrategia de imputación combinada:

Segmentación por grupo ambiental, los registros fueron agrupados por:

- sitio de monitoreo
- estación del año

En adelante, se denomina "grupo" a cada combinación única de sitio y estación.

Definición de variables predictoras: Para cada grupo, las variables predictoras seleccionadas dependen de la disponibilidad de datos:

Si en el grupo no existían valores faltantes en PHT (µg/l) y PRS (µg/l), se seleccionaron como variables predictoras:

- Total Algas Sumatoria (Cel/L)
- Cianobacterias Total
- PHT (µg/l)
- PRS (µg/l)

Si existían faltantes en PHT o PRS:

- Total Algas Sumatoria (Cel/L)
- Cianobacterias Total
- T° (°C)

El objetivo de esta selección fue maximizar la calidad de las predicciones utilizando la mejor información disponible en cada caso.

Modelo predictivo

Se entrenó un modelo de Random Forest Regressor, elegido por su capacidad de capturar relaciones no lineales entre las variables y su robustez ante datos ambientales ruidosos.

Se entrenó el modelo utilizando únicamente registros completos, es decir, aquellos sin faltantes en Clorofila ni en las variables predictoras seleccionadas.

Mediana local

Como método alternativo de imputación, se calculó la mediana de Clorofila (µg/l) dentro de cada grupo. Esta mediana se usó como imputación fija para todos los registros faltantes del grupo.

Criterio de selección: Para cada registro faltante se seleccionó el valor imputado que generaba menor error:

Si el modelo tenía mejor desempeño → se utilizó su predicción

Si la mediana local resultaba más precisa → se usó la mediana

Tratamiento del error de imputación

Para evaluar la precisión de las imputaciones realizadas en la variable Clorofila, se calcularon dos métricas de error para grupo y estación del año:

1. Error del modelo predictivo

En primer lugar, se entrenó un modelo de Random Forest Regressor utilizando los registros que presentaban datos completos, es decir, sin valores faltantes ni en la variable objetivo (Clorofila) ni en las variables seleccionadas como predictoras (Total Algas Sumatoria, Cianobacterias Total, y según disponibilidad, PHT y PRS o Temperatura).

Se implementó una estrategia de validación cruzada de 3 pliegues (3-fold cross-validation), donde el conjunto de entrenamiento se dividió en tres partes, y el modelo fue evaluado secuencialmente en cada una de ellas usando como métrica el Error Absoluto Medio (MAE).

Este procedimiento permitió estimar la capacidad de generalización del modelo, es decir, su desempeño esperado sobre registros nuevos que no fueron utilizados durante el entrenamiento.

El error entre la predicción y el valor real se cuantificó mediante el cálculo del Error Absoluto Medio (MAE), utilizando la siguiente fórmula:

$$MAE = \frac{\sum_{i=1}^{n} |y_i - x_i|}{n}$$

y es el valor real de Clorofila, x es la predicción del modelo

2. Error de la imputación por mediana

De forma alternativa, se estimó el error que se cometería si se imputara siempre la mediana de Clorofila correspondiente al grupo.

La simulación se realizó aplicando el mismo principio: para cada registro completo del grupo, se asignó como predicción el valor constante de la mediana de Clorofila calculada dentro de ese grupo. Luego, se comparó este valor constante con el valor real de Clorofila registrado para cada observación.

El error se calculó, nuevamente, utilizando la métrica del Error Absoluto Medio (MAE), en este caso entre el valor real y la mediana local:

$$MAE = \frac{\sum_{i=1}^{n} |y_i - x_i|}{n}$$

y es el valor real de Clorofila, x es el valor constante de la mediana

En ambos casos (modelo y mediana), el cálculo del MAE se realizó utilizando los mismos registros completos del grupo, asegurando así una comparación justa y consistente.

El cálculo de ambos errores se realizó a nivel de grupo (combinación de sitio y estación), y no a nivel individual por registro faltante. Por lo tanto, todos los registros imputados dentro de un mismo grupo comparten el mismo error del modelo y error de la mediana.

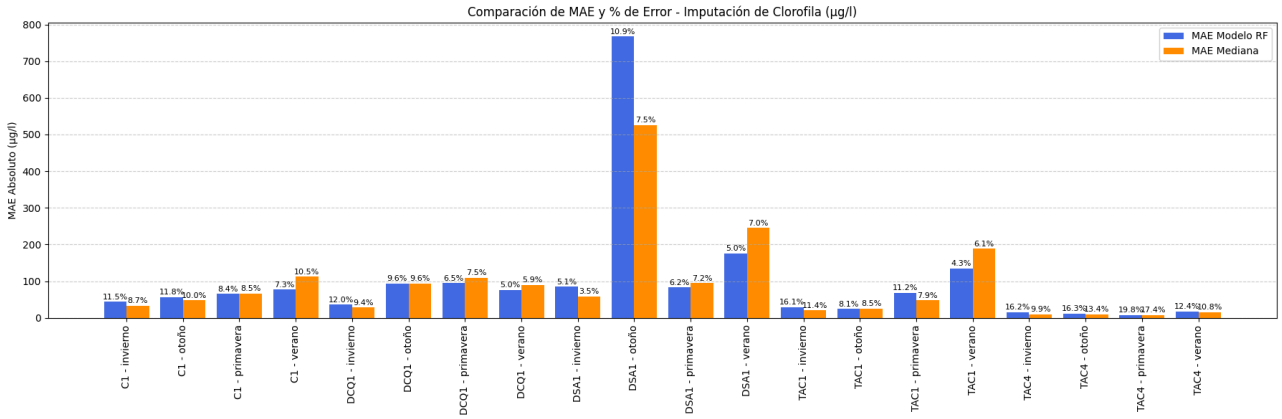
El método de imputación seleccionado para cada registro fue aquel que presentó el menor error de los dos comparados.

Además, se calculó el porcentaje de error relativo respecto al rango de valores reales del grupo. Este indicador permitió analizar si un error era realmente significativo en función de la variabilidad del grupo.

$$\text{Error relativo (\%)} = \frac{MAE}{\text{valor máximo} - \text{valor mínimo}} \times 100$$

Análisis del error de imputación en términos relativos

En las siguientes figuras se presentan los gráficos de barras que comparan el MAE del modelo y de la mediana para cada combinación de sitio de monitoreo y estación del año. Sobre cada barra se indica el porcentaje de error relativo, calculado como el **MAE dividido por el rango real de valores observados en ese grupo**. Este indicador permite visualizar en qué medida el error medio representa una porción significativa del rango del grupo, facilitando así una evaluación más contextualizada de la precisión de cada método de imputación.



Tratamiento de la variable PHT (µg/l)

Se detectaron 11 registros con valores faltantes en la columna PHT (µg/l), sobre un total de 1231 registros, correspondiente a un 0,89%. Se utilizó una estrategia de imputación combinada:

Segmentación por grupo ambiental: Los registros fueron agrupados por combinación de:

- sitio de monitoreo
- estación del año

Definición de variables predictoras: Para cada grupo, se consideraron como variables predictoras:

- Clorofila (µg/l)
- Total Algas Sumatoria (Cel/L)

- Cianobacterias Total
- Temperatura (°C)

La variable PRS (Fósforo reactivo soluble) no fue incluida como predictor, ya que se comprobó en los registros donde faltaba PHT también faltaba PRS, imposibilitando su uso como insumo confiable.

Modelo predictivo

Se entrenó un modelo de Random Forest Regressor, debido a su capacidad de capturar relaciones no lineales y su robustez frente a la variabilidad natural de los datos ambientales.

El modelo se ajustó utilizando exclusivamente registros completos del grupo (sin faltantes en PHT ni en las variables predictoras).

Se aplicó validación cruzada interna (3 particiones) para estimar el error medio absoluto (MAE) del modelo dentro de cada grupo.

Mediana local

Como alternativa al modelo, se calculó la mediana de PHT (µg/l) dentro de cada grupo. Esta medida fue utilizada como imputación constante para todos los valores faltantes del grupo cuando el modelo no resultaba más preciso.

Tratamiento del error de imputación

El análisis del error se realizó a nivel de grupo, no individual, para garantizar coherencia metodológica. En cada grupo con al menos 5 registros completos:

- Se entrenó un modelo Random Forest sobre esos registros.
- Se compararon las predicciones internas del modelo contra los valores reales de PHT.
- Paralelamente, se imputó la mediana del grupo y se comparó con los mismos valores reales.
- En ambos casos, el error se calculó usando la métrica de Error Absoluto Medio (MAE).

Además, se calculó el porcentaje de error relativo respecto al rango de valores reales del grupo. Este indicador permitió analizar si un error era realmente significativo en función de la variabilidad del grupo.

Este procedimiento permitió cuantificar objetivamente la calidad de ambas estrategias y aplicar la más adecuada por grupo.

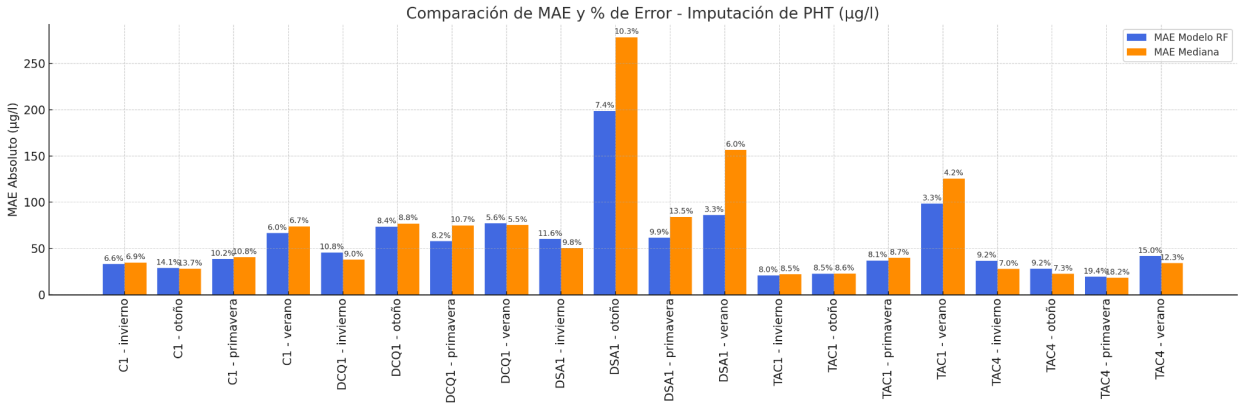
Criterio de selección

El valor imputado en cada grupo se eligió según el menor error:

- Si el modelo tenía menor MAE, se usó su predicción.
- Si la mediana era más precisa, se imputó con ese valor constante.

Resultados

En las siguientes figuras se presentan los gráficos de barras que comparan el MAE del modelo y de la mediana para cada combinación de sitio de monitoreo y estación del año.
Sobre cada barra se indica el porcentaje de error relativo, calculado como el **MAE dividido por el rango real de valores observados en ese grupo**. Este indicador permite visualizar en qué medida el error medio representa una porción significativa del rango del grupo, facilitando así una evaluación más contextualizada de la precisión de cada método de imputación.



Tratamiento de la variable PRS (µg/l)

Se detectaron 11 registros con valores faltantes en la columna PRS (µg/l), sobre un total de 1231 registros, correspondiente a un 0,89%. Se utilizó una estrategia de imputación combinada:

Segmentación por grupo ambiental: Los registros fueron agrupados por combinación de:

- sitio de monitoreo
- estación del año

Definición de variables predictoras:

Para cada grupo, se utilizaron como variables predictoras aquellas que mostraron relaciones limnológicas significativas con PRS y estaban disponibles en los registros completos:

- PHT (µg/l)
- Clorofila (µg/l)
- Total Algas Sumatoria (Cel/L)
- Cianobacterias Total
- Temperatura (°C)

Modelo predictivo

Se entrenó un modelo de Random Forest Regressor, elegido por su capacidad de capturar relaciones no lineales entre variables ambientales.

El modelo se entrenó exclusivamente sobre registros completos dentro de cada grupo (sin valores faltantes en PRS ni en las variables predictoras).

Para evaluar la precisión del modelo, se aplicó validación cruzada interna de 3 particiones (KFold) y se calculó el error medio absoluto (MAE) entre las predicciones y los valores reales.

Mediana local

Como estrategia alternativa, se calculó la mediana de PRS dentro de cada grupo. Este valor fue utilizado como imputación constante en aquellos casos donde el modelo no superó en precisión a la mediana.

Tratamiento del error de imputación

En cada grupo:

- Se entrenó el modelo Random Forest y se validó mediante particiones internas.
- Se calculó el MAE del modelo y el MAE asociado a la imputación por mediana.

La comparación de estos errores permitió seleccionar objetivamente la mejor estrategia.

Además, se calculó el porcentaje de error relativo respecto al rango de valores reales del grupo. Este indicador permitió analizar si un error era realmente significativo en función de la variabilidad del grupo.

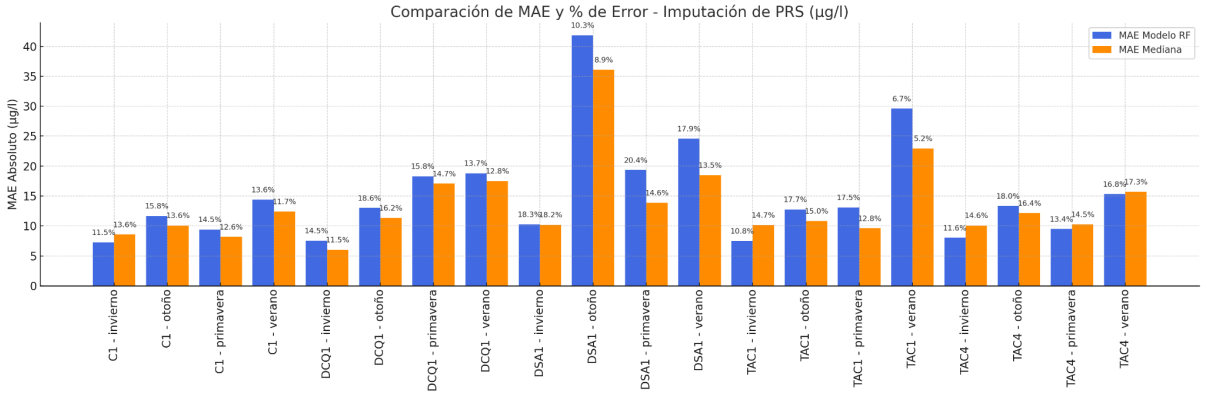
Criterio de selección

La estrategia de imputación se eligió con base en la menor tasa de error por grupo:

- Si el modelo presentó menor MAE que la mediana, se imputaron los valores faltantes con la predicción del modelo.
- Si la mediana fue más precisa, se utilizó como imputación fija en ese grupo.

Resultados

En las siguientes figuras se presentan los gráficos de barras que comparan el MAE del modelo y de la mediana para cada combinación de sitio de monitoreo y estación del año. Sobre cada barra se indica el porcentaje de error relativo, calculado como el **MAE dividido por el rango real de valores observados en ese grupo**. Este indicador permite visualizar en qué medida el error medio representa una porción significativa del rango del grupo, facilitando así una evaluación más contextualizada de la precisión de cada método de imputación.



Imputación conjunta de nutrientes inorgánicos ($N\text{-}NH_4$, $N\text{-}NO_2$, $N\text{-}NO_3$):

Se detectaron 39 registros con valores faltantes simultáneamente en las columnas $N\text{-}NH_4$ (µg/l), $N\text{-}NO_2$ (µg/l) y $N\text{-}NO_3$ (mg/l) sobre un total de 1231 registros, lo que representa un 3,16% del total. Dado que las ausencias coincidían en los tres casos, se adoptó una estrategia de imputación conjunta que respetara la interdependencia química entre las distintas formas de nitrógeno. Se utilizó una estrategia de imputación combinada:

Segmentación por grupo ambiental: Los registros fueron agrupados por combinación de Sitio de monitoreo y Estación del año.

Definición de variables predictoras

Para cada grupo, se consideraron como variables predictoras:

- PHT (µg/l)
- PRS (µg/l)

- Clorofila (µg/l)
- Total Algas Sumatoria (Cel/L)
- Cianobacterias Total
- Temperatura (°C)

Modelo predictivo multisalida

Se entrenó un modelo de Random Forest Regressor con estructura de salida múltiple (MultiOutputRegressor), lo que permitió predecir de forma simultánea las tres variables de nitrógeno. Esta estrategia se eligió por su capacidad para:

- Capturar relaciones cruzadas entre $N\text{-}NH_4$, $N\text{-}NO_2$ y $N\text{-}NO_3$..
- Evitar inconsistencias lógicas en las proporciones relativas entre compuestos.
- Adaptarse a la alta variabilidad natural del sistema ambiental.

El modelo se ajustó utilizando exclusivamente registros completos del grupo (sin faltantes en las tres variables objetivo ni en las predictoras). Se aplicó validación cruzada interna (3 particiones) para estimar el error medio absoluto (MAE) de cada variable dentro del grupo.

Mediana local

Como alternativa al modelo, se calculó la mediana local de cada variable de nitrógeno dentro del grupo. Esta medida fue utilizada como imputación constante para los valores faltantes cuando el modelo no resultaba más preciso.

Tratamiento del error de imputación

El análisis del error se realizó a nivel de grupo. En cada grupo con al menos 5 registros completos, se siguió el siguiente procedimiento para cada variable:

- Se entrenó un modelo Random Forest sobre esos registros.
- Se compararon las predicciones internas del modelo con los valores reales.
- Paralelamente, se imputó la mediana del grupo y se comparó con los mismos valores reales.
- En ambos casos, el error se calculó usando la métrica MAE (Error Absoluto Medio).
- Además, se estimó el porcentaje de error relativo respecto al rango de valores reales del grupo. Este indicador permitió analizar si un error era realmente significativo en función de la variabilidad del grupo.

Criterio de selección

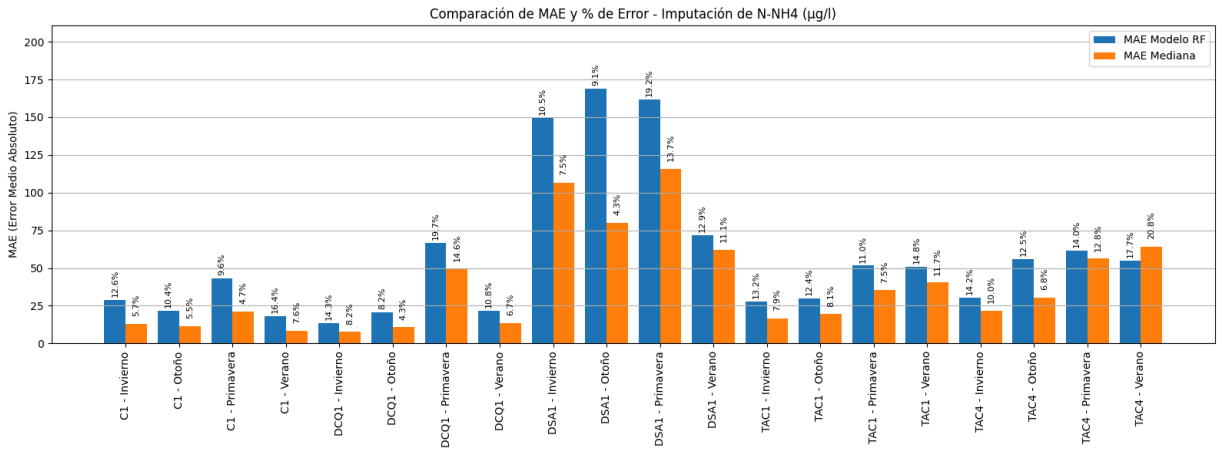
El valor imputado en cada grupo se eligió según el menor error:

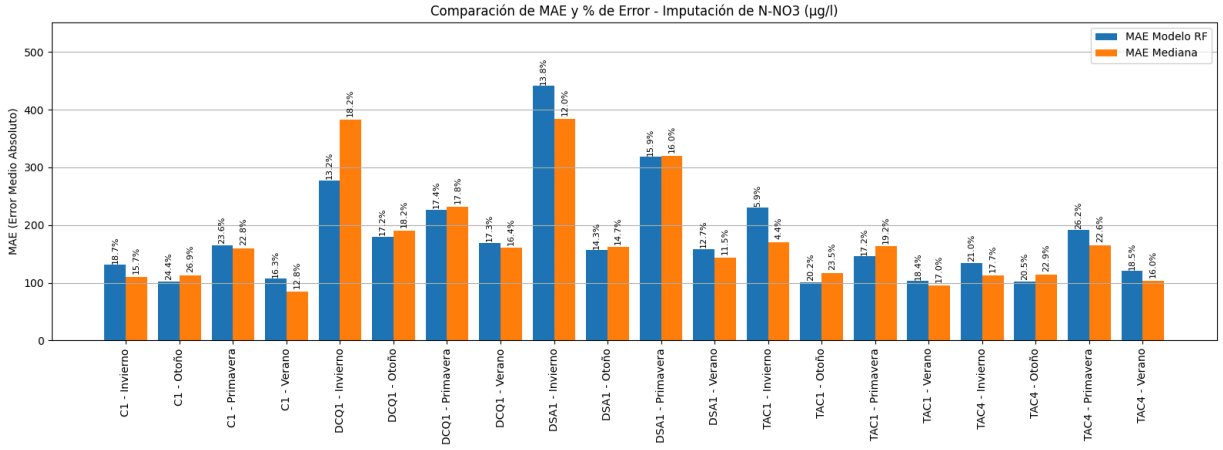
- Si el modelo presentaba menor MAE que la mediana, se aplicó la predicción del modelo.
- Si la mediana era más precisa, se imputó con ese valor constante.

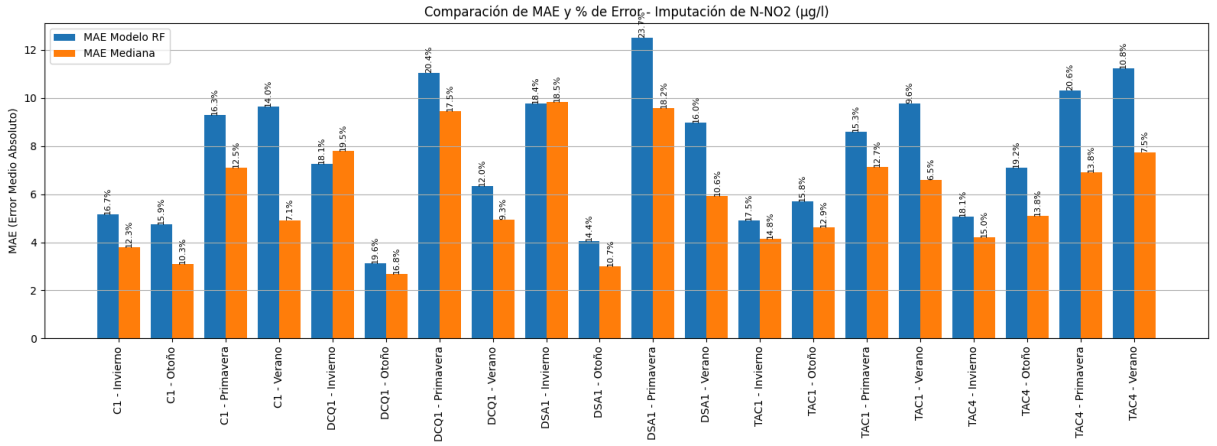
Resultados

En las siguientes figuras se presentan los gráficos de barras que comparan el MAE del modelo y de la mediana para cada combinación de sitio de monitoreo y estación del año. Cada gráfico corresponde a una de las tres variables de nitrógeno inorgánico: N-NH_4, N-NO_2 y N-NO_3.

Sobre cada barra se indica el porcentaje de error relativo, calculado como el **MAE dividido por el rango real de valores observados en ese grupo**. Este indicador permite visualizar en qué medida el error medio representa una porción significativa del rango del grupo, facilitando así una evaluación más contextualizada de la precisión de cada método de imputación.



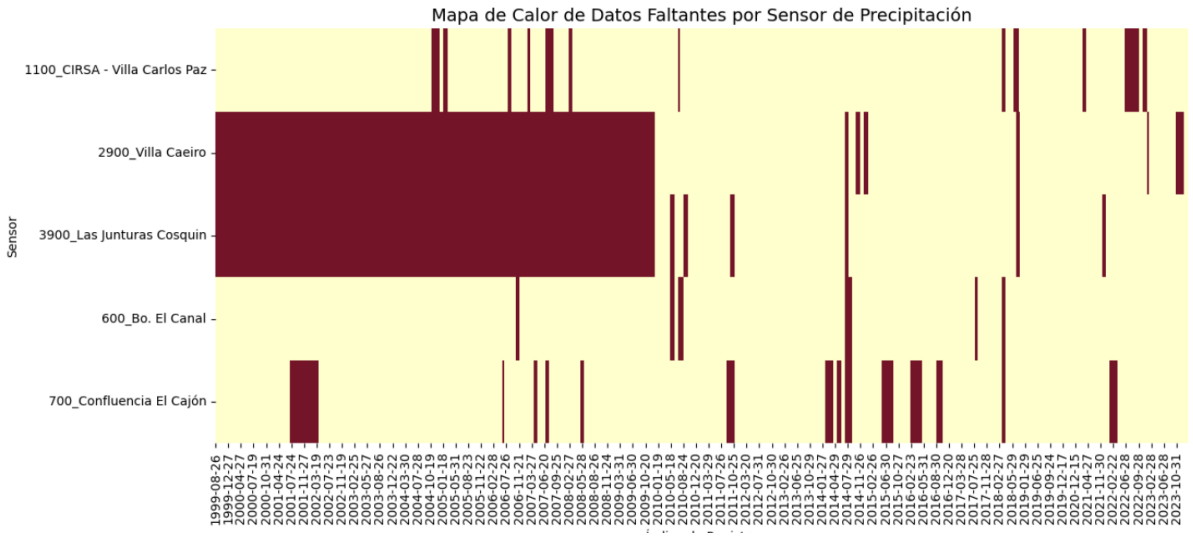




Análisis sobre los datos de precipitación de la base de hidrometeorología, *alerts.*

Se realizó un análisis sobre cada uno de los sensores seleccionados:

Detección de valores faltantes:



El gráfico presenta un mapa de calor de datos faltantes para los cinco sensores de precipitación seleccionados, abarcando el rango temporal que coincide con los datos disponibles en la base de datos *"Water Quality"*. En la visualización, los valores amarillos representan presencia de datos válidos, mientras que los sectores rojos indican períodos con datos faltantes.

Se destacan las siguientes observaciones:

- Sensor 2900_Villa Caeiro: presenta la mayor cantidad de datos faltantes, con el 47.77%, una gran franja continua sin registros desde el año 1999 hasta mediados de 2009. Esto se debe a que el sensor no estaba disponible en el inicio de los registros de precipitaciones.
- Sensor 3900_Las Junturas Cosquín: como el sensor 2900 también muestra una cantidad significativa de ausencias, el 47.52% en el mismo periodo, y además de los faltantes intermitentes a lo largo del resto del período. Esto afecta su confiabilidad para modelos basados en series temporales completas.
- Sensor 700_Confluencia El Cajón: presenta períodos aislados de ausencia de datos, pero en general mantiene una cobertura aceptable, 11.29% de datos faltantes. Es destacable una interrupción prolongada entre 2000 y 2001.
- Sensor 1100_SCIRSA - Villa Carlos Paz y 600_Bo El Canal: son los sensores más completos y confiables en cuanto a continuidad de los datos, con faltantes relativamente escasos y distribuidos de forma esporádica (600_Bo El Canal 2.84% y 1100_SCIRSA - Villa Carlos Paz 6.58% de datos faltantes).

Con este análisis se tomó la decisión de excluir los sensores 2900 y 3900 del modelado principal, ya que ambos presentan más del 45% de sus datos faltantes. Este nivel de incompletitud compromete gravemente la representatividad temporal de las series, impidiendo capturar adecuadamente patrones estacionales o tendencias de largo plazo. Además, la proporción tan alta de imputación requerida excede los límites razonables para cualquier método de reconstrucción, ya que cuando la mayoría de los valores deben ser estimados en lugar de observados, la incertidumbre del proceso supera la capacidad de los modelos para generar resultados confiables. Por lo tanto, su inclusión podría introducir sesgos significativos o falsos patrones, afectando negativamente la validez de las conclusiones.

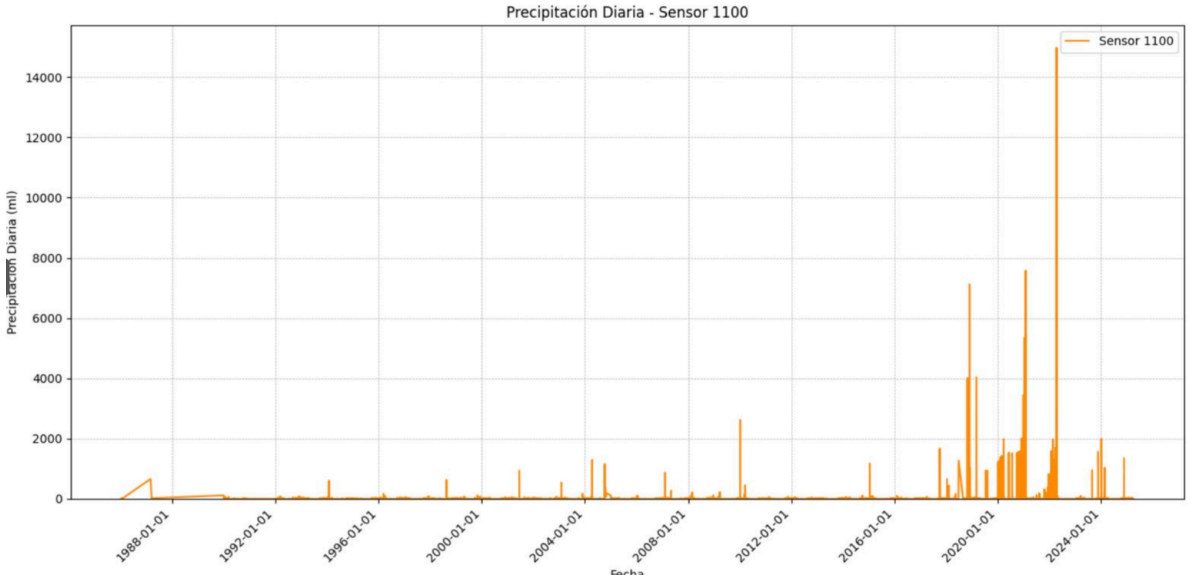
Análisis de series temporales

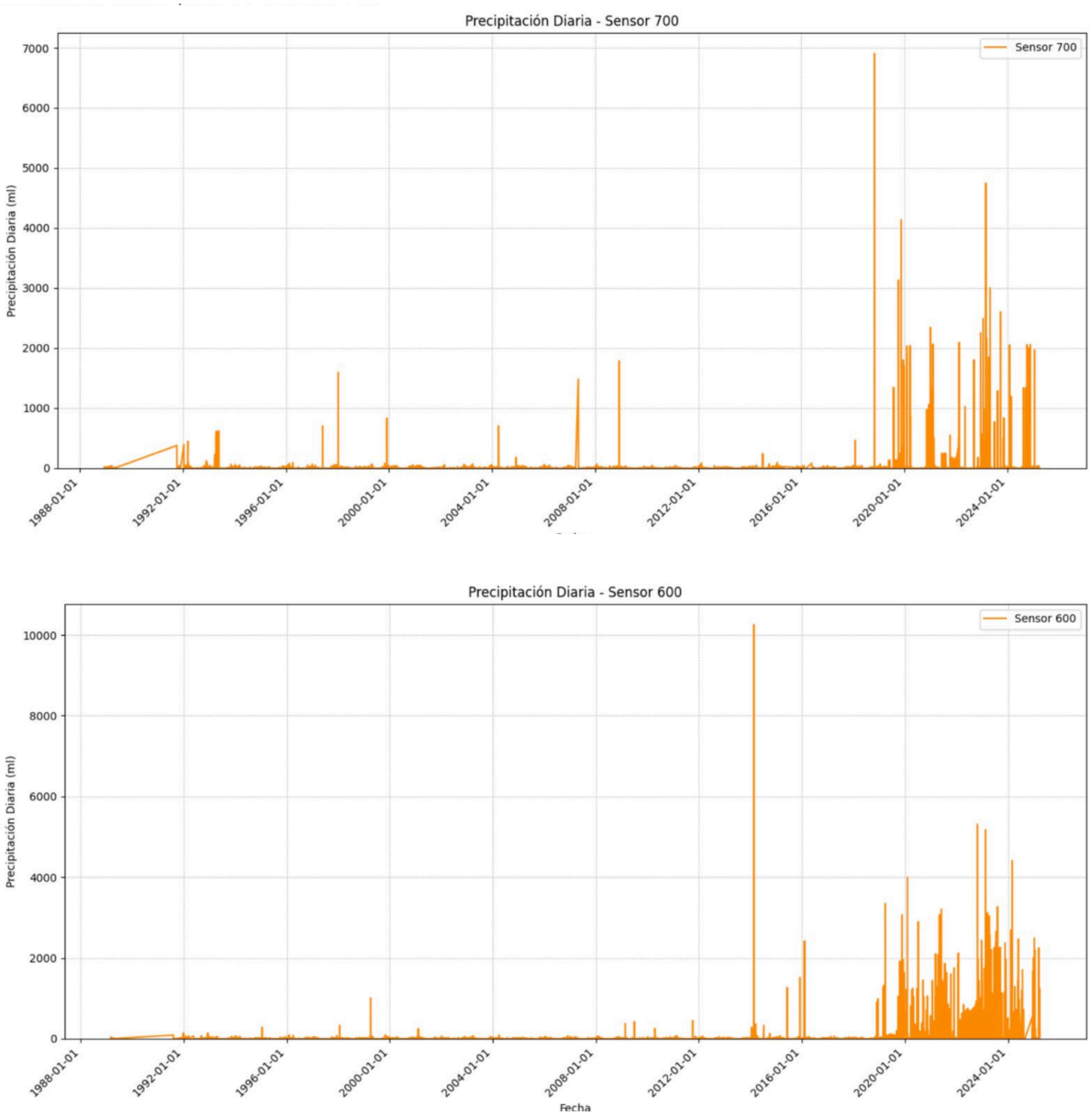
Previo al análisis cabe destacar, que como se mencionó antes, existe un proceso ETL, y un proceso en el que un experto es el encargado de completar la columna *observation,* con la palabra “Perturbación”. El procedimiento consta de identificar como anomalías aquellos saltos abruptos —por ejemplo, diferencias superiores a 500 mm en un intervalo corto—, que no son compatibles con precipitaciones reales y podrían deberse a errores de transmisión, reinicios no documentados o corrupción de datos.

Para este análisis se decidió estimar la precipitación diaria en cada uno de los sensores, en la vista precipitación diaria, mencionada previamente, la cual fue creada para el análisis.

Análisis perturbaciones:

Estos gráficos representan la evolución de la precipitación diaria registrada por cada sensor desde 1988 al 2024.





El análisis de las series de precipitación diaria de los sensores 600 (Bo. El Canal), 700 (Confluencia El Cajón) y 1100 (SCIRSA - Villa Carlos Paz) muestra un patrón común: hasta aproximadamente el año 2020, las series presentan valores en su mayoría bajos o moderados, con algunos picos aislados. Sin embargo, a partir de 2019–2020 se evidencia un cambio abrupto en la dinámica de los datos, con un incremento marcado en la magnitud de los registros, incluyendo numerosos eventos que superan ampliamente los 5000 e incluso los 10.000 mm diarios.

Estas magnitudes exceden por completo los límites físicos de precipitación posible y sugieren la aparición de perturbaciones sistemáticas no detectadas adecuadamente por el proceso ETL, posiblemente vinculadas a errores de lectura del acumulador, reinicios mal

tratados o fallas en el procesamiento. Esta situación exige una depuración rigurosa de los registros extremos para asegurar la confiabilidad de las series para el uso del modelo.

Limpieza de datos

Detección y subsanación de perturbaciones en base *alerts*

Para garantizar que la precipitación acumulada refleje sólo variaciones meteorológicas reales, adaptamos y ampliamos los scripts iniciales del trabajo realizado por Andreoni (2020) , desarrollados y aplicados en SQL. Estos scripts detectan como anomalías aquellos saltos o caídas abruptas que no se ajustan con el comportamiento normal de un pluviómetro.

Lógica original y sus límites
En Andreoni (2020) se indicaba como perturbación cualquier valor que fuera menor (caída) o mayor (subida) que sus dos vecinos inmediatos, siempre que el valor siguiente fuera mayor o igual al anterior.
Esto funciona bien para picos aislados, pero falla en dos casos frecuentes:

- Secuencias erráticas prolongadas, donde varios registros consecutivos fluctúan sin patrón claro. Por ejemplo: En esta secuencia el sensor está dando valores aleatorios tras una falla, sin seguir un patrón.

204	<null>
204	<null>
8	Perturbacion (caida)
204	<null>
11	Perturbacion (caida)
204	<null>
25	Perturbacion (caida)
24	Perturbacion (caida)
25	Perturbacion (caida)
212	Perturbacion (subida)
204	<null>

- **“Mesetas”**, bloques de valores constantes que no corresponden a acumulaciones reales.
  Por ejemplo: En esta secuencia se detecta una repetición de la falla

662	<null>
662	<null>
662	<null>
24	Perturbacion (caida)
24	Perturbacion (caida)
662	<null>

Mejoras introducidas

- Detección en cuatro etapas: en lugar de combinar subidas y caídas en un solo pase, ejecutamos secuencialmente detección de caídas, subidas, una segunda detección de caídas y, finalmente, una segunda de subidas.
    - Esto permite capturar casos como la serie 204 → 25 → 24 → 25 → 204, donde el “24” se detecta en la primera ejecución del script de caída y los “25” en la primera ejecución del script de subida, evitando que queden sin marcar.

    - En este segundo caso con la lógica anterior, al haber detectado el “25” como caída, el “63” se ignoraba, el “44” se detectaba como caída, y se detectaba el 684 (parte de la secuencia correcta del contador) como perturbación subida.

684	<null>
684	<null>
25	Perturbacion (caida)
63	Perturbacion (caida)
44	Perturbacion (caida)
684	<null>
58	Perturbacion (caida)
58	Perturbacion (caida)

- Agrupamiento de “mesetas”: identificamos grupos de valores idénticos y los comparamos con los grupos adyacentes para atrapar bloques constantes de error que la lógica puntual no detecta. Por ejemplo en este caso antes el 6 no se detectaba, y ahora se detecta como meseta de error.

307	<null>
307	<null>
7	Perturbacion (caida)
307	<null>
6	Perturbacion (caida)
6	Perturbacion (caida)
16	Perturbacion (subida)
307	<null>

Correcciones manuales y precauciones

Tras las cuatro fases automatizadas, quedan casos muy erráticos que solo se corrigen revisando manualmente los días con precipitación diaria por encima de 200 ml.

Por ejemplo:

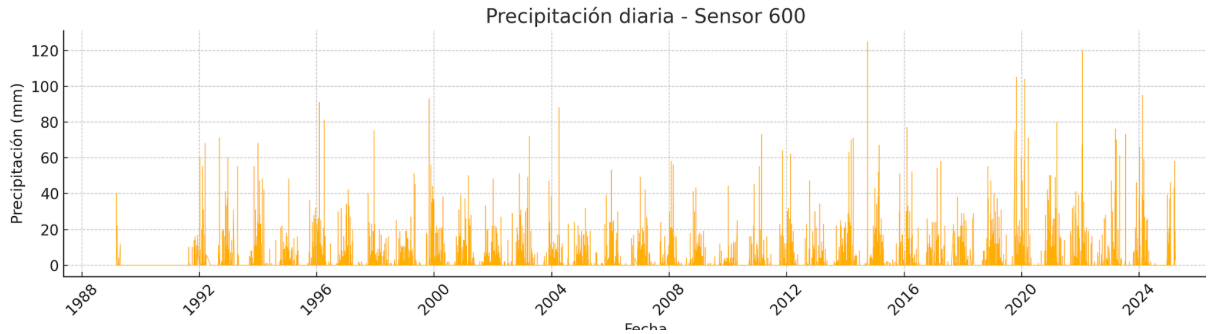
662	<null>
662	<null>
48	Perturbacion (caida)
48	Perturbacion (caida)
56	Perturbacion (caida)
24	Perturbacion (caida)
662	<null>
662	<null>

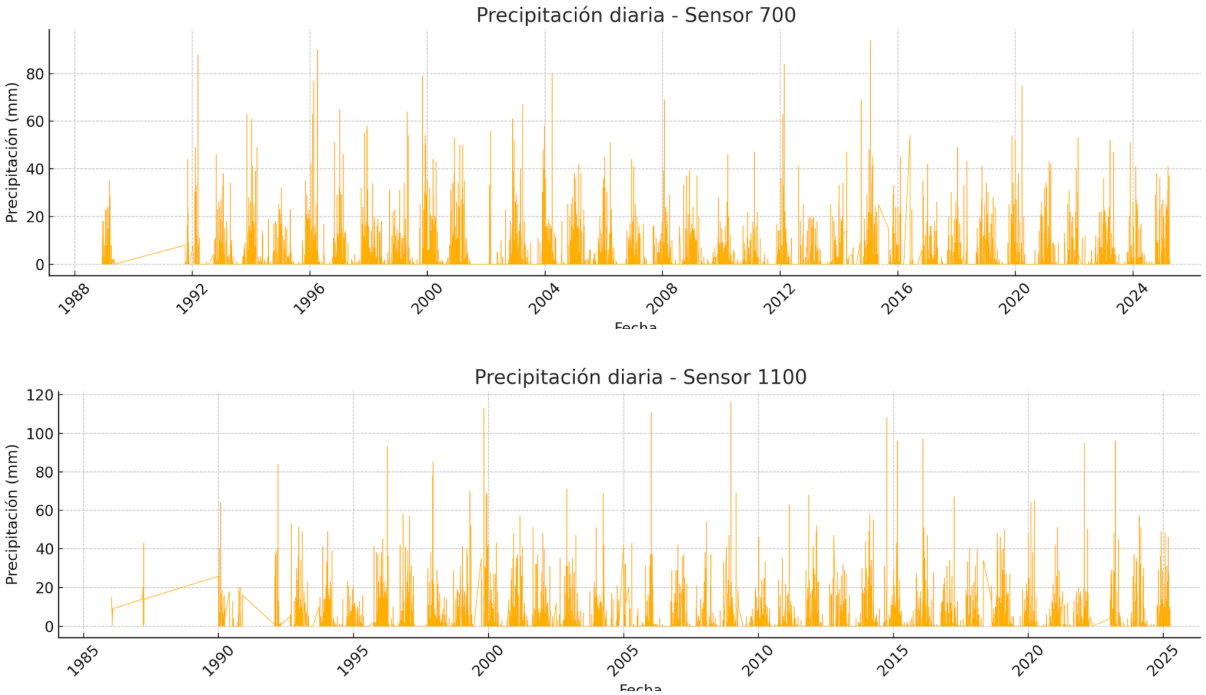
Evitamos ejecutar más de dos ciclos de cada script porque, ya que incrementa el riesgo de marcar como errores comportamientos naturales de la serie. Esto se debe a que en las ejecuciones previas, puede haber detectado erróneamente algún valor válido (por la cantidad de perturbaciones entre los valores reales registrados por los sensores), y por consiguiente sus vecinos no serán valores reales. Esto puede llevar a la pérdida de datos válidos, afectando la representatividad de la información y reduciendo la confiabilidad del análisis.

Resultados

Este procedimiento resultó en los siguientes gráficos de precipitación diaria en el mismo rango temporal que los anteriores, mostrando una representación mucho más confiable y realista de las series temporales para los sensores 600, 700 y 1100.

A continuación, se presentan los principales hallazgos y observaciones:





- Eliminación exitosa de perturbaciones extremas: Ya no se observan valores imposibles como los que superaban los 5000 o 10.000 mm. Los valores máximos actuales oscilan entre 80 y 120 mm, lo cual está dentro de rangos razonables para eventos extremos reales de precipitación diaria.
- Representación más fiel de la variabilidad climática: Se observa una distribución más homogénea de eventos de lluvia en el tiempo, sin "picos artificiales". Esto permite confiar en la base para el modelo predictivo.
- Estacionalidad visible: Hay concentración de eventos de mayor magnitud entre fines de primavera y verano (en general, de octubre a marzo), especialmente notable por la repetición cíclica de picos. Esto es esperable para la región serrana de Córdoba, donde las lluvias más intensas se concentran en esos meses.
- Ausencia clara de tendencia a largo plazo: A diferencia del análisis de precipitaciones acumuladas anteriores, aquí no se observa una tendencia clara de aumento o disminución en la intensidad o frecuencia diaria.
- Diferencias entre sensores: El sensor 600 muestra una densidad de eventos elevada en años recientes, probablemente por mejor cobertura de datos.

El proceso de limpieza eliminó perturbaciones evidentes, y permitió rescatar el comportamiento real del régimen de lluvias, evidenciar patrones estacionales, y garantizar la usabilidad de las series. Estos gráficos ahora reflejan de forma fidedigna la dinámica climática regional, sentando bases más sólidas para los modelos.

Finalmente, cabe destacar que este proceso de limpieza no debería repetirse en el futuro, ya que los profesionales del INA, al conocer el estado de perturbaciones en los sensores, se comprometieron a mejorar el proceso ETL que originalmente debía encargarse de la detección de perturbaciones, incorporando una lógica más robusta y preventiva.

Transformación de variables

Creación de variables calculadas:

Se definió la generación de variables calculadas a partir de otras ya presentes en la base de datos, con el fin de facilitar el análisis y mejorar la capacidad predictiva del modelo.

a. En lugar de analizar individualmente cada género de cianobacteria, se creó una nueva variable, que representa, como su nombre lo indica, el “Total de Cianobacterias”, lo que simplifica su tratamiento como una única variable biológica clave.
    i. Suma todos los géneros de cianobacterias para obtener una única variable total.
b. Del mismo modo, se construyó la variable "Nitrógeno Inorgánico Total" (NIT) combinando las concentraciones de nitrito, nitrato y amonio, ya que los tres agrupados, son más representativos del impacto real del nitrógeno sobre la proliferación algal.
    i. Suma nitrito, nitrato (convertido a µg/l) y amonio para calcular el NIT.
    ii. Validamos que en caso de que alguna de las variables que componen la suma sea nula, la suma también lo es, que no exista el dato no quiere decir que esta sea 0.
c. También se calculó la “Dominancia de Cianobacterias (%)”, que indica el porcentaje de cianobacterias sobre el total de algas, lo que permite entender su peso relativo en el ecosistema.
    i. Calcula la dominancia como porcentaje de cianobacterias respecto al total de algas.

Las variables Total de Cianobacterias y Dominancia de Cianobacterias (%), además de aportar valor analítico, fueron seleccionadas como variables objetivo del modelo, siendo fundamentales para las predicciones que se esperan obtener.

Precipitación acumulada

El cálculo de la precipitación, se da en la creación de la vista_precipitacion_3d en la base de datos alerts, que consta de acumular el valor en mm de precipitación de 3 días incluido el día de la medición.

Una vez generada la vista con la precipitación acumulada, esta se consume desde el backend y los datos son incorporados al data frame principal. El proceso de integración se llevó a cabo de la siguiente manera:

- Se recuperaron los datos de precipitación acumulada desde la base, asegurando su correcta correspondencia por sensor y fecha.
- Se transformó la información para que cada sensor cuente con una columna específica de precipitación acumulada.
- Se realizó una unión por fecha entre los registros de calidad de agua y los datos de precipitaciones.
- En los casos donde un sensor no registró datos para una fecha específica, se imputó un valor nulo (NaN), conservando la integridad del dataset.

Este tratamiento permite contar, para cada muestra de calidad de agua, con información contextualizada sobre el régimen hídrico reciente, contribuyendo a mejorar la capacidad explicativa del modelo respecto de fenómenos asociados a eventos de lluvia.

IMPLEMENTACIÓN

BASE DE DATOS

El modelo se integra con datos provenientes de dos bases de datos relacionales, provistas por el Instituto Nacional del Agua (INA), ambas implementadas en PostgreSQL:

- water_quality: contiene registros históricos sobre parámetros fisicoquímicos y biológicos relacionados con la calidad del agua.
- alerts: almacena datos hidrometeorológicos obtenidos de sensores automáticos distribuidos en distintos puntos de la cuenca del embalse.

Con el fin de optimizar el acceso y procesamiento de los datos, se implementaron vistas en ambas bases. Las vistas actúan como estructuras intermedias que encapsulan la lógica de filtrado, transformación y consolidación de información, desplazando parte de la carga

computacional hacia el motor de base de datos y simplificando el procesamiento en el backend.

Este enfoque presenta ventajas significativas:

- Eficiencia: reduce la cantidad de consultas complejas ejecutadas desde la aplicación.
- Modularidad: permite modificar criterios o estructuras sin alterar el código del backend. El backend accede directamente a una estructura optimizada, sin necesidad de gestionar consultas complejas.
- Escalabilidad: mejora la gestión de grandes volúmenes de datos al evitar múltiples operaciones JOIN en tiempo de ejecución. Y permite agregar nuevos parámetros sin modificar la estructura del backend.

Base de datos Water_Quality

1. VistaConjunto (base de datos water_quality)

Esta vista fue diseñada para unificar los datos más relevantes sobre calidad del agua, considerando únicamente los puntos de monitoreo definidos como críticos para el modelo predictivo, y filtrando las variables de entrada seleccionadas según criterios técnicos de los profesionales del INA.

Se aplicó un filtro sobre el campo *code* de la tabla *profile*, seleccionando únicamente perfiles de tipo subsuperficial o correspondientes a la toma de agua (TAC). Esto garantiza la homogeneidad en la profundidad de las mediciones, condición clave para la comparación temporal y espacial de los registros.

Los perfiles incluidos son:

1. C1: Centro del embalse
2. DCQ1: Desembocadura del Río Cosquín
3. DSA1: Desembocadura del Río San Antonio
4. DLM1: Desembocadura del Arroyo Las Mojarras
5. DLC1: Desembocadura del Arroyo Los Chorrillos
6. TAC1: Zona de presa (subsuperficial)
7. TAC4: Zona de presa (altura de toma para Aguas Cordobesas S.A.)

Estos sitios fueron seleccionados en conjunto con los equipos del INA debido a su importancia estratégica en la dinámica del ecosistema, tanto por su ubicación (puntos de

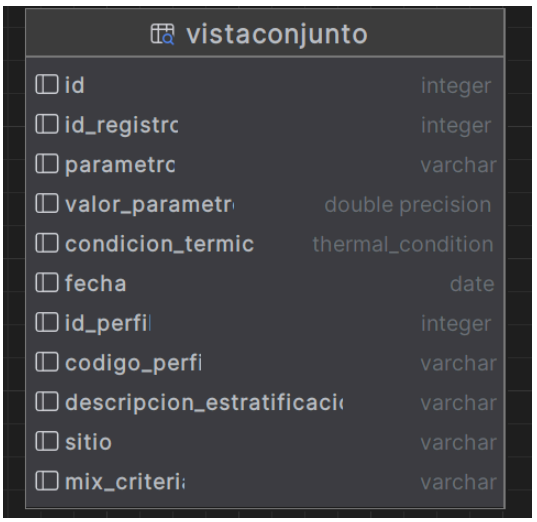
entrada y dentro del cuerpo principal) como por su historial de registros con alta variabilidad. Se filtraron aquellas tuplas, en las que se registraba el valor para las siguientes variables, definidas como variables de entrada para el modelo:

1. Clorofila (µg/l)
2. Cota (m)
3. Total Algas Sumatoria (cel/L)
4. Condición térmica
5. T° (°C)
6. PHT (µg/l)
7. PRS (µg/l)
8. N-NH_4 (µg/l)
9. N-NO_2 (µg/l)
10. N-NO_3 (mg/l)
11. Algas cianobacterias (Anabaena, Anabaenopsis, Aphanizomenon, Aphanocapsa, Aphanothece, Chroococcus, Dolichospermum, Geitlerinema, Leptolyngbya, Merismopedia, Microcystis, Nostoc, Oscillatoria, Phormidium, Planktothrix, Pseudoanabaena, Raphydiopsis, Romeria, Spirulina, Synechococcus)

Estructura y unión de tablas

La vista se construye mediante la unión de múltiples tablas del esquema relacional original, combinando información sobre los parámetros medidos, sus valores, el contexto espacial y temporal del muestreo y las condiciones térmicas registradas. Las tablas y atributos incluidos son:

1. Measurement: contiene identificador de la medición, vínculo con el registro, el identificador del parámetro y el valor medido del mismo (id, record_id, parameter_id, value)
2. Parameter: define identificador y el nombre
3. Record: incluye la condición térmica del agua
4. Monitoring: provee la fecha exacta de monitoreo
5. Sample: asocia cada muestra a un perfil
6. Profile: describe el perfil de medición, incluyendo código y tipo de estratificación
7. Site: contiene el nombre del sitio de monitoreo



Esta estructura permite generar un único conjunto de datos que consolida toda la información necesaria para el análisis predictivo, evitando múltiples consultas cruzadas y asegurando la alineación espacial, temporal y contextual de cada observación.

Actualización automática de datos

Para garantizar que el modelo de predicción siempre trabaje con la información más reciente de calidad de agua, se implementó un mecanismo automático de notificación en la base de datos water_quality, concretamente sobre la tabla measurement. A continuación se describe su configuración, propósito y efectos en el flujo de datos:

- **Trigger**
  Se crearon triggers que se activan tras cada operación de inserción o actualización en las tablas measurement, record, sample y monitoring. Se configuró cada trigger como “por sentencia” (FOR EACH STATEMENT) para que, incluso si una sola operación SQL afecta a decenas o cientos de filas a la vez, se envíe una única notificación en bloque. De este modo se evitan múltiples avisos redundantes y no se sobrecarga el canal de notificaciones.
- **Función de notificación**
  Al dispararse uno o más triggers, se invoca una función interna de PostgreSQL cuya única misión es emitir una señal por medio del mecanismo LISTEN/NOTIFY. Esta señal—enviada al canal datos_agua_actualizados con un mensaje estándar (“Hubo un cambio en los datos de agua”)—no modifica los registros, sino que alerta al sistema de backend sobre la necesidad de actualizar el conjunto de datos base.

Se optó por centrar la notificación en los datos de calidad de agua por lo siguiente:

- **Frecuencia de actualización**: las mediciones de calidad de agua se registran de forma mensual, mientras que las variables hidrometeorológicas pueden llegar con mayor frecuencia (cada 12 horas), y no se puede hacer una predicción nueva sin los datos de calidad, por lo que se corre todo el proceso una vez que estos estén cargados en un nuevo mes.
- **Prioridad para el modelo**: cualquier nuevo dato de calidad de agua representa un punto crítico para la predicción de cianobacterias, justificando la re-ejecución del pipeline de preprocesamiento en esta instancia.

Base de datos alerts

La base de datos *alerts* almacena datos hidrometeorológicos obtenidos a través de sensores automáticos instalados en distintas estaciones dentro de la cuenca del Embalse San Roque. Estos datos son fundamentales para complementar la información de calidad de agua y permitir la modelización de procesos ecológicos, como la floración de cianobacterias.

Dado que los registros de esta base incluyen múltiples variables y sensores, se diseñaron vistas específicas con el objetivo de estructurar los datos de forma clara, eficiente y orientada a las necesidades del modelo predictivo.

Objetivos de las vistas en alerts

- Extraer únicamente las variables de interés para el análisis (precipitación y temperatura del aire).
- Agrupar y consolidar los valores por día, facilitando su alineación con los registros de calidad de agua.
- Implementar reglas específicas de cálculo para obtener nuevas variables derivadas, como la precipitación acumulada.
- Optimizar la consulta desde el backend, evitando operaciones costosas en tiempo de ejecución.

1. Vista vista_alerts: filtrado de registros relevantes

Esta vista constituye el primer paso en el tratamiento de los datos hidrometeorológicos. Su función es extraer únicamente los registros de las variables hidrometeorológicas útiles para el modelo, según el criterio de los profesionales del INA:

- Precipitación

- Temperatura del aire

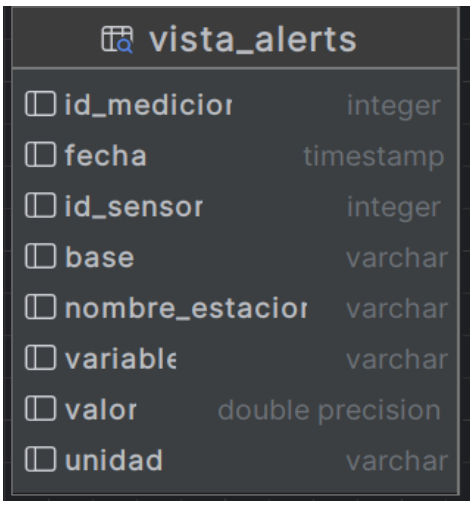
Al igual que en la base de calidad de agua se aplicó un filtrado por sitios de monitoreo, en el caso de las variables hidrometeorológicas se seleccionaron cinco estaciones representativas de distintas subcuencas. Estas estaciones fueron escogidas por su cobertura espacial, que coincide con las áreas correspondientes a los sitios de medición previamente definidos en la base de calidad de agua:

1. 600: Bo. El Canal
2. 700: Confluencia El Cajón
3. 1100: SCIRSA – Villa Carlos Paz
4. 2900: Villa Caeiro
5. 3900: Las Junturas Cosquín

Cada registro proveniente de los sensores en la base de datos incluye un campo de observaciones que señala posibles perturbaciones durante la medición. Estos comentarios indican interferencias o condiciones que comprometen la validez del dato registrado. Por este motivo, se optó por excluir del análisis todos los registros que presentaban observaciones, con el fin de evitar la incorporación de valores alterados que pudieran introducir ruido en los datos de entrada y afectar negativamente el rendimiento del modelo predictivo.

Campos generados por la vista:

1. Identificador del sensor
2. Nombre de la estación y cuenca (base)
3. Fecha de registro
4. Variable registrada (Precipitación o Temperatura)
5. Valor observado y unidad de medida
6. Observaciones (debe ser nulo)



Esta vista constituye la base sobre la cual se construyen las vistas más especializadas.

2. Vista precipitación diaria: cálculo de precipitación por día

La lógica implementada en esta vista consiste en:

1. Cálculo estándar de precipitación diaria: se toma la diferencia entre el último y el primer valor registrado en el día. Si esta diferencia es mayor o igual a cero, se interpreta como el valor acumulado correctamente por el sensor durante esa jornada.
2. Manejo de reinicio del contador: si la diferencia es negativa, lo que indica un posible reinicio del contador, se calcula la suma de las diferencias entre cada par de registros consecutivos del día. En este caso, cada diferencia negativa se reemplaza por un valor fijo de 1 mm, asumiendo que se trata de un reseteo parcial del sensor, como práctica común para evitar subestimar la precipitación real.
3. Días con un solo registro: si en un día se cuenta con solo un valor válido, se considera que no hay forma de estimar una variación, y la precipitación diaria se asigna como 0 mm, bajo el supuesto de que no hubo acumulación apreciable o que el evento comenzó o terminó fuera del rango de registros.
4. Ausencia total de datos válidos: para aquellos días en los que no se cuenta con ningún registro, o bien todos los registros del día han sido marcados como perturbación, el valor de precipitación diaria se deja como nulo, ya que no existen datos confiables sobre los cuales realizar el cálculo.

3. Vista precipitacion_acumulada_3d: cálculo de lluvia acumulada

Para obtener una estimación confiable de la precipitación acumulada en tramos de tres días consecutivos, se construyó una vista que garantiza la continuidad temporal del calendario y

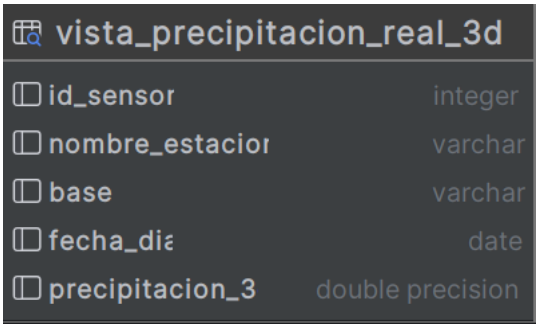
preserva la validez de los datos reales. Esta vista se compone de tres etapas fundamentales:

1. Generación de la serie de fechas completas: se crea una combinación entre todos los días del período 1999-07-24 en adelante, y cada uno de los sensores de precipitación disponibles. Esto asegura que incluso los días sin registros explícitos queden representados como filas con valores nulos.
2. Integración con los datos reales: se realiza un LEFT JOIN entre la serie completa de fechas y la vista de precipitaciones diarias previamente calculada, incorporando la precipitación real en los días donde está disponible y dejando NULL cuando no hay datos válidos.
3. Cálculo de acumulado móvil condicional: mediante funciones de ventana, se calcula la suma de precipitación en un tramo de tres días consecutivos para cada sensor. Esta suma se realiza solo si al menos uno de los tres días tiene un valor válido. En caso de que los tres días contiguos tengan valores nulos, se asigna NULL al acumulado, para evitar generar datos artificiales o distorsionados.

Este enfoque permite preservar la estructura temporal real de las precipitaciones, evitando interpolaciones implícitas y manteniendo la trazabilidad de los datos faltantes dentro del análisis.

Campos de la vista resultante:

1. id_sensor,
2. nombre_estacion
3. fecha_dia: fecha
4. precipitacion_3d: suma de lluvias en los tres días previos



Este valor es utilizado directamente como una de las variables de entrada del modelo predictivo.

4. Vista vista_temperatura: consolidación de extremos térmicos diarios

La temperatura del aire es una variable clave en los procesos de estratificación térmica del embalse y en la dinámica de crecimiento de cianobacterias.

Luego de un análisis por parte de los profesionales del INA, se mantuvo que lo más representativo para el modelo, era analizar los picos de temperatura máximos y mínimos por día. Esta vista agrupa todos los registros diarios de temperatura por fecha y calcula:

- Temperatura máxima del día
- Temperatura mínima del día

Se utilizan las mismas estaciones que en las vistas anteriores, y también se excluyen registros con observaciones.

Campos generados:

1. fecha_dia: fecha del registro
2. temperatura_max: valor máximo diario
3. temperatura_min: valor mínimo diario

Estas variables permiten modelar tanto la amplitud térmica como el régimen de temperatura diaria, ambas asociadas a fenómenos de mezcla o estratificación en la columna de agua.

Integración con el Backend

Desde el backend, se realiza la conexión a las bases de datos alerts y water_quality mediante SQLAlchemy.

La vista, vista_conjunto, de water_quality es la base para el inicio del proceso de construcción del data frame. Mientras que vista_condicion_termica, se consume en una función específica del backend para realizar el cálculo de la variable.

Las vistas precipitacion_acumulada_3d y vista_temperatura se consultan y se integran utilizando un merge por la columna fecha_dia.

El resultado es un único Data Frame que para cada día posee:

1. Valores de temperatura máxima
2. Valores de temperatura mínima.
3. Precipitación acumulada en los tres días previos, discriminada por sensor.

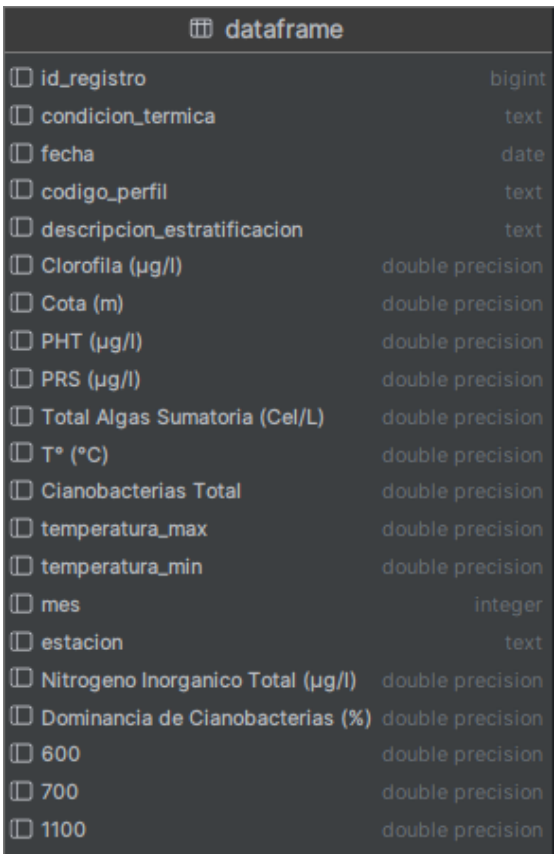
Este Data Frame se alinea posteriormente al Data Frame completo, con los registros de calidad de agua, según la fecha, generando el conjunto de datos final utilizado en los procesos de entrenamiento y predicción.

Base de datos model_data

La base de datos model_data, alojada en PostgreSQL, funciona como el repositorio central de toda la información necesaria para el entrenamiento de los modelos de predicción y, a su vez, como archivo histórico de las predicciones realizadas. En ella convergen dos componentes esenciales: el conjunto de datos procesados que alimenta el pipeline de entrenamiento (data frame) y el registro cronológico de cada predicción, que luego se exhibe en el módulo de “Predicciones Históricas” del frontend.

La tabla dataframe contiene el DataFrame definitivo que ha sido extraído, limpiado, e imputado durante el proceso de creación del mismo en el backend. Cada fila representa un mes de medición para un sitio de muestreo concreto (identificado por su código de perfil) y agrupa las variables de entrada como, clorofila, cianobacterias, temperatura, prs, pht, etc. El script de entrenamiento lee esta tabla directamente, garantizando que los modelos siempre se ajustan sobre el mismo estado de los datos y evitando ambigüedades en el origen de la información.

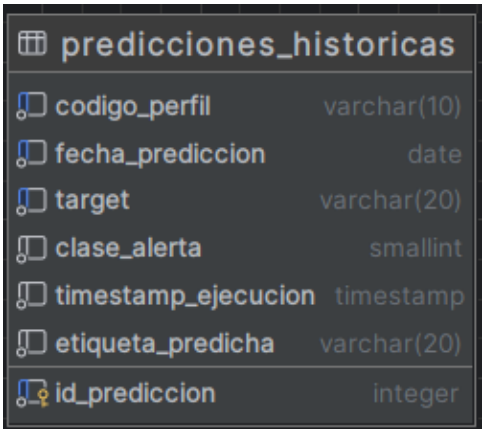
Dando como resultado una tabla con las siguientes columnas:



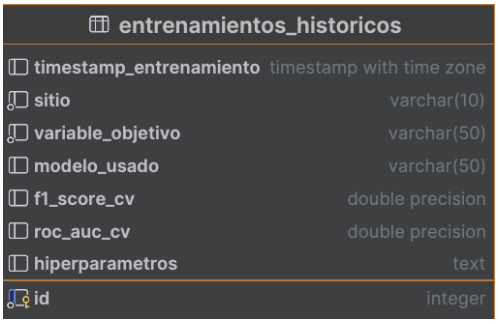
Por otro lado, la tabla **predicciones_historicas** registra cada ejecución de inferencia que el sistema realiza—tanto si se solicita para un único sitio como si se pide de manera masiva para todos—almacenando el código de perfil, la fecha del mes predicho, la variable objetivo (Clorofila, Cianobacterias y/o Dominancia), la clase de alerta resultante y la marca temporal de la ejecución. De este modo, se construye un archivo de serie temporal que permite al frontend ofrecer al investigador un historial completo de los resultados obtenidos.

Dando como resultado una tabla con la siguiente información:

- id
- codigo_perfil
- variable_objetivo
- fecha_a_predecir
- clasificación
- timestap_prediccion



Y por último, contiene la tabla **entrenamientos_historicos,** en la que se almacenan las métricas y los modelos elegidos para cada sitio y variable, luego de un proceso de entrenamiento del modelo. Con esto logramos un archivo de serie temporal que permite al frontend generar gráficos de tendencias sobre las métricas de los modelos a lo largo del tiempo.



BACKEND

El backend, desarrollado en Flask, gestiona la conexión con la base de datos, el preprocesamiento de datos, el entrenamiento del modelo, las predicciones y la exposición de endpoints para que frontend consuma.

Tecnologías utilizadas

- Python como lenguaje principal.
- Flask para exponer la API.
- pandas para la creación y manipulación de data frame (filtrar, agrupar, eliminar y transformar datos).
- sqlalchemy: conectar Python con bases de datos SQL
- seaborn y matplotlib.pyplot: Construyen gráficos (dispersión, distribuciones, matrices y mapas de calor) facilitando la exploración de relaciones entre variables.

- numpy: Soporte para funciones matemáticas, cálculos numéricos intensivos y manipulación de grandes conjuntos de datos en machine learning.
- Lineal Regression, Random Forest y Redes Neuronales MLP para el modelo de predicción.
- PostgreSQL + SQLAlchemy para el almacenamiento de datos.

Funcionalidades del Backend

Ingesta de datos

El backend se conecta a las bases de datos (water_quality y alerts), consume las vistas de cada una.

Tratamiento de datos

Comienza un proceso para construir el dataframe final, que será utilizado por el modelo de machine learning, como entrada. En este módulo se aplican las estrategias de datos faltantes, atípicos, transformación y cálculo de variables, que se desarrollarán más adelante.

Almacenamiento de dataframe

Además se conecta a la base de datos *model_data,* en el que almacena el dataframe final, para luego consumirlo en el entrenamiento y predicción del modelo, evitando tener que construir el mismo, en cada entrenamiento del modelo, y para poder enseñarlo en el frontend.

Entrenamiento de los modelos

Para separar el costoso proceso de entrenamiento del servicio en vivo, se creó un script independiente. Su única función es entrenar, evaluar, guardar los modelos y las métricas resultantes de los modelos elegidos para cada sitio y variable. Se ejecuta automáticamente cuando se detectan cambios en el data frame, para mantener los modelos actualizados y poder aprovechar el ingreso de nuevos datos, para mejorar las métricas de los mismos.

Las funcionalidades dentro de este módulo son:

1. Ingesta de Datos: Lee el dataframe desde la base model_data.
2. Ingeniería de Features: Aplica transformaciones específicas para el entrenamiento, como la creación de variables de retardo (lags).
3. Entrenamiento y Optimización: Ejecuta el pipeline que realiza la división temporal de los datos, la optimización de hiperparámetros (GridSearchCV, KerasTuner), el

entrenamiento y la evaluación de todos los modelos candidatos (Random Forest, MLP, CNN, etc.), que se explicara con más detalle más adelante en el documento.

4. Persistencia de artefactos: Tras identificar el mejor modelo para cada sitio y variable, guarda un "paquete de artefactos" en la carpeta /modelos_entrenados. Estos componentes se almacenan en un diccionario global en memoria, estructurado por *target* (“Clorofila”, “Cianobacterias”, “Dominancia de Cianobacterias”) y por *sitio* de muestreo. Así, cuando llegue una petición, el sistema ya tiene todo lo necesario sin penalizar la latencia, evitando el acceso al disco en cada solicitud. Este paquete incluye:
    - El objeto del modelo entrenado (guardado como .keras para TensorFlow o serializado dentro del .pkl para Scikit-learn).
    - Los objetos StandardScaler y KNNImputer ya ajustados.
    - La lista de nombres de las features en el orden exacto que el modelo espera.
5. Persistencia de Entrenamientos: Almacena el historial de rendimiento de los modelos luego de los entrenamientos. En el módulo en el que se re-entrenan los modelos, después de identificar el mejor modelo, se inserta un nuevo registro en la tabla entrenamientos_historicos, de la base model_data, para ser utilizados para renderizar un gráfico de monitoreo del modelo.

Actualización del dataframe, reentrenamiento y recarga de los modelos

Para garantizar que el backend siempre trabaje con los datos más recientes de calidad de agua sin interrumpir el servicio, se implementa un mecanismo basado en un listener permanente y un worker de procesamiento. Ambos componentes se ejecutan en hilos daemon (background threads), lo que permite a la aplicación web continuar atendiendo solicitudes sin bloqueos. Este proceso consta de los siguientes componentes:

1. Listener de Notificaciones

El listener es un hilo daemon que recibe de forma continua los NOTIFY emitidos por los triggers de la base de datos y agrupa estas señales para disparar la actualización del Dataframe, solo cuando se haya terminado de recibir notificaciones. Sus características principales son:

- Hilo daemon permanente: Se lanza al iniciar la aplicación y permanece activo mientras dure la sesión de Flask, sin bloquear ni retrasar el cierre del proceso principal.
- Suscripción a PG NOTIFY: Ejecuta LISTEN datos_agua_actualizados, para detectar cambios en las tablas con triggers configurados.

- Debounce de 10 s: Cuando llega un NOTIFY, el listener añade el nombre de la tabla que generó la notificación a un conjunto temporal de tablas con cambios (_changed_tables) y (re)inicia un temporizador de 10 segundos. Esto es útil porque, si durante ese intervalo llegan más notificaciones (por ejemplo, porque el proceso ETL modifica varias tablas) el temporizador se reinicia, evitando un proceso de actualización prematuro. Solo cuando transcurren 10s sin nuevas señales se invoca la actualización.
- Espera eficiente y resiliencia: Usa select.select() con timeout de 60s para no consumir CPU, y en caso de fallo de conexión, reintenta automáticamente cada 30s sin intervención manual.

2. Worker de Procesamiento

Cuando el debounce expira, se lanza otro hilo daemon que ejecuta el pipeline completo:

- Ejecución en hilo independiente, creando un hilo que llama a actualizar_df() (función encargada de llamar a las funciones correspondientes de cada proceso de actualización, obtener_dataframe(), reentrenar_modelos(), recargar_modelos()), liberando al listener para seguir recibiendo notificaciones.
- Contexto de Flask: Dentro de ese hilo, actualizar_df() se envuelve en with app.app_context():, asegurando acceso a la configuración, conexiones y extensiones de Flask.
- Pipeline de datos:
    1. Extracción: Invoca obtener_dataframe(), que lee las vistas SQL consolidadas y devuelve el Data Frame unificado.
    2. Validación: Comprueba que el DataFrame no sea None ni esté vacío.
    3. Almacenamiento: Reemplaza la tabla dataframe en model_data con los nuevos datos.
    4. Retraining y recarga de modelos: Llama a reentrenar_modelos() (subprocess que ejecuta entrenar_modelos.py) y, tras su éxito, a recargar_modelos(), que vuelve a leer los artefactos desde disco y actualiza el dict global modelos_cargados bajo un threading.Lock, para que mientras un worker está recargando el diccionario modelos_cargados, ningún otro proceso (por ejemplo, uno de predicción) intente leerlo o modificarlo al mismo tiempo.
    5. Bloqueo de interfaz: A su vez, mientras se corre el proceso de actualización, para evitar que los usuarios realicen predicciones mientras los modelos se están actualizando (un proceso que puede tardar varios minutos y dejar al

sistema en un estado inconsistente), se implementó un mecanismo de sincronización de estado entre el backend y el frontend.

Se optó por una estrategia de sondeo de estado (polling) por su simplicidad y robustez para este caso de uso. La lógica se centra en informar el estado actual del proceso de reentrenamiento a través de un endpoint.

- Se utiliza una variable global APP_STATUS (un diccionario Python) protegida por un threading.Lock para gestionar de forma segura el estado del sistema ('is_retraining': True/False) a través de los diferentes hilos de la aplicación. El estado se actualiza al inicio y al final de la función actualizar_df().
- Endpoint de Estado (/api/status): Se creó un endpoint GET que no requiere parámetros. Su única función es consultar la variable APP_STATUS y devolver un objeto JSON simple con el estado actual del sistema, por ejemplo: {'status': 'retraining'} o {'status': 'idle'}. Este es consumido por el front para exponer un componente que bloquea cualquier accion dentro de /predict, hasta recibir un {'status': 'idle'}, que implica que el proceso de actualizacion, reentrenamiento y recarga, finalizó.

6. Manejo de errores: Cualquier excepción en el pipeline se captura y se registra en consola; el listener permanece activo y listo para el siguiente lote de cambios.

3. Tolerancia y Resiliencia

- Reconexión automática: Si la conexión del listener a la base de datos falla, se reintenta cada 30 s, garantizando disponibilidad continua.
- Debounce robusto: El temporizador de 10 s se reinicia con cada notificación, evitando ejecuciones solapadas y asegurando que el pipeline siempre procese el estado completo de la base de datos tras un lote de cambios.
- Hilos daemon: Tanto listener como worker son daemon threads, por lo que no bloquean el shutdown de la aplicación principal.

<u>Flujo final para una predicción</u>

1. Carga de modelos y artefactos al inicio: Al arrancar la aplicación, si existen, se leen del disco todos los artefactos (.pkl, .keras), los ficheros que resultan del proceso de entrenamiento y que luego se utilizan para hacer predicciones sin tener que volver a

entrenar. En caso de que la carpeta no exista o se encuentre vacía, se realiza el entrenamiento de los modelos, llamando al script correspondiente, para obtener los modelos para cada sitio y variable.

2. Recepción de la petición: El endpoint /predict acepta un JSON con la clave "option", que puede ser el identificador de un sitio (C1, TAC1, TAC4, DCQ1, DSA1) o la palabra "Todos", y selecciona el paquete de artefactos apropiado (modelo, scaler, etc.) del diccionario en memoria.
    - Si se elige “Todos”, el backend itera sobre cada sitio disponible.
    - Si se especifica uno sólo, procesa únicamente ese.

3. Preparación del vector de entrada, para cada sitio o para el solicitado:
    1. Lectura de histórico: Se extraen las últimas filas de la tabla intermedia dataframe, ordenadas por fecha.
    2. Recálculo de lags: Sobre cada variable predictiva se generan columnas desplazadas (lag 1, 2, 3) para capturar la información de los meses anteriores.
    3. Construcción del vector
        - Se calculan las componentes temporales (mes, seno/coseno del mes, estación).
        - Para variables no-lag, se toma el último valor conocido.
        - Para sensores de precipitación, se utiliza la mediana histórica del mes si hay datos; en caso contrario, se asume cero.

    Esta función construye el vector de características para el siguiente paso de tiempo (t+1), replicando la misma lógica de creación de lags y features que en el entrenamiento.

4. Escalado y predicción, una vez obtenido el vector completo, utiliza el método .predict() del modelo en memoria para obtener la clase de alerta.:

    - Se aplica el StandardScaler para normalizar cada característica con la misma escala usada durante el entrenamiento.

    - Si el modelo es de Keras, se invoca model.predict(...) y se elige la clase con mayor probabilidad. Ya que en redes neuronales creadas con Keras, la llamada a model.predict(X) devuelve para cada muestra un vector de probabilidades que indica la confianza del modelo en cada una de las posibles clases (por ejemplo, [0.1, 0.7, 0.2]). Para convertir ese vector en una

etiqueta de clase concreta, es necesario elegir la posición de la probabilidad más alta.

- Si es un sklearn estimator, basta con llamar a model.predict(...). Por su parte, los estimadores de scikit-learn (RandomForestClassifier, LogisticRegression, etc.) ya internamente calculan la clase más probable cuando se invoca model.predict(X), de modo que devuelven directamente el valor de la etiqueta (por ejemplo, 0, 1 o 2), sin necesidad de aplicar un paso adicional de selección sobre probabilidades.

5. Mapeo de clases y respuesta al frontend: La salida es un entero (0, 1 o 2) que se traduce a una etiqueta legible (“Vigilancia”, “Alerta” o “Emergencia”). El backend devuelve un JSON que incluye:

    - El código de perfil (sitio)
    - Para cada target, la etiqueta de alerta calculada
    - Modelo elegido para predecir con sus métricas
    - Mensajes de error cuando no hay modelo disponible o los datos históricos son insuficientes

Con este diseño, el backend reacciona inmediatamente a cualquier cambio en los datos de calidad de agua, ejecuta de forma independiente el pipeline de preprocesamiento y mantiene la tabla dataframe siempre sincronizada. A su vez, los modelos se re-entrenan y recargan automáticamente en memoria, sin necesidad de reiniciar el servidor ni afectar la continuidad del servicio. Así, los endpoints y la interfaz de usuario disponen en todo momento de la información actualizada, sin sacrificar la capacidad de respuesta de la aplicación.

Entorno Virtual de Python

El modelo de machine learning en este proyecto se ejecuta dentro de un **entorno virtual de Python**, específicamente llamado *new_vevnv*. Este entorno virtual es una instalación aislada de Python que permite gestionar paquetes y dependencias sin afectar la instalación global del sistema.

Se creó para encapsular todas las dependencias necesarias para ejecutar el backend y el modelo de machine learning sin conflictos con otros proyectos.

Beneficios:

1. Aislamiento de dependencias: Permite instalar versiones específicas de paquetes como *Flask, pandas, scikit-learn, XGBoost y SQLAlchemy* sin afectar otros proyectos.
2. Reproducibilidad: Garantiza que el modelo pueda ejecutarse en otro entorno sin problemas, siempre que se instalen las mismas dependencias.
3. Gestión eficiente de paquetes: Facilita la actualización y mantenimiento de bibliotecas sin comprometer la estabilidad del sistema.
4. Compatibilidad con despliegue: Permite trasladar el entorno a servidores, Docker o máquinas de producción con facilidad.

Una vez activado, cualquier paquete instalado con *pip install* quedará dentro del entorno virtual sin afectar al sistema global. Y si el modelo se despliega en otro sistema, solo es necesario replicar *new_vevnv* instalando las mismas dependencias.

FRONTEND

El front-end de la plataforma ha sido diseñado para ser intuitivo y responsivo, de modo que los investigadores puedan explorar datos, generar predicciones y recibir alertas sin curva de aprendizaje. A continuación se detallan sus principales módulos y características:

Menú principal con acceso rápido a:

- Explorador de tabla de datos (dataframe): Tabla interactiva (react-table) que muestra los registros de calidad de agua e hidrometeorología, que se utilizan como entrada para el modelo (data frame), esto cuenta además con:
    - Paginación
    - Filtros personalizados (mes, año, estación y sitio)
- Historial de predicciones: listado cronológico de ejecuciones de predicción previas, con opción de volver a visualizar cualquier fecha, y explorar predicciones históricas. También cuenta con paginación y filtros (mes, año, sitio, etiqueta de alerta)
- Módulo de predicciones: donde se generará la predicción, contiene:
    - Un selector de sitio, dropdown para elegir uno o varios sitios, incluso la opción “Todos”.

- Resumen de predicción, cuadro con variables objetivo y la categoría de alerta (Vigilancia, Alerta, Emergencia). Contiene también, el modelo seleccionado como el mejor y sus métricas, a modo informativo para ayudar a la interpretación de las salidas de las predicciones.
- En caso de haber solicitado más de un sitio, ya sea por haber elegido la opción “Todos” o haber solicitado varios de a una vez, cada uno tiene una pestaña donde se detalla el resumen de predicción.
- Al cerrar la pestaña de algún sitio, se puede volver a solicitar la predicción del mismo.

- Bloqueo de Interfaz en proceso de Actualización
  - El frontend es el responsable de consultar periódicamente el estado del backend y reaccionar en consecuencia para bloquear o desbloquear la interfaz. Se ejecuta la función que consulta el estado de la app (entrenando o inactivo) cada 3 segundos, cuando lee el valor de respuesta inactivo, se desbloquea la interfaz, permitiendo al usuario operar en la misma.

- Visualización de Monitoreo del Modelo
  - Renderiza un gráfico de líneas dinámico. El eje X representa el tiempo (la fecha de cada reentrenamiento) y el eje Y representa el valor de las métricas. Se grafican simultáneamente las curvas de F1-Score, Precision Weighted y ROC AUC, por sitio y variable, para permitir una comparación directa.
  - Filtros Interactivos: Para facilitar el análisis, la interfaz incluye menús desplegables que permiten al usuario filtrar los datos visualizados por sitio de muestreo (C1, TAC1, etc.) y por variable objetivo (Clorofila, Cianobacterias, etc.). Al cambiar un filtro, el gráfico se actualiza en tiempo real para reflejar únicamente la evolución del modelo específico seleccionado.

Tecnologías y Herramientas

- React.js:
  - Hooks y Context API para gestión de estado global (sitio seleccionado, datos cargados).
  - React Router para navegación fluida entre vistas.
  - HTML, define la estructura y el contenido básico de la página web.

    - Javascript**,** proporciona interactividad y lógica al frontend. Permite realizar acciones dinámicas, en nuestro caso manejar eventos (clics), actualizar contenido en tiempo real sin recargar la página.

- Librerías de UI:
    - Material UI (o Bootstrap) para componentes prediseñados y responsividad móvil.
    - React-Table para tablas avanzadas.
- Comunicación con el API:
    - Axios para llamadas HTTP, con manejo de errores y timeouts.
    - Integración con WebSockets (en caso de futuras notificaciones en tiempo real).
- Estilos:
    - CSS modular para un diseño limpio, consistente y fácilmente escalable.

Desarrollo del Modelo Predictivo

El modelo predictivo fue diseñado para anticipar condiciones ambientales críticas en el Embalse San Roque, utilizando los datos disponibles hasta el mes t para **predecir la clase de alerta en el mes siguiente** $(t + 1)$. Esta aproximación permite generar **alertas tempranas** basadas en la evolución reciente del sistema.

La implementación sigue un enfoque específico por sitio de monitoreo, generando modelos independientes para cada combinación de sitio y variable objetivo. Este enfoque permite capturar patrones locales asociados a las condiciones particulares de cada zona del embalse.

La estrategia general se organizó en cuatro fases principales:

Fase 1: Ingeniería de Variables

Variables Objetivo:
Se definieron tres variables categóricas clave, transformadas en clases de alerta:

- **Clorofila (µg/l)**: *(0 = <10, 1 = 10–24, 2 = >24)*

- **Cianobacterias Totales (cel/mL)**: *(0 = <5.000, 1 = 5.000–60.000, 2 = >60.000)*
- **Dominancia de Cianobacterias (%)**: *(0 = <50, 1 = ≥50)*

Variables Predictoras Iniciales:

- Clorofila (µg/l)
- Cianobacterias (cel/mL)
- Total de Algas (cel/mL)
- Dominancia de Cianobacterias (%)
- Temperatura del agua (T° (°C))
- Fósforo hidrolizable total (PHT µg/l)
- Fósforo reactivo soluble (PRS µg/l)
- Nitrógeno inorgánico total (µg/l)
- Nivel del lago (Cota (m))
- Temperatura máxima del aire
- Temperatura mínima del aire
- Valores de sensores hidrometeorológicos 600, 700 y 1100
- Condición térmica *(variable categórica)*

Variables Temporales:

- Mes_sin y mes_cos: componentes trigonométricos del mes para capturar estacionalidad.

Variables Derivadas Temporales:

Con el objetivo de capturar patrones relevantes en la evolución de las condiciones ambientales, se generaron variables de retardo temporal (**lags**) correspondientes a los valores de 1, 2 y 3 meses previos a cada medición. Estas variables permiten al modelo incorporar información sobre el estado reciente del sistema.

Además, para las mismas variables predictoras, se calcularon medidas de tendencia y variabilidad de corto plazo mediante el **promedio** y el **desvío estándar** de los valores registrados en los 3 y 6 meses anteriores a cada observación. Estas estadísticas resumen cómo se comportó cada variable en los meses previos a la medición, permitiendo al modelo

detectar posibles tendencias ascendentes, descendentes o inestabilidad en los parámetros ambientales.

Limpieza post-transformaciones:
Las transformaciones introdujeron valores nulos en los primeros registros por sitio, los cuales fueron eliminados para garantizar consistencia antes del entrenamiento.

Fase 2: Entrenamiento y Optimización del Modelo

Selección de Variables Relevantes

Dado que el conjunto de datos incluye una gran cantidad de variables derivadas y transformadas, se aplicó un proceso de selección de características para reducir la dimensionalidad y mejorar la eficiencia del entrenamiento. Para ello, se utilizó el método ***SelectKBest*** con la función de puntuación mutual information (información mutua), que mide la dependencia estadística entre cada variable y la variable objetivo. Este análisis se realizó de forma independiente para cada combinación de sitio y variable objetivo, seleccionando las 25 variables más informativas en cada caso.

Modelos Evaluados

Se entrenaron y compararon diferentes algoritmos de clasificación:

- **Regresión Logística**: modelo lineal base, útil como referencia.
- **Random Forest**: conjunto de árboles de decisión que mejora la precisión al reducir el sobreajuste.
- **Red Neuronal Multicapa (MLP)**: modelo no lineal que puede capturar relaciones complejas entre las variables.

Optimización de Hiperparámetros

Para garantizar el mejor rendimiento posible, se realizó una búsqueda de hiperparámetros para cada modelo:

Modelos Scikit-learn (LogisticRegression, RandomForest):

Se utilizó **GridSearchCV**, una búsqueda exhaustiva dentro de un espacio definido de hiperparámetros. La evaluación se realizó mediante validación cruzada temporal (*TimeSeriesSplit*), que respeta el orden cronológico de los datos, entrenando sobre periodos pasados y validando en periodos futuros. Como métrica principal se utilizó el **F1-score**

macro, que equilibra el rendimiento entre todas las clases, especialmente útil en escenarios con clases desbalanceadas.

Redes Neuronales (MLP):

Se empleó **Keras Tuner**, que permite buscar automáticamente la mejor combinación de hiperparámetros como la cantidad de neuronas, tasa de aprendizaje y dropout. Esta búsqueda se realiza sobre un conjunto de validación separado, con uso de early stopping para evitar el sobreajuste. El criterio de selección fue la **accuracy** en validación.

La descripción detallada de la arquitectura y de los hiperparámetros considerados en cada caso se encuentra disponible en el **Anexo II: Arquitectura e Hiperparámetros de los Modelos.**

Escalado de Datos

Previo al entrenamiento de cada modelo, todas las variables predictoras fueron estandarizadas utilizando **StandardScaler**. Este paso transforma las variables numéricas para que tengan media cero y desviación estándar uno, mejorando la estabilidad y el rendimiento de los algoritmos que son sensibles a la escala de los datos.

Fase 3: Evaluación del Desempeño del Modelo

Una vez optimizados los modelos para cada combinación de sitio y variable objetivo, se procedió a su evaluación utilizando validación cruzada temporal (TimeSeriesSplit) con tres divisiones. Este tipo de validación garantiza que el modelo se entrena en períodos anteriores y se evalúa en períodos futuros, respetando la naturaleza secuencial del problema.

Métricas de Evaluación

Para cuantificar el desempeño del modelo se utilizaron dos métricas principales: **F1-score macro y AUC-ROC macro.**

F1-score macro: Es la media del F1-score calculado por clase, sin ponderar por la frecuencia de cada clase. El F1-score en sí mismo es la media armónica entre precisión y sensibilidad (recall), y se calcula como:

$$\text{Precision} = \frac{TP}{TP + FP}, \quad \text{Recall} = \frac{TP}{TP + FN}$$

$$F1 = 2 \cdot \frac{\text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}}$$

Donde:

- **TP**: Verdaderos positivos.
- **FP**: Falsos positivos.
- **FN**: Falsos negativos.

Un valor de F1-macro cercano a 1 indica un modelo con alto rendimiento en todas las clases.

AUC-ROC macro: El Área Bajo la Curva ROC (Receiver Operating Characteristic) mide la capacidad del modelo para distinguir entre clases. La curva ROC se construye graficando la tasa de verdaderos positivos (TPR) frente a la tasa de falsos positivos (FPR) a medida que varía el umbral de clasificación.

En problemas multiclase, se utiliza una estrategia "uno contra el resto" (OvR), en la que se calcula un AUC-ROC individual para cada clase, tratando cada una como la clase positiva y agrupando las demás como negativas. El valor **macro** del AUC-ROC se obtiene promediando aritméticamente estos valores individuales, sin tener en cuenta la frecuencia relativa de cada clase en el conjunto de datos. Este enfoque da igual peso a todas las clases, independientemente de su representación, lo que lo hace útil para evaluar el desempeño general del modelo en contextos con **clases desbalanceadas.**

Valores cercanos a 1 indican una alta capacidad de discriminación del modelo, mientras que valores cercanos a 0.5 reflejan un desempeño equivalente al azar.

Criterio de Selección del Mejor Modelo

Para cada combinación de sitio y variable objetivo, se aplicó un criterio jerárquico de selección basado en umbrales mínimos de desempeño:

1. **Nivel 1 (Óptimo):** Se priorizaron los modelos con AUC-ROC ≥ 0.60. Entre ellos, se seleccionó el que presentó el mayor F1-score macro.
2. **Nivel 2 (Aceptable):** Si ningún modelo alcanzó el umbral anterior, se consideraron aquellos con AUC-ROC ≥ 0.50, eligiendo nuevamente el de mejor F1-score macro.

3. **Nivel 3 (Exploratorio):** En ausencia de modelos con AUC-ROC ≥ 0.50, se seleccionó el modelo con mayor F1-score macro, aunque se señaló que su uso debe realizarse con máxima cautela.

Este enfoque busca garantizar un balance entre **capacidad de discriminación global (AUC-ROC)** y **equilibrio en la clasificación de todas las clases (F1-score macro)**.

Resultados del Entrenamiento y Evaluación de Modelos

Los resultados obtenidos variaron según el sitio de monitoreo y la variable objetivo. En general, la dominancia de cianobacterias fue la variable con mejor desempeño, seguida por la predicción de cianobacterias totales, mientras que la clorofila presentó los mayores desafíos. Esto evidencia que la efectividad del modelo depende tanto del tipo de variable como de las características particulares de cada sitio.

Desempeño por Variable Objetivo

Dominancia de Cianobacterias (clasificación binaria):

Esta variable mostró el mejor desempeño global. En múltiples sitios, los modelos lograron capturar con eficacia las condiciones que determinan la dominancia de cianobacterias en el total de algas.

- F1-score macro: entre 0.61 y 0.69, destacándose el sitio TAC1 (0.69).
- AUC-ROC: entre 0.63 y 0.70, lo que indica buena capacidad de discriminación entre situaciones dominantes y no dominantes.
- Variables clave: temperatura del agua, cianobacterias totales, clorofila y fósforo hidrolizable total.

Cianobacterias Totales (clasificación multiclase):

Esta variable también obtuvo resultados positivos, especialmente en los extremos de la distribución, aunque con mayores desafíos para la clase intermedia, donde el solapamiento entre condiciones fue más común.

- F1-score macro: entre 0.35 y 0.49, destacándose TAC4 (0.49).
- AUC-ROC: entre 0.62 y 0.69, confirmando una buena sensibilidad del modelo a cambios en la abundancia de cianobacterias.
- Variables clave: clorofila, temperatura del agua, nitrógeno inorgánico.

Clorofila (clasificación multiclase):

Fue la variable con mayor variabilidad y menor desempeño predictivo general. Esto se atribuye a una mayor complejidad del fenómeno.

- F1-score macro: entre 0.34 y 0.42, con sitios como DSA1 (0.39) y TAC1 (0.40) destacándose levemente.
- AUC-ROC: entre 0.55 y 0.63, reflejando una capacidad moderada de clasificación.
- Variables clave: dominancia de cianobacterias, nitrógeno inorgánico total, temperatura máxima, fósforo total y valores históricos (lags, promedios y desviaciones estándar de los valores pasados de clorofila, cianobacterias o nutrientes).

Fase 4: Generación de Predicciones

Una vez seleccionado el mejor modelo para cada caso, se procedió a la generación de predicciones para el mes siguiente $(t + 1)$. Para ello, se construyó un **vector de características.** Este vector incluyó los valores de las variables predictoras, temporales, lags y las estadísticas móviles del mes t que representan la dinámica reciente del sistema.

Antes de ser utilizado por el modelo, el vector fue **escalado** con el mismo StandardScaler aplicado durante el entrenamiento. Finalmente, el modelo correspondiente produjo una predicción de la **clase de alerta esperada** para el próximo período para cada variable, permitiendo anticipar condiciones críticas en cada sitio de monitoreo.

PRUEBAS

El objetivo de las pruebas es verificar que cada componente cumpla con los requisitos funcionales y no funcionales establecidos, detectando defectos antes de su despliegue en producción. El proceso de pruebas busca garantizar la calidad, fiabilidad y mantenimiento del sistema.

Backend

En el desarrollo del backend, se implementó una estrategia de pruebas exhaustiva para garantizar la estabilidad y mantenibilidad del código. Se utilizaron los frameworks pytest y pytest-mock para la creación y ejecución de las pruebas, las cuales se dividen en dos categorías principales: Pruebas Unitarias y Pruebas de Integración.

Tabla 1: Casos de Prueba - Pruebas Unitarias
Estas pruebas validan componentes de código individuales en aislamiento.

ID de Prueba	Descripción	Objetivo de la Prueba	Entrada	Pasos	Resultado Esperado
PB-U-001	Prueba unitaria para la función asignar_estacion.	Asignación exitosa de estación.	Un número entero representando el mes (e.g., 1, 4, 7, 11).	1. Se invoca la función asignar_estacion con el número del mes.	La función devuelve el string de la estación correcta ('Verano', 'Otoño', 'Invierno', 'Primavera').
PB-U-002	Prueba unitaria para la imputación de la columna Cota (m).	Imputación correcta de valores nulos en la cota.	Un DataFrame de pandas con la columna Cota (m) conteniendo valores NaN.	1. Se prepara un DataFrame de entrada con cotas nulas. 2. Se invoca la función imputar_cota_m.	El DataFrame de salida no tiene valores nulos en Cota (m). El valor NaN es reemplazado por el último valor válido de la misma fecha.
PB-U-003	Prueba unitaria para la selección de la medición mensual más representativa.	Selección determinista de la mejor medición del mes.	Un DataFrame con múltiples mediciones para un mismo sitio en el mismo mes, algunas más incompletas que otras.	1. Se prepara un DataFrame de entrada con mediciones duplicadas. 2. Se invoca la función seleccionar_medicion_mensual.	El DataFrame de salida contiene una única fila por sitio/mes, correspondiente a la medición que originalmente tenía menos valores nulos.
PB-U-004	Prueba unitaria para la imputación de temperatura del agua.	Imputación correcta de T° del agua por media de grupo.	Un DataFrame con valores nulos en la columna T° (°C).	1. Se prepara un DataFrame de entrada. 2. Se invoca la función imputacion_temperatura_agua.	El valor nulo de T° (°C) es reemplazado por la media de los otros valores del mismo sitio y estación.
PB-U-005	Prueba unitaria para la imputación de temperatura del aire.	Imputación correcta de T° del aire por media de grupo.	Un DataFrame con valores nulos en temperatura_max y temperatura_	1. Se prepara un DataFrame de entrada. 2. Se invoca la función imputacion_te	Los valores nulos son reemplazados por la media de su respectivo grupo (mes y estación).

			min.	mperatura_aire.	
PB-U-006	Prueba unitaria para la lógica condicional de imputación de clorofila.	Selección robusta del método de imputación de clorofila.	Un DataFrame con un valor atípico en Clorofila (µg/l) y un valor nulo a imputar.	1. Se prepara un DataFrame que haga que el método de la mediana sea más preciso que un modelo. 2. Se invoca imputacion_clorofila.	La función elige 'mediana' como el método de imputación y el valor nulo se rellena con la mediana del grupo, no con la predicción del modelo.
PB-U-007	Prueba unitaria para la imputación de fósforo (PHT y PRS).	Imputación exitosa de PHT y PRS usando modelos predictivos.	Un DataFrame con valores nulos en las columnas PHT (µg/l) y PRS (µg/l).	1. Se prepara un DataFrame de entrada. 2. Se invocan las funciones imputar_pht e imputar_prs.	Las columnas PHT (µg/l) y PRS (µg/l) en el DataFrame de salida no contienen valores nulos.

Tabla 2: Casos de Prueba - Pruebas de Integración

Estas pruebas validan la interacción entre diferentes componentes del sistema.

ID de Prueba	**Descripción**	**Objetivo de la Prueba**	**Entrada**	**Pasos**	**Resultado Esperado**
PB-I-001	Prueba de integración para la unión de datos externos (precipitación y temperatura).	Correcta fusión de datos hidrometeorológicos en el DataFrame principal.	Un DataFrame principal y una simulación (mock) de los DataFrames de precipitación y temperatura leídos desde la BD.	1. Se simula la respuesta de pandas.read_sql para que devuelva datos predefinidos. 2. Se invocan las funciones union_precipitacion y union_temperatura_aire.	El DataFrame de salida se enriquece con las nuevas columnas (temperatura_max, temperatura_min, y columnas por sensor de precipitación) con los valores fusionados correctamente por fecha.
PB-I-002	Prueba de integración para el cálculo de la condición	Correcta integración de la lógica de condición	Un DataFrame principal y una simulación	1. Se simula la respuesta de pandas.read_sql para que devuelva datos	El DataFrame principal de salida tiene la columna condicion_termic

	térmica.	térmica con datos de la BD.	(mock) de la vista vista_condicion_termica.	que indiquen una condición 'ESTRATIFICADA'. 2. Se invoca la función condicion_termica.	a actualizada a 'ESTRATIFICADA'.
PB-I-003	Prueba de integración del pipeline completo de creación del DataFrame.	Verificar la orquestación de todo el pipeline de datos.	Mocks para todas las lecturas de pandas.read_sql que la función obtener_dataframe realiza.	1. Se configuran mocks para devolver DataFrames simulados para cada vista de la BD. 2. Se invoca la función obtener_dataframe().	La función se ejecuta sin errores y devuelve un único DataFrame final, no vacío, que contiene todas las columnas generadas y transformadas por los pasos intermedios.
PB-I-004	Prueba de integración del listener de la BD con el trigger de actualización.	Verificar que una notificación NOTIFY de la BD dispara el pipeline de actualización.	Una notificación en el canal datos_agua_actualizados emitida en la base de datos de prueba.	1. Se inicia el database_listener en un hilo. 2. Se emite una notificación NOTIFY desde una conexión a la BD de prueba. 3. Se verifica (usando un mock) si la función actualizar_df fue llamada.	La función actualizar_df es invocada exactamente una vez, confirmando que el mecanismo de escucha y respuesta funciona de extremo a extremo.
PB-I-005	Prueba de integración del pipeline de actualización con el script de reentrenamiento.	Verificar que la actualización de datos dispara el reentrenamiento del modelo.	Una llamada a la función actualizar_df.	1. Se simulan las funciones obtener_dataframe, to_sql y recargar_modelos. 2. Se simula la llamada subprocess.run. 3. Se invoca actualizar_df().	La función subprocess.run es invocada una vez con los argumentos correctos para ejecutar el script entrenar_modelos.py.
PB-I-006	Prueba de	Verificar	Una	1. Se simula la	La API responde

	integración para el endpoint /get-options.	que la API devuelve la lista de sitios de muestreo.	petición HTTP GET al endpoint /get-options .	lectura de la BD (pd.read_sql). 2. Se realiza la petición GET usando el cliente de prueba.	con un código de estado 200 y un JSON que contiene una lista de los codigo_perfil únicos y sin nulos.
PB-I-007	Prueba de integración para el endpoint /predict.	Verificar que la API procesa una solicitud de predicción exitosa.	Una petición HTTP POST a /predict con el JSON {'option': 'C1'}.	1. Se simula la función hacer_prediccion_para_sitio. 2. Se realiza la petición POST con el cliente de prueba.	La API responde con un código de estado 200 y un JSON que contiene los resultados de la predicción simulada.
PB-I-008	Prueba de integración para el manejo de errores del endpoint /predict.	Verificar que la API maneja correctamente las peticiones inválidas.	Una petición HTTP POST a /predict con un JSON vacío {}.	1. Se realiza la petición POST inválida con el cliente de prueba.	La API responde con un código de estado 400 (Bad Request) y un JSON que contiene un mensaje de error claro.

Tabla 3: Casos de Prueba - Conexión a Base de Datos

Esta prueba es fundamental para validar la correcta configuración del entorno de desarrollo y pruebas, asegurando que el backend puede interactuar con la base de datos gestionada por Docker.

ID de Prueba	**Descripción**	**Objetivo de la Prueba**	**Entrada**	**Pasos**	**Resultado Esperado**
PB-I-009	Prueba de conexión e interacción con la base de datos de prueba.	Verificar la conectividad con la base de datos PostgreSQL y la capacidad de ejecutar un ciclo completo de operaciones CRUD (Crear, Leer,	La URL de conexión a la base de datos de prueba levantada con Docker.	1. Se establece una conexión con la BD. 2. Se crea una tabla de prueba (CREATE TABLE). 3. Se inserta un registro en la tabla (INSERT). 4. Se leen los datos insertados (SELECT). 5. Se elimina la tabla de prueba para limpiar el	La conexión es exitosa. Todas las operaciones SQL se ejecutan sin errores y los datos leídos coinciden con los datos insertados. Si la conexión falla, la prueba falla con un mensaje claro que indica que el contenedor Docker podría no estar en

		Eliminar).		estado (DROP TABLE).	ejecución.

Tabla 4: Casos de Prueba - Pipeline de Entrenamiento de Modelos

Estas pruebas se centran en la lógica del script entrenar_modelos.py, asegurando que tanto la preparación de los datos como la generación de las variables objetivo (target) se realicen correctamente antes del entrenamiento.

ID de Prueba	Descripción	Objetivo de la Prueba	Entrada	Pasos	Resultado Esperado
PE-U-001	Prueba unitaria para la clasificación de alerta de Clorofila.	Verificar que la función classify_chlorophyll_alerta asigna la clase correcta (0, 1, 2) según los umbrales definidos.	Valores numéricos que representan la concentración de clorofila, incluyendo valores en los límites de los umbrales y NaN.	1. Se invoca la función con valores por debajo del primer umbral, entre umbrales y por encima del segundo umbral.	La función devuelve el entero de clase correcto para cada valor: 0 para 'Vigilancia', 1 para 'Alerta', y 2 para 'Emergencia'. Devuelve NaN si la entrada es nula.
PE-U-002	Prueba unitaria para la clasificación de alerta de Cianobacterias.	Verificar que classify_cyanobacteria_alerta asigna la clase correcta, manejando la conversión de unidades (cel/L a cel/mL).	Valores numéricos que representan la concentración de cianobacterias (en cel/L), incluyendo límites y NaN.	1. Se invoca la función con valores que, tras la conversión de unidades, se sitúen en las diferentes categorías de alerta.	La función devuelve el entero de clase correcto (0, 1, 2) y se confirma que la conversión de unidades es implícita en la lógica. Devuelve NaN para entradas nulas.
PE-U-003	Prueba unitaria para la clasificación de alerta de Dominancia.	Verificar que classify_dominance_alerta asigna la clase correcta (0 o 1) según el umbral de dominancia (>50%).	Valores numéricos que representan el porcentaje de dominancia, incluyendo el límite del 50% y	1. Se invoca la función con valores por debajo, en el límite y por encima del umbral.	La función devuelve 0 para 'No Dominante' y 1 para 'Dominante'. Devuelve NaN para entradas nulas.

			NaN.		
PE-I-001	Prueba de integración para la ingeniería de características (lags y ventanas móviles).	Verificar que la función preprocess_and_feature_engineer crea correctamente las columnas de retardo (lag) y de media móvil (rolling mean).	Un DataFrame de pandas con una serie temporal de datos (fecha y al menos una variable numérica).	1. Se prepara un DataFrame de entrada simple con una secuencia temporal. 2. Se invoca la función preprocess_and_feature_engineer.	El DataFrame de salida contiene las nuevas columnas de ingeniería de características (e.g., T° (°C)lag1, T° (°C)roll_mean3) y los valores calculados en estas columnas son correctos según la lógica de pandas.

Frontend

Para validar la interfaz de usuario y su interacción con el backend, se desarrollaron pruebas utilizando la librería React Testing Library y Jest. Estas pruebas garantizan que los componentes se rendericen correctamente, que el usuario pueda interactuar con ellos de la manera esperada y que la comunicación con la API sea la correcta.

Tabla 5: Casos de Prueba - Frontend (Unitarias)

Estas pruebas validan funciones de utilidad aisladas que no dependen del renderizado de componentes.

ID de Prueba	Descripción	Objetivo de la Prueba	Entrada	Pasos	Resultado Esperado
PF-U-001	Prueba unitaria para la función de utilidad formatDate.	Verificar el correcto formateo de fechas de 'YYYY-MM-DD' a 'DD/MM/YYYY'.	Un string de fecha en formato 'YYYY-MM-DD' (e.g., '2025-07-28').	1. Se invoca la función formatDate con el string de fecha.	La función devuelve la fecha en formato 'DD/MM/YYYY' (e.g., '28/07/2025').
PF-U-002	Prueba de manejo de errores para formatDate.	Verificar el manejo de formatos de fecha inválidos.	Un string de fecha en un formato incorrecto (e.g., '28-07-2025').	1. Se invoca la función formatDate con el string inválido.	La función devuelve el string 'Fecha inválida'.
PF-U-003	Prueba de	Verificar el	null o	1. Se invoca	La función

	manejo de nulos para formatDate.	comportamiento de la función ante entradas nulas o indefinidas.	undefined.	la función formatDate con un valor nulo.	devuelve un guion ('-').

Tabla 6: Casos de Prueba - Frontend (Integración de Componentes)

Esta prueba válida el flujo completo de una de las funcionalidades principales de la aplicación, desde la carga de datos hasta la interacción del usuario y la visualización del resultado.

ID de Prueba	Descripción	Objetivo de la Prueba	Entrada	Pasos	Resultado Esperado
PF-I-001	Prueba de integración del flujo de predicción en el componente Predict.	Verificar el flujo completo: 1. Carga de opciones desde la API. 2. Selección de una opción por el usuario. 3. Envío de la petición de predicción. 4. Renderizado de los resultados.	Interacción simulada del usuario. Datos simulados (mocks): - Respuesta de axios.get para /api/get-options. Respuesta de axios.post para /api/predict.	1. Se simulan las respuestas de la API para las peticiones GET y POST. 2. Se renderiza el componente Predict. 3. (Simulación de usuario) Se selecciona una opción del menú desplegable. 4. (Simulación de usuario) Se hace clic en el botón "Predecir".	1. El componente se carga y muestra correctamente el menú desplegable poblado con las opciones simuladas. 2. Tras hacer clic, la sección de resultados se muestra en pantalla. 3. Los resultados (nivel de alerta, modelo usado, métricas) coinciden con los datos de la respuesta POST simulada. 4. Se verifica que la llamada a axios.post se realizó con la opción seleccionada por el usuario.
PF-I-002	Prueba	Verificar	Respuesta	1. Se simula que la	El componente

	de integración para el overlay de reentrenamiento.	que el componente App muestre y oculte correctamente una pantalla de bloqueo (overlay) en respuesta al estado (retraining/idle) informado por el backend.	simulada del endpoint /api/status, alternando entre {'status': 'retraining'} y {'status': 'idle'}.	API /api/status responde con 'idle'. 2. Se renderiza el componente App. 3. Se verifica que el overlay no es visible. 4. Se simula que la API ahora responde con 'retraining'. 5. Se avanza el tiempo del test para disparar el sondeo. 6. Se verifica que el overlay es visible. 7. Se vuelve a simular la respuesta 'idle'. 8. Se avanza el tiempo del test. 9. Se verifica que el overlay ya no es visible.	muestra el overlay de "Espere..." únicamente cuando la respuesta de la API es {'status': 'retraining'}. Cuando la respuesta es {'status': 'idle'}, el overlay no se muestra en la interfaz.

Pruebas End-to-End (E2E)

Además de las pruebas unitarias y de integración, se implementó una estrategia de pruebas de extremo a extremo (End-to-End) utilizando el framework Cypress. La superación de esta prueba asegura que la arquitectura del frontend (enrutamiento, estado de componentes), la interacción entre componentes y la lógica de visualización de datos funcionan de manera integrada. Proporciona una alta confianza en que un usuario real puede completar las tareas de la aplicación sin encontrar errores funcionales.

Caso de Prueba: Flujo Principal de la Aplicación

- **ID del Caso de Prueba:** PF-E2E-001
- **Descripción:** Esta prueba simula el recorrido de un usuario a través de las funcionalidades de la aplicación: navegar a la página de predicción, ejecutar una predicción, ver los resultados y, finalmente, navegar a las otras secciones principales ("Datos", "Historial de Predicciones" y “Acerca del Modelo”).

Estrategia de Simulación de API

Para garantizar que esta prueba sea rápida, determinista e independiente del estado del backend, se utiliza la funcionalidad cy.intercept() de Cypress. Esta estrategia consiste en "interceptar" las llamadas a la API que realiza el frontend y devolver respuestas predefinidas y controladas. En este caso:

1. Interceptación de Opciones (getOptionsRequest): Se intercepta la llamada GET /api/get-options. En lugar de depender de lo que haya en la base de datos, la prueba fuerza que la respuesta sea siempre la lista ['C1', 'TAC1', 'TAC4', 'DCQ1', 'DSA1'].
2. Interceptación de Predicción (predictRequest): Se intercepta la llamada POST /api/predict. La prueba devuelve un objeto de predicción fijo, asegurando que el resultado a verificar en la interfaz sea siempre el mismo (Emergencia para C1).

Flujo de Pasos y Verificaciones Ejecutadas

La prueba ejecuta la siguiente secuencia de acciones y comprobaciones:

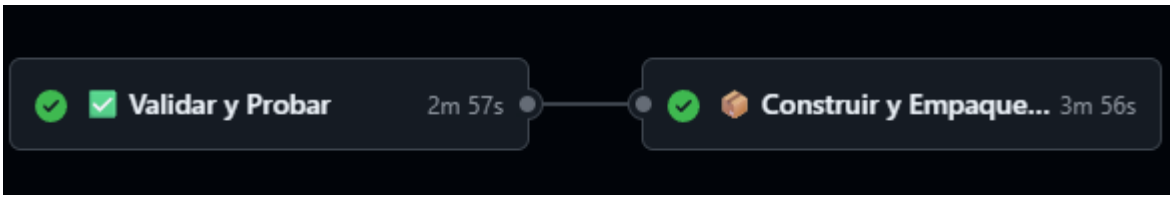
1. Visita y Navegación Inicial:
    a. Se visita la página de inicio (http://localhost:3000).
    b. Se busca y hace clic en el botón "Predecir".
    c. Se comprueba que la URL del navegador haya cambiado a /predict, confirmando que la navegación funcionó.
2. Ejecución de una Predicción:
    a. La prueba espera explícitamente (cy.wait) a que la llamada para obtener las opciones (getOptionsRequest) se complete. Esto es crucial para evitar errores por interactuar con elementos que aún no se han cargado.
    b. Se selecciona la opción 'C1' del menú desplegable.
    c. Se hace clic en el botón "Predecir" del formulario.
    d. Se espera (cy.wait) a que la llamada de predicción (predictRequest) finalice.
3. Validación de Resultados:
    a. Se comprueba que el título "Resultados para C1" sea visible en la página.
    b. Se confirma que la etiqueta de predicción "Emergencia" y el modelo "Cypress-Model" (definidos en la simulación) se muestran correctamente.
4. Verificación de Navegación a Otras Secciones:
    a. Se navega a la sección de "Datos".
    b. Verificación: Se comprueba que la URL incluya /datos y que el título "Tabla de Datos Procesado" esté presente, así como la tabla de datos.
    c. Se navega a la sección de "Historial de Predicciones".

d. Verificación: Se comprueba que la URL incluya /predicciones y que el título "Historial de predicciones" esté visible junto con los datos de la tabla.

DESPLIEGUE

El sistema se desplegó mediante contenedores Docker, lo que permitió encapsular de forma aislada y estandarizada los componentes del frontend y backend, garantizando portabilidad y facilitando su instalación, en el servidor del Instituto Nacional del Agua (INA). El frontend, desarrollado en React, se construye en una imagen de Node.js que compila la aplicación y se sirve mediante un contenedor Nginx, mientras que el backend, desarrollado en Python con el framework Flask, se ejecuta en una imagen ligera que incluye todas las dependencias necesarias para el procesamiento de datos y la ejecución del modelo de predicción. Ambas imágenes se orquestan utilizando Docker Compose, que permite una ejecución coordinada del sistema y define la exposición de los servicios a través de los puertos correspondientes.

Adicionalmente, se implementó una pipeline de integración y entrega continua (CI/CD) utilizando GitHub Actions, que automatiza todo el flujo de desarrollo con cada cambio que se realice en el repositorio: desde la validación de las pruebas, hasta la construcción y empaquetado de las imágenes Docker. Esta pipeline se compone de dos etapas principales: la primera, denominada “Validar y Probar”, ejecuta pruebas unitarias y de integración, tanto en el backend como en el frontend, asegurando la integridad del sistema; la segunda, “Construir y Empaquetar Imágenes”, genera los contenedores listos para ser desplegados en producción y los almacena como artefactos. Esto permite que, ante cada modificación del repositorio, se garantice un despliegue confiable, reproducible y controlado, reduciendo errores manuales y acelerando los tiempos de entrega.



Una vez construidas, las imágenes son transferidas e instaladas en el servidor del INA, donde el sistema se ejecuta de forma continua. Las computadoras conectadas a la red interna del organismo pueden acceder a la aplicación a través del navegador (192.168.191.164:3000), lo que permite a los usuarios visualizar las predicciones y alertas directamente desde su puesto de trabajo sin necesidad de instalar software adicional.


192.168.191.164:3000
INA
Instituto Nacional del Agua
MODELO DE PREDICCIÓN - INA CIRSA
Predecir
Ver Datos
Ver Predicciones
Acerca del Modelo
Monitorear Modelos

BENEFICIOS POST-IMPLEMENTACIÓN

La implementación del modelo predictivo basado en técnicas de machine learning traerá una serie de beneficios significativos al Instituto Nacional del Agua (INA) y al monitoreo del Embalse San Roque. Estos beneficios, tanto tangibles como intangibles, se enumeran a continuación:

- **Reducción de la carga operativa**: La automatización del análisis de datos y la generación de predicciones disminuirá significativamente el trabajo manual de los profesionales del INA, permitiéndoles enfocarse en la interpretación de los resultados y en la planificación de acciones correctivas.

- **Incremento en la precisión y rapidez de las decisiones**: El sistema predictivo mejorará la detección temprana de eventos críticos, como floraciones algales nocivas, lo que permitirá implementar medidas de mitigación con mayor anticipación y efectividad.

- **Mejora en la gestión ambiental**: Al anticipar condiciones favorables para la proliferación de cianobacterias, el modelo contribuirá a una gestión más eficiente del recurso hídrico, promoviendo la conservación del ecosistema acuático y reduciendo el impacto ambiental.

- **Optimización del uso de recursos**: Al evitar decisiones basadas en interpretaciones tardías o erróneas, se disminuirán los costos asociados a medidas reactivas y se facilitará la planificación estratégica de operativos de monitoreo.

- **Mayor disponibilidad y accesibilidad a la información**: A través de la interfaz gráfica desarrollada, los usuarios podrán visualizar los resultados de manera clara e intuitiva, promoviendo una mayor comprensión y difusión de la situación del embalse.

IMPACTO ECONÓMICO

Si bien este proyecto se enmarca en el ámbito académico y no implicó una inversión económica directa para su desarrollo, el análisis de impacto económico permite proyectar el valor potencial que una implementación real podría generar para el Instituto Nacional del Agua (INA) y la gestión ambiental del Embalse San Roque.

El principal aporte económico del modelo propuesto radica en su capacidad para **optimizar procesos existentes**, actualmente manuales y demandantes en recursos humanos. La automatización del análisis de datos y la generación de predicciones permitiría una reducción significativa en los tiempos dedicados por técnicos y profesionales al procesamiento e interpretación de la información, lo que se traduce en una mejora en la eficiencia operativa.

Además, la **detección temprana de eventos críticos**, como floraciones masivas de cianobacterias, contribuiría a evitar costos asociados a medidas reactivas, como tratamientos de emergencia del agua, cierres preventivos de tomas o impactos negativos sobre el turismo y la salud pública.

En resumen, aunque esta tesis no contempló costos económicos reales en su desarrollo, el sistema propuesto demuestra un **alto potencial para reducir costos operativos, prevenir gastos imprevistos y mejorar la asignación de recursos** en el contexto de un posible escenario de implementación institucional.

IMPACTO SOCIAL

El desarrollo de un modelo predictivo para anticipar floraciones de cianobacterias en el Embalse San Roque tiene un impacto social significativo, al contribuir a la preservación de un recurso hídrico clave para miles de personas.

Beneficio General y Bienestar Social

El embalse San Roque es una fuente esencial de agua para consumo humano, actividades recreativas y el turismo regional. Mejorar la calidad del monitoreo y la capacidad de respuesta ante episodios de contaminación contribuye directamente a proteger la salud pública, asegurar el acceso a agua segura y mantener la calidad ambiental del entorno. De esta manera, se promueve un mayor bienestar social y una mejor calidad de vida para los habitantes de la región.

Segmentos de la Población Beneficiados

La población beneficiada incluye tanto a los usuarios del agua potable, como a los sectores vinculados al turismo y recreación, que dependen de la buena calidad del agua del embalse. También se ven favorecidos los trabajadores del INA y otras instituciones públicas, al contar con herramientas más eficientes que fortalecen su labor.

Reducción de Brechas Tecnológicas

Este trabajo también promueve la incorporación de tecnologías avanzadas en instituciones públicas, contribuyendo a reducir brechas tecnológicas y fomentando una cultura de innovación orientada al bien común.

IMPACTO MEDIOAMBIENTAL

El desarrollo del modelo predictivo para el monitoreo del Embalse San Roque también representa un aporte relevante desde la perspectiva medioambiental. Su aplicación efectiva permite anticipar fenómenos críticos, como las floraciones de cianobacterias, que comprometen gravemente la calidad del agua y el equilibrio ecológico del embalse.

Mediante la automatización del análisis y la mejora en la detección temprana de eventos, el sistema facilita una intervención oportuna que puede reducir la necesidad de tratamientos intensivos, contribuyendo así a una menor generación de residuos y un uso más racional de los recursos.

Además, la implementación de esta herramienta puede fortalecer una visión institucional orientada a la sostenibilidad, promoviendo prácticas de gestión más responsables y sustentadas en datos.

CONCLUSIÓN

El desarrollo de este modelo predictivo para el análisis del Embalse San Roque representó una experiencia enriquecedora tanto en el plano técnico como en el personal. A lo largo del proyecto, se profundizó en el estudio de técnicas de machine learning aplicadas al monitoreo ambiental, enfrentando desafíos reales vinculados a la calidad del agua, el tratamiento de datos y la integración de múltiples fuentes de información.

Uno de los mayores aprendizajes fue la importancia de una adecuada preparación de los datos, desde su limpieza hasta la imputación de valores faltantes, y cómo estos procesos impactan directamente en la calidad de las predicciones. También se valoró el rol fundamental que tiene la visualización de resultados para facilitar la toma de decisiones por parte de los profesionales involucrados.

Los objetivos principales del proyecto fueron alcanzados: se diseñó e implementó un modelo capaz de predecir variables clave relacionadas con las floraciones de cianobacterias, se evaluaron múltiples enfoques predictivos y se construyó una interfaz intuitiva que permite explorar los resultados de manera accesible. Si bien quedan desafíos por resolver, el trabajo realizado constituye una base sólida para futuras implementaciones y mejoras.

Finalmente, este proyecto permitió aplicar conocimientos adquiridos durante la carrera en un contexto real, vinculando ciencia, tecnología y gestión ambiental.

REFERENCIAS

Andreoni, P. F. (2020). *Incidencia de condiciones hidrometeorológicas sobre calidad del agua del Embalse San Roque, mediante aplicación de minería de datos con series temporales* (Tesis de maestría). Universidad de Buenos Aires. https://datamining.dc.uba.ar/datamining/Files/Tesis/Tesis_Pablo_Andreoni.pdf

Barafani, F., & Dubowez, J. C. (2023). *Solución tecnológica para el aprovechamiento de datos del Instituto Nacional del Agua* (Tesis de grado). Universidad Católica de Córdoba. https://pa.bibdigital.ucc.edu.ar/4174/1/A_Barafani_Dubowez.pdf

Catalini, C. G., Rico, A. F., Dasso, C. M., Capone, M. E., & Mortarino, N. (2016). *Sistema de gestión de alertas en el INA–SCIRSA.* Instituto Nacional del Agua. https://www.ina.gob.ar/ifrh-2016/trabajos/IFRH_2016_paper_82.pdf

Fournier, C., Fernandez-Fernandez, R., Cirés, S., López-Orozco, J. A., Besada-Portas, E., & Quesada, A. (2024). *LSTM networks provide efficient cyanobacterial blooms forecasting even with incomplete spatio-temporal data.* https://doi.org/10.1016/j.watres.2024.122553

Hwang, S.-Y., Choi, B.-W., Park, J.-H., Shin, D.-S., Chung, H.-S., Son, M.-S., Lim, C.-H., Chae, H.-M., Ha, D.-W., & Jung, K.-Y. (2023). *Evaluating statistical machine learning algorithms for classifying dominant algae in Juam Lake and Tamjin Lake, Republic of Korea.* https://doi.org/10.3390/w15091738

Pussetto, N., Ruiz, M., Ruibal Conti, A. L., Rodríguez, M. I., & Dasso, C. (2018). *Caracterización de la calidad de agua y de variables meteorológicas relacionadas con eventos extremos de floración en el Embalse San Roque*. Instituto Nacional del Agua. https://www.ina.gov.ar/archivos/publicaciones/2018_Pussetto%20et%20al_Caracterizaci%C3%B3n%20Calidad%20Agua%20Eventos%20Extremos.pdf

Roldán Pérez, G., & Ramírez Restrepo, J. J. (2008). Fundamentos de Limnología Neotropical. Medellín: Editorial Universidad de Antioquía.

Rosa, S. M., Yema, L., Torres, P. L. M., Buemi, M. E., Balderrama, R., Palstani, M. S., Sanguinetti, A., & Martínez, C. (2024). *Machine Learning como herramienta para monitoreo*

e identificación rápida de cianobacterias. https://revistas.unlp.edu.ar/JAIIO/article/view/17887

ANEXO I

Análisis de datos

Estadísticas Descriptivas

Clorofila (µg/l)

- Media: ~101.6
- Mediana: ~19.3
- Desviación estándar: muy alta(~360.1)
- Máximo: 7049, mientras que el percentil 75 es 59

Hay una gran asimetría hacia valores altos (sesgo positivo extremo).

- La mayoría de los datos están en niveles bajos, pero hay outliers enormes que elevan mucho la media. El valor promedio no es representativo → usar mediana o percentiles es más confiable.

N-NH4 (µg/l) (Amonio)

- Media: ~75.9
- Mediana: 40
- Máximo: 1900

Alta variabilidad y asimetría fuerte: otra vez muchos valores bajos y unos pocos extremos muy altos.

N-NO2 (µg/l) (Nitrito)

- Media: 13.1
- Mediana: 9.0
- Máximo: 106

Menor dispersión que el amonio, pero sigue con algunos valores extremos.

N-NO3 (mg/l) (Nitrato)

- Media: 0.42
- Percentil 75: 0.6
- Máximo: 4.0

Mayoría de valores muy bajos. El rango es pequeño, pero con algunos valores que sobresalen.

Temperatura (°C)

- Media: ~19.5 °C
- Mediana: ~20.3 °C

Percentiles muy cercanos → datos estables y bien distribuidos.

Cianobacterias Total

- Media: ~14.4 millones de células/L
- Mediana: ~65.800
- Máximo: 2.690 millones

Increíblemente sesgado → hay pocos eventos de floraciones masivas que distorsionan las estadísticas.

Mediana mucho más representativa que la media.

Total Algas Sumatoria (Cel/L)

- Media: ~17.3 millones
- Mediana: ~2.44 millones
- Máximo: 2.775 millones

Patrón muy similar a cianobacterias → valores extremos puntuales elevan muchísimo la media.

La distribución es también fuertemente asimétrica (outliers grandes).

PHT (µg/l) – Fósforo total

- Media: ~119 µg/l
- Mediana: 74 µg/l
- Máximo: 3008 µg/l

Alta dispersión con presencia de outliers extremos, lo que indica posibles episodios de elevado aporte de fósforo, uno de los nutrientes más críticos para la proliferación de algas.

PRS (µg/l) – Fósforo reactivo soluble

- Media: ~29.4 µg/l
- Mediana: 21 µg/l
- Máximo: 447 µg/l

También muestra una distribución sesgada, aunque menos extrema que el PHT.

Cota (m)

- Media: 33.4 m
- Desviación estándar: muy baja (~1.77)
- Rango: 27.5 a 37.8

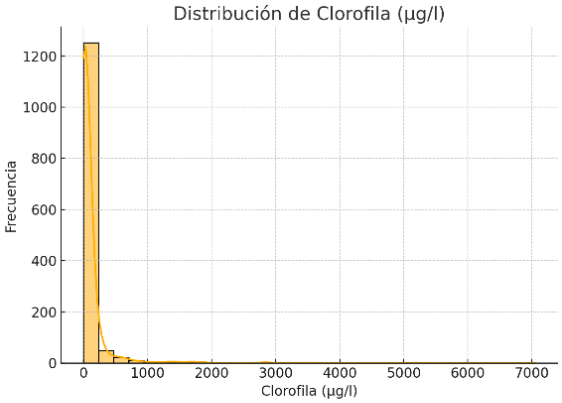
Distribución de las variables numéricas

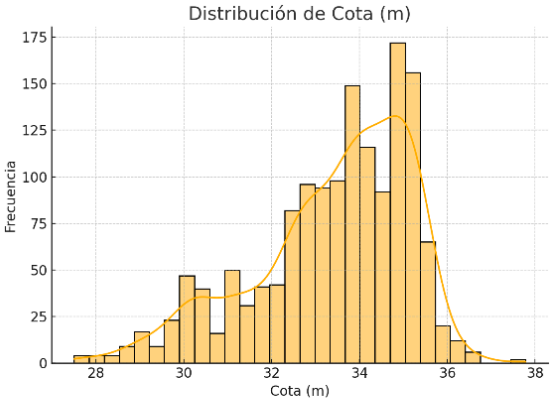
Histogramas con curvas KDE para las variables numéricas del dataset.

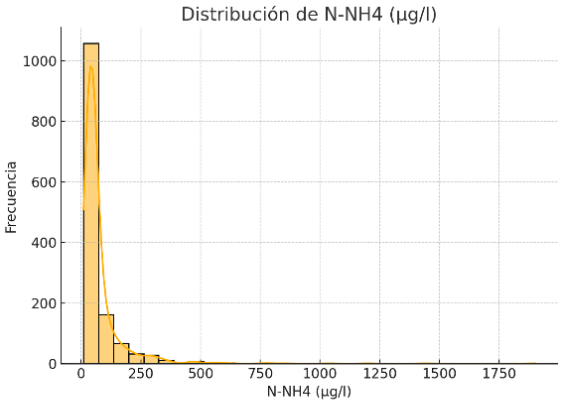
- Muchas variables, como **Clorofila**, **Cianobacterias Total**, **Total Algas**, y algunos nutrientes, presentan distribuciones **sesgadas a la derecha**, con valores altos poco frecuentes (outliers).

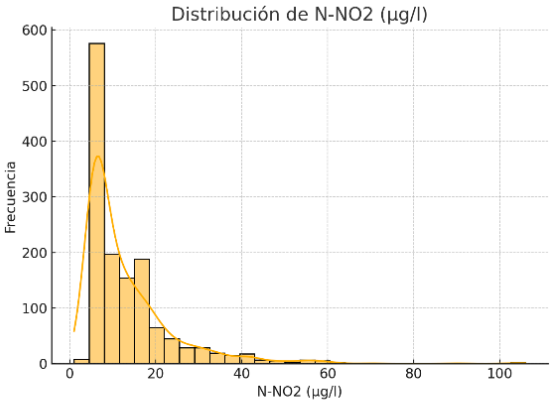
- La temperatura (**T° (°C)**) tiene una distribución más simétrica, posiblemente normal.

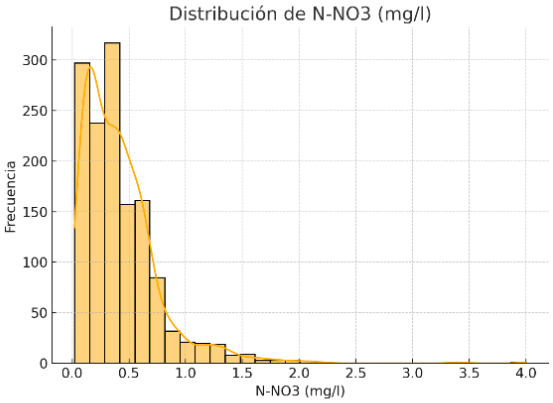
- La **Cota (m)** tiene una distribución bastante concentrada.

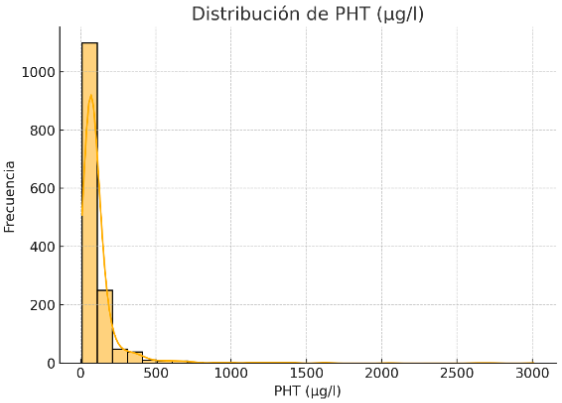

Distribución de Clorofila (µg/l)
Frecuencia
Clorofila (µg/l)

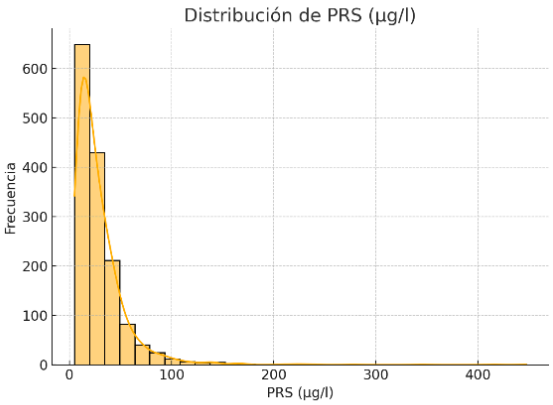

Distribución de Cota (m)
Frecuencia
Cota (m)

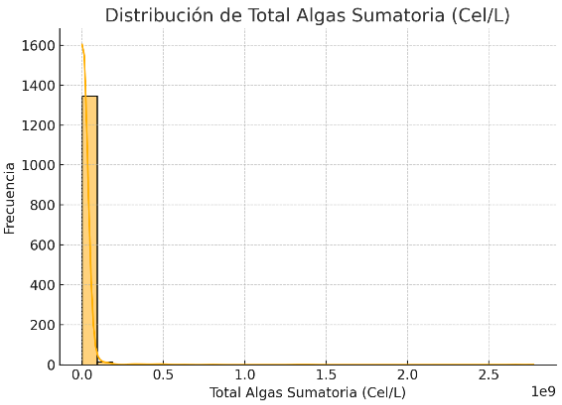

Distribución de N-NH4 (µg/l)
Frecuencia
N-NH4 (µg/l)

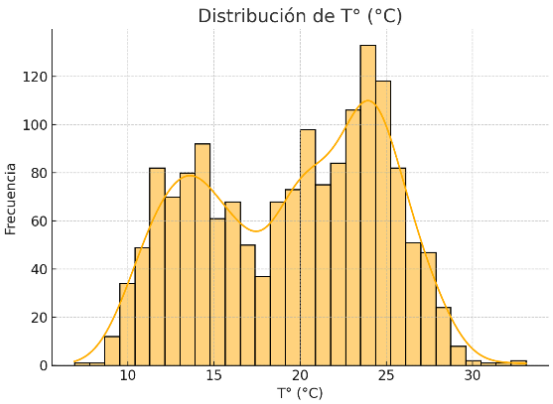

Distribución de N-NO2 (µg/l)
Frecuencia
N-NO2 (µg/l)

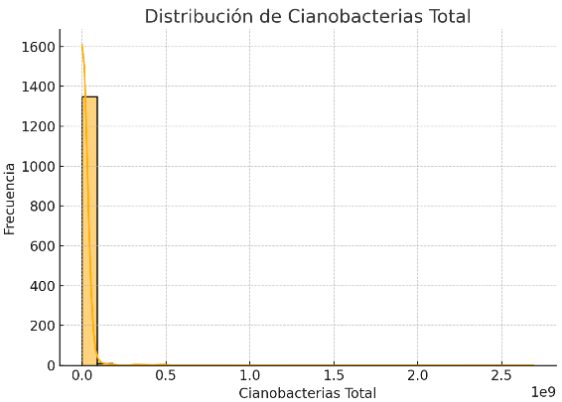

Distribución de N-NO3 (mg/l)
Frecuencia
N-NO3 (mg/l)


Distribución de PHT (µg/l)
Frecuencia
PHT (µg/l)

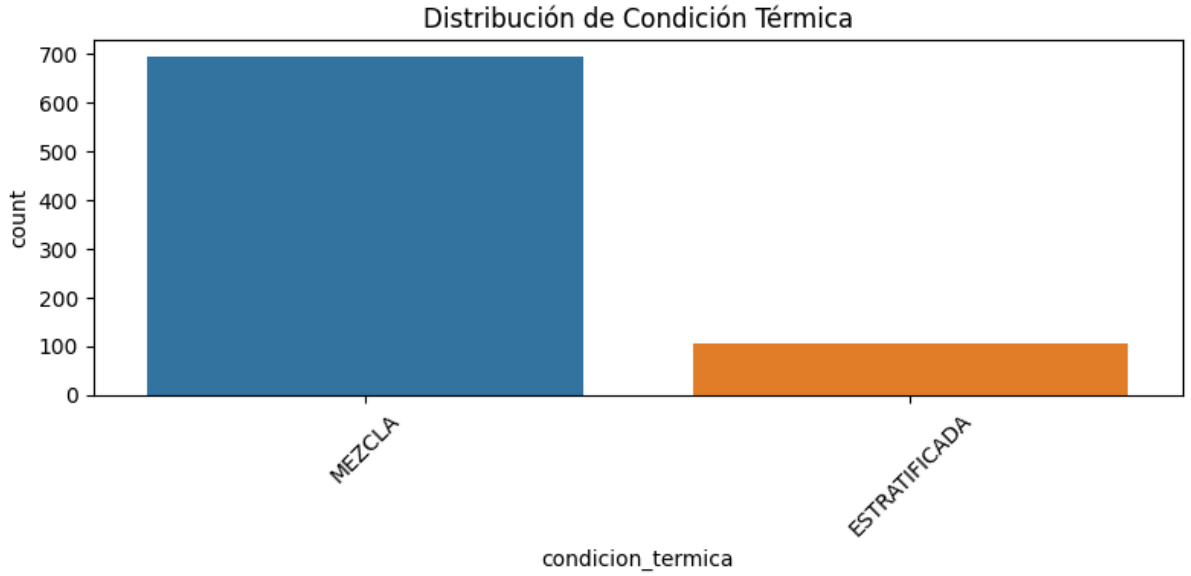

Distribución de PRS (µg/l)
Frecuencia
PRS (µg/l)

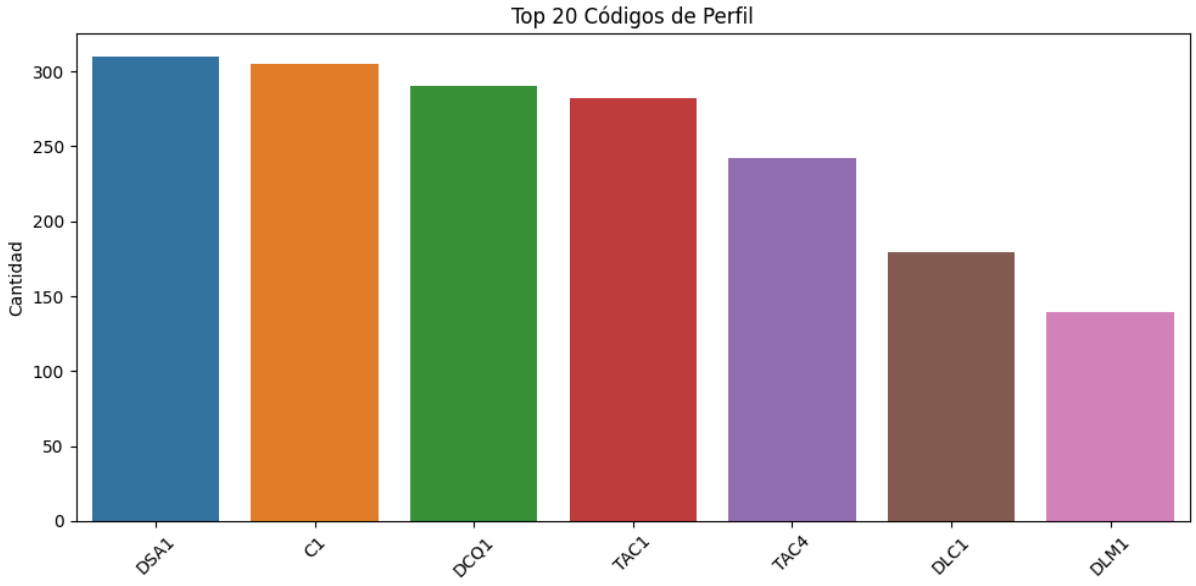

Distribución de Total Algas Sumatoria (Cel/L)
Frecuencia
Total Algas Sumatoria (Cel/L)
1e9


Distribución de T° (°C)
Frecuencia
T° (°C)


Distribución de Cianobacterias Total
Frecuencia
Cianobacterias Total
1e9

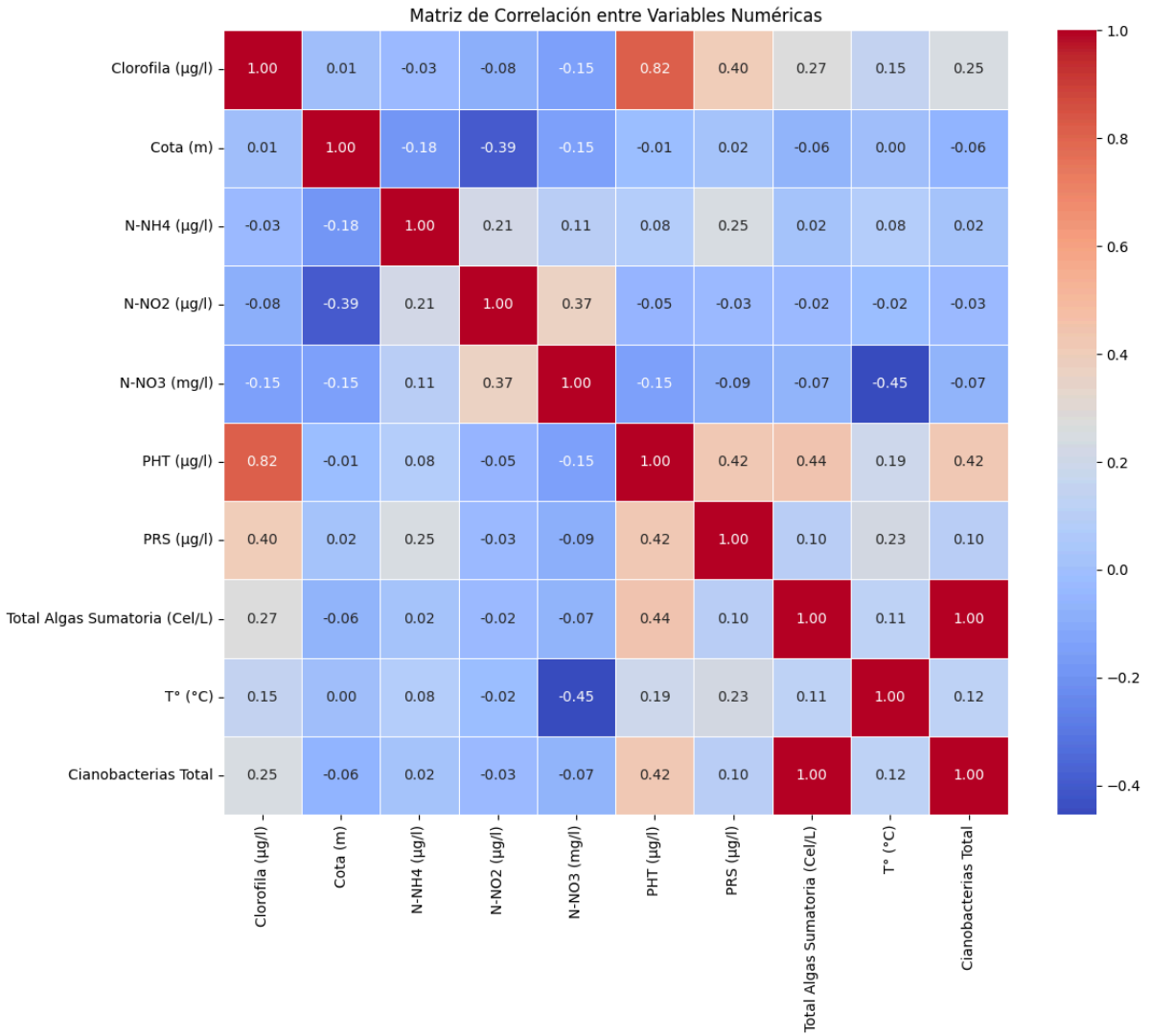
Gráficos para variables categóricas





Muestra que los sitios DLC1 y DLM1 tienen una significativa cantidad menor de datos

Matriz de Correlación entre Variables Numéricas



Fuerte correlación positiva entre:

- **Cianobacterias Total** y **Total Algas Sumatoria**
- **Clorofila** y **Cianobacterias/Algas Totales**
- **PHT** y **Clorofila**

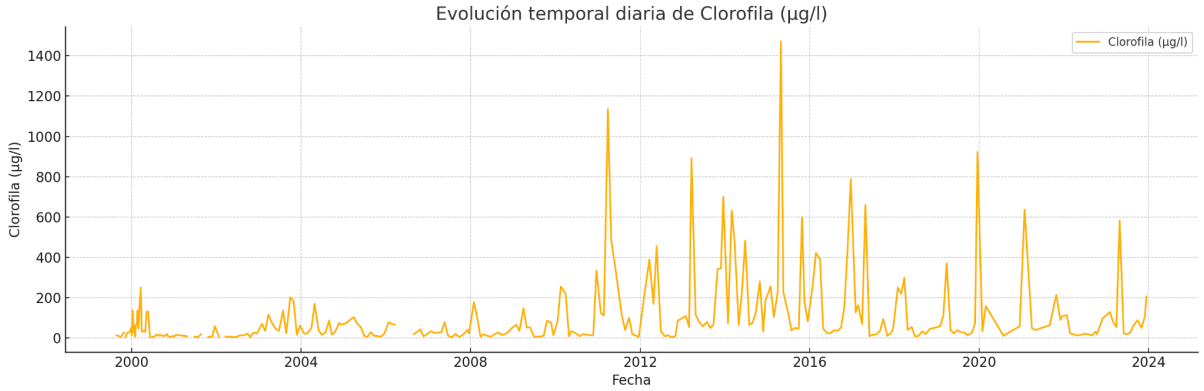
Más adelante se realizan matrices de correlación por sitio de medición y por estación del año

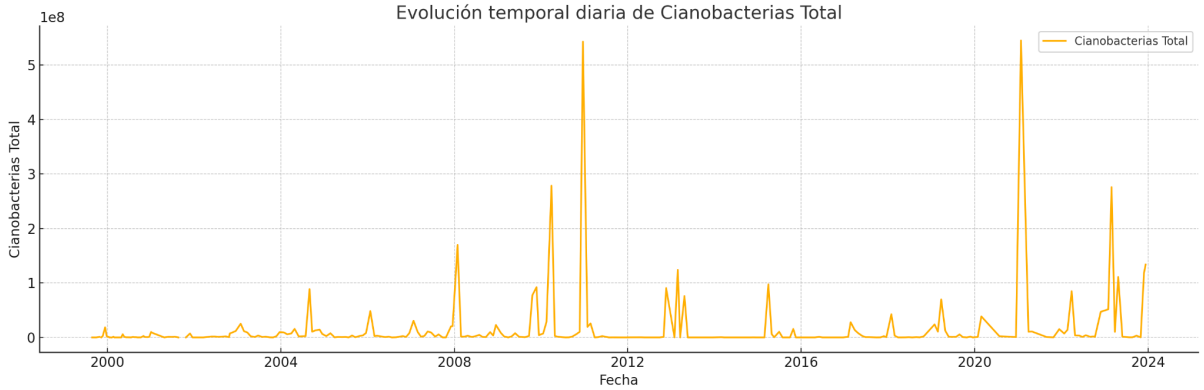
Gráficos de evolución temporal

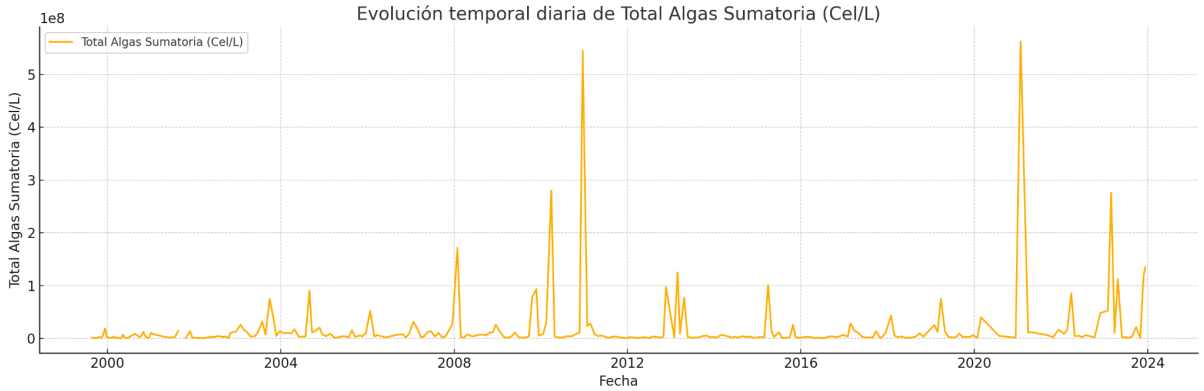
Clorofila, **Cianobacterias**, **Total Algas**:

Muestran una evolución con **eventos puntuales de alta concentración**, pero también muchas etapas de calma.

Coinciden los picos → podría indicar **floraciones estacionales** o condiciones específicas de cada año.



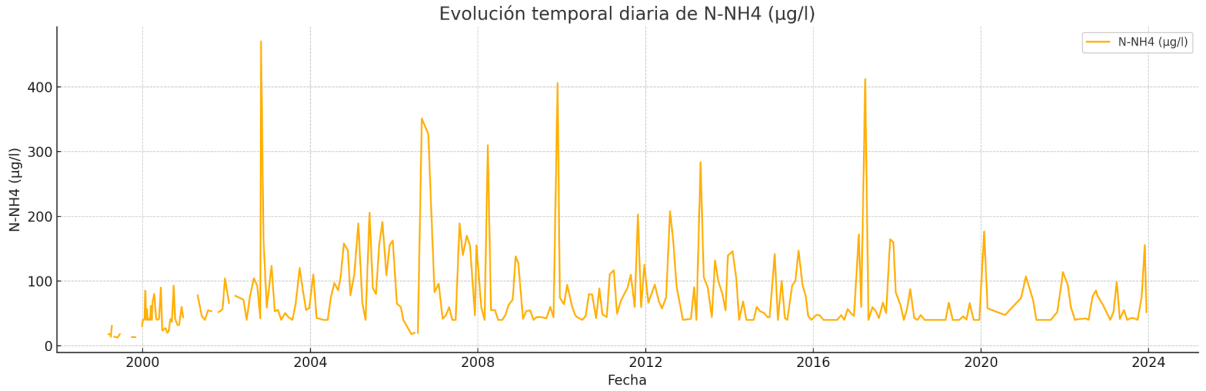


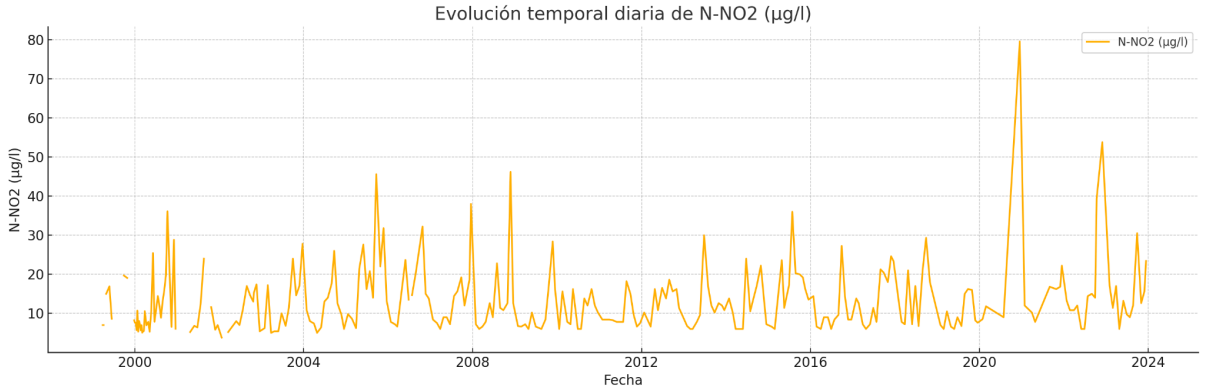


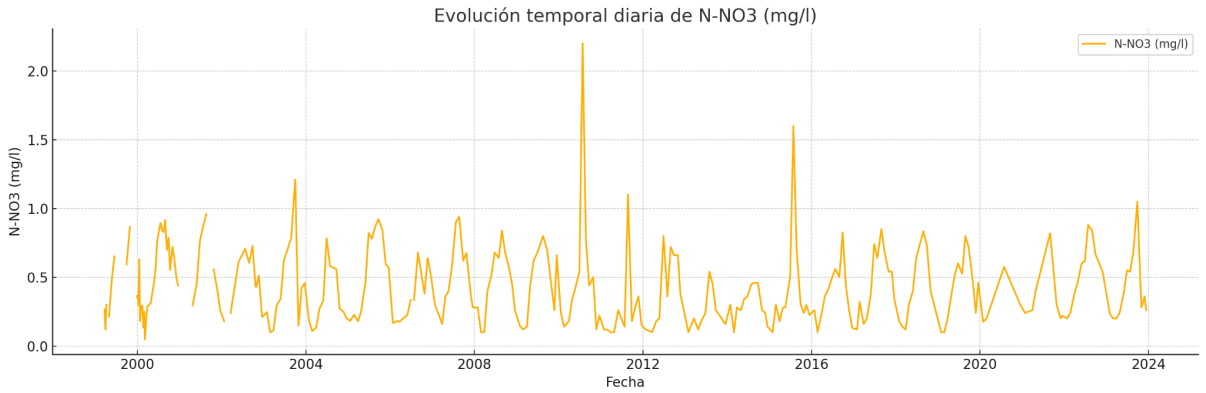
Nutrientes (NH4, NO2, NO3):

Gran variabilidad, especialmente en NH4 (amonio), con picos abruptos.

N-NO3 (nitrato) es más estable, pero también muestra aumentos puntuales.



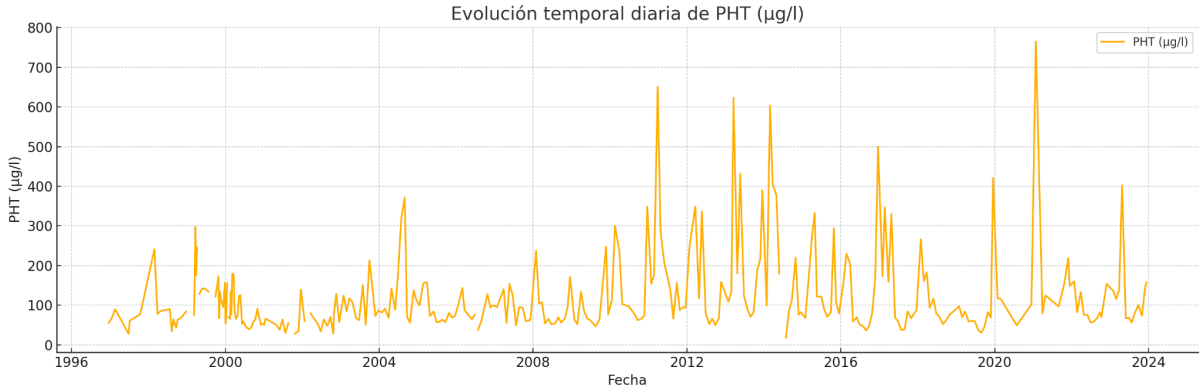


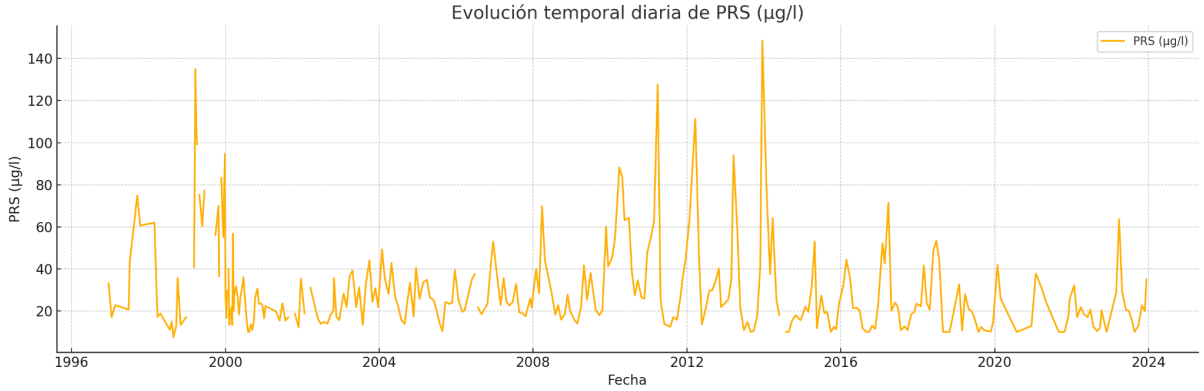


Fósforo (PHT y PRS):

PHT tiene algunos valores extremos muy altos

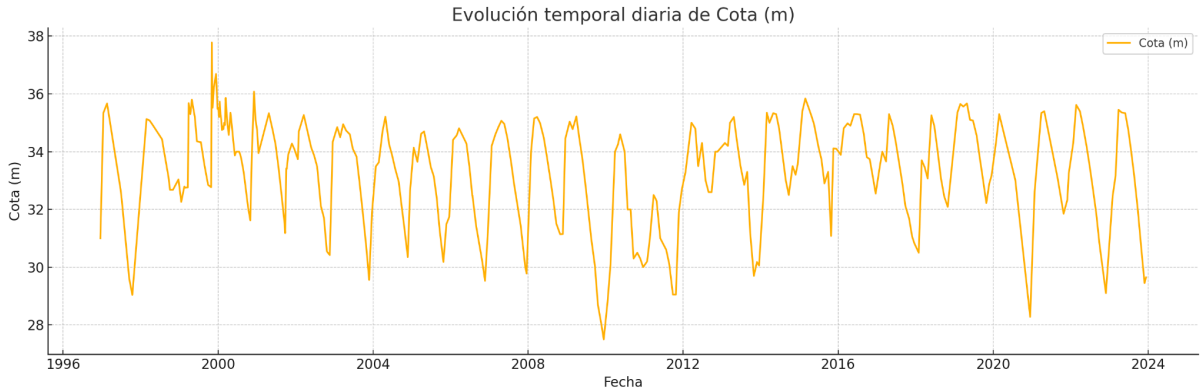
PRS sigue un patrón parecido, aunque más atenuado.





Cota (m):

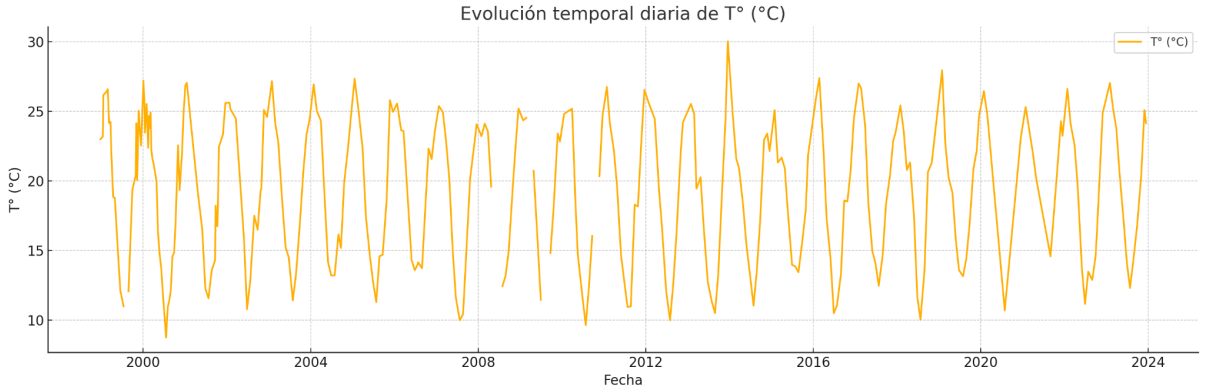
Altamente constante en el tiempo.



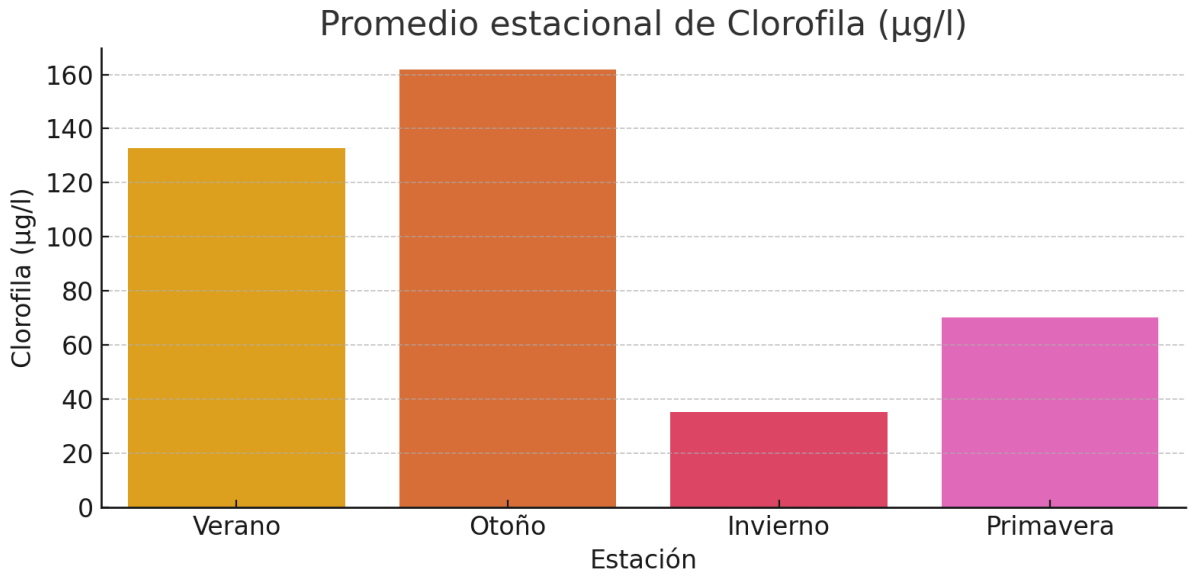
Temperatura (T°):

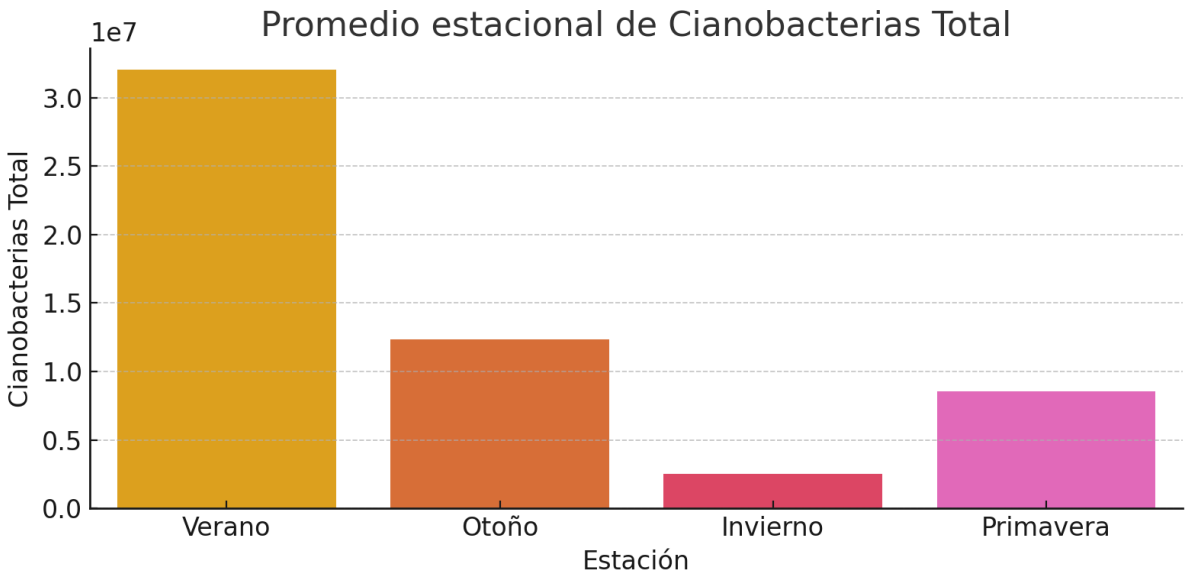
Patrón estacional marcado.

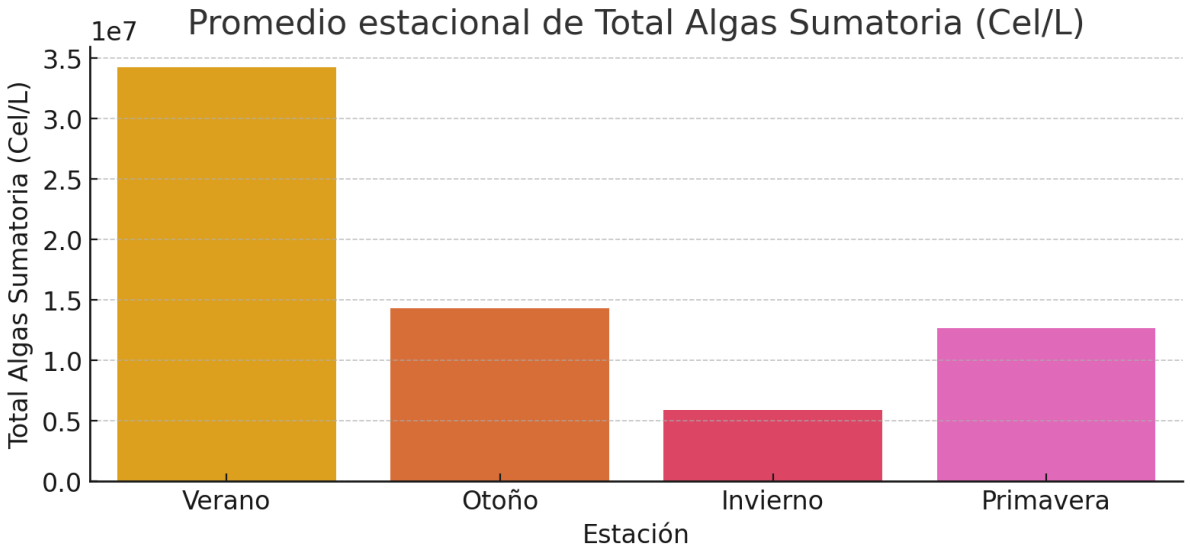
Útil para correlacionar con aumentos de clorofila y cianobacterias.

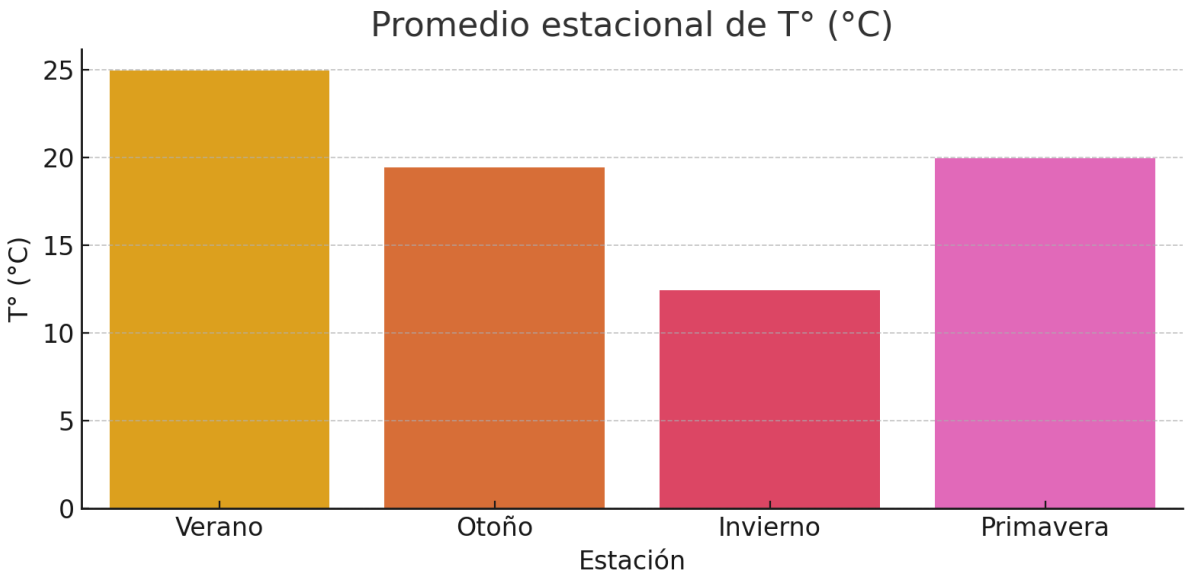


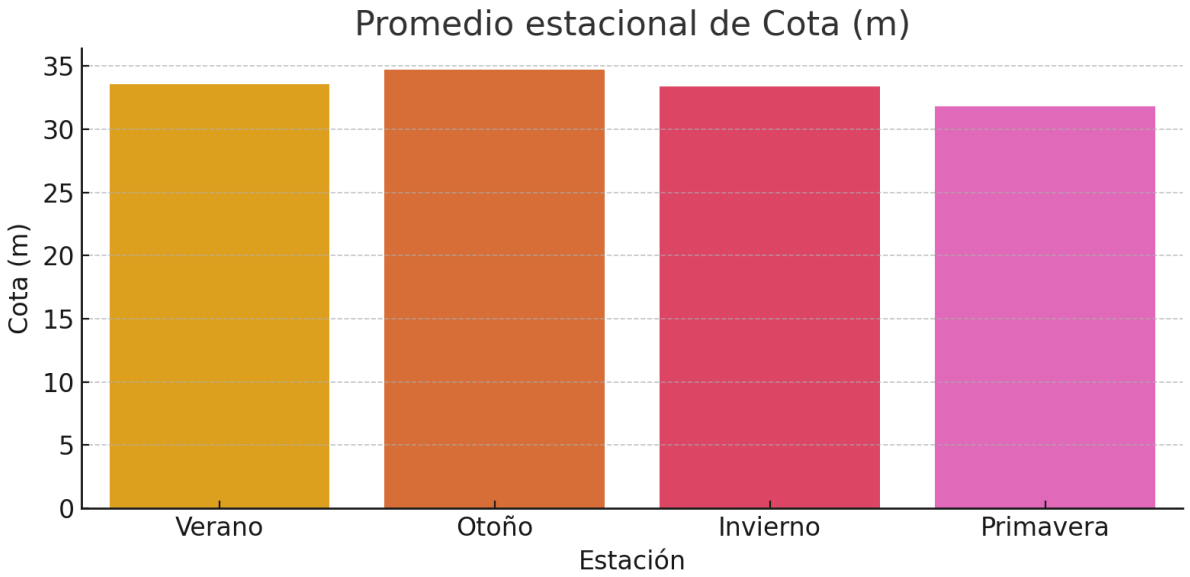
Promedios estacionales:

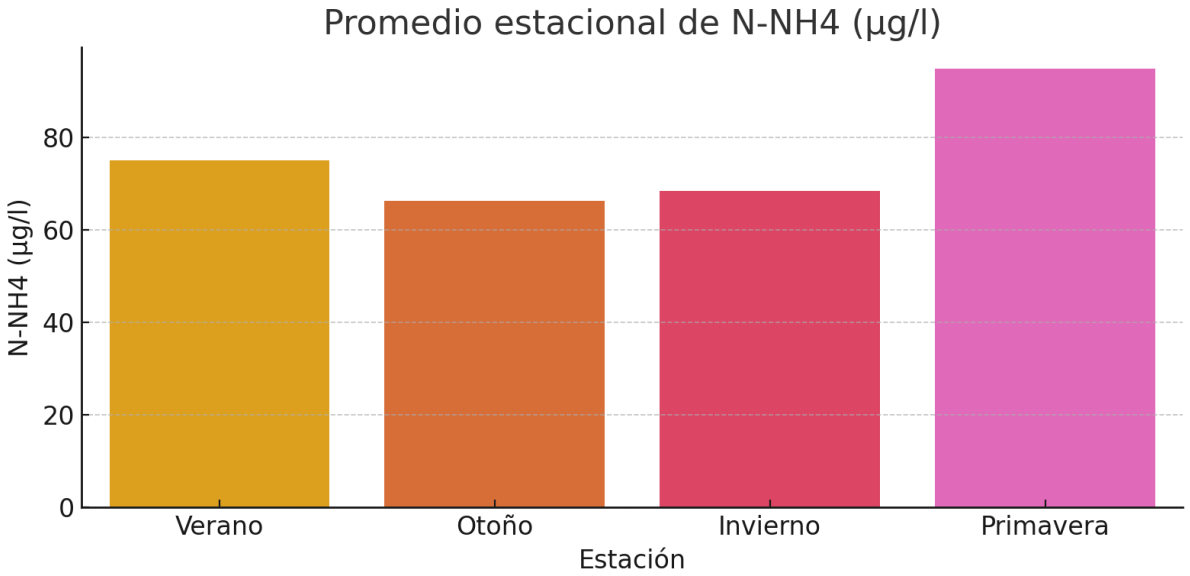


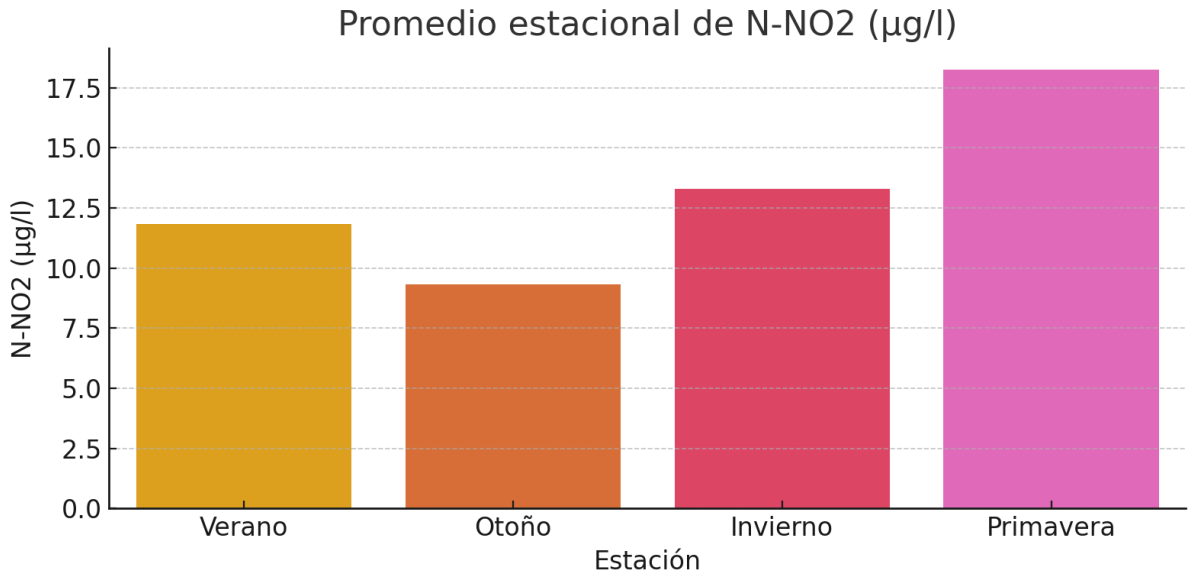

Promedio estacional de Cianobacterias Total
1e7
3.0
2.5
2.0
1.5
1.0
0.5
0.0
Cianobacterias Total
Verano
Otoño
Invierno
Primavera
Estación

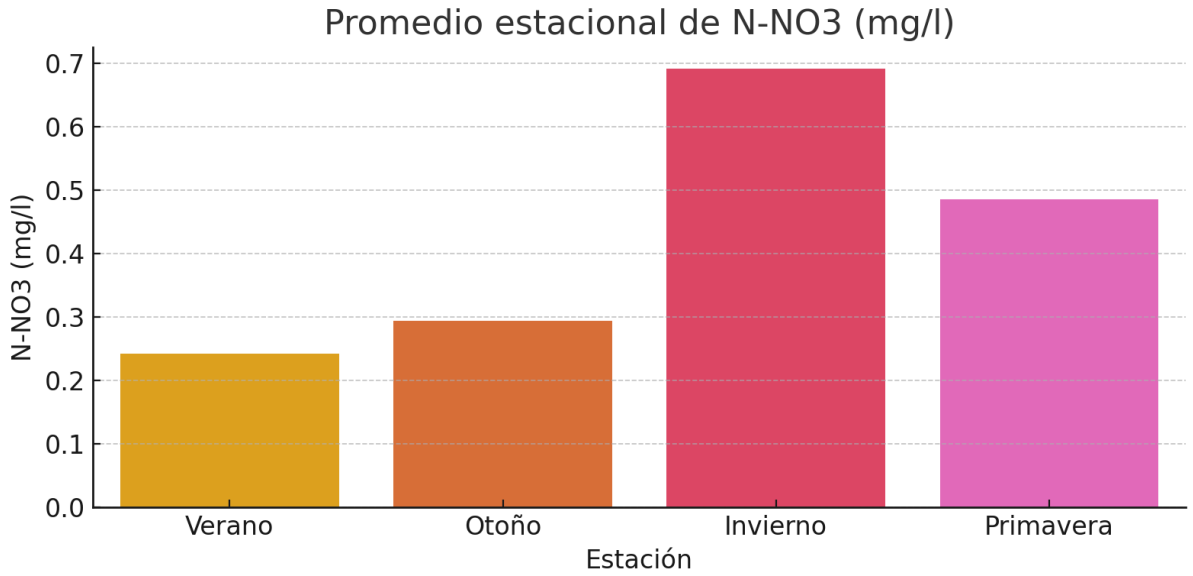

Promedio estacional de Total Algas Sumatoria (Cel/L)
1e7
3.5
3.0
2.5
2.0
1.5
1.0
0.5
0.0
Total Algas Sumatoria (Cel/L)
Verano
Otoño
Invierno
Primavera
Estación

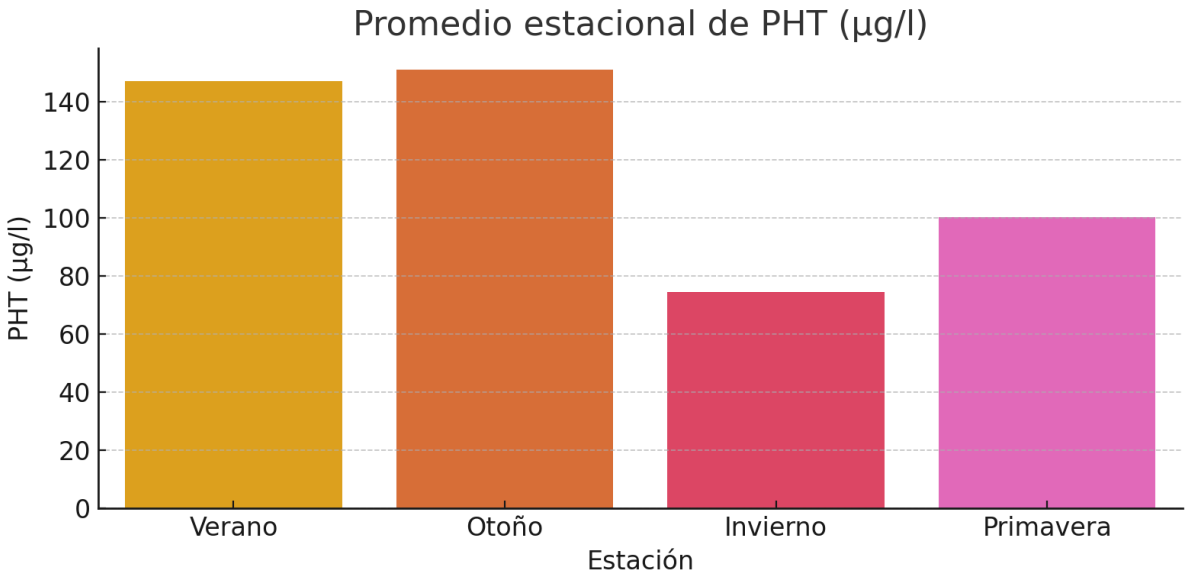

Promedio estacional de T° (°C)
25
20
15
10
5
0
T° (°C)
Verano
Otoño
Invierno
Primavera
Estación

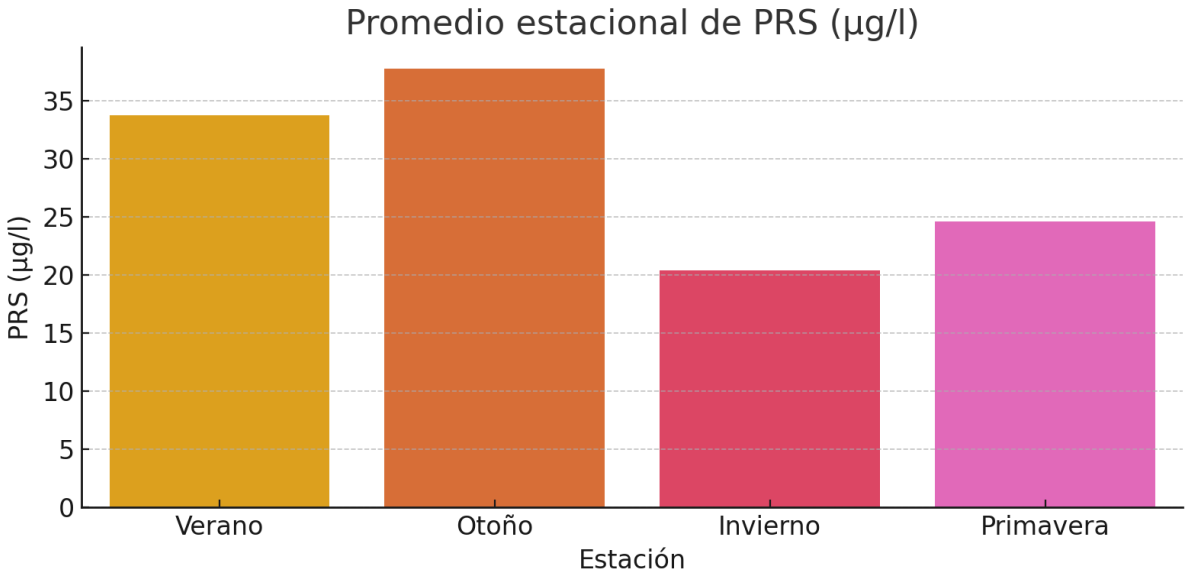

Promedio estacional de Cota (m)
Cota (m)
0
5
10
15
20
25
30
35
Verano
Otoño
Invierno
Primavera
Estación


Promedio estacional de N-NH4 (µg/l)
N-NH4 (µg/l)
0
20
40
60
80
Verano
Otoño
Invierno
Primavera
Estación

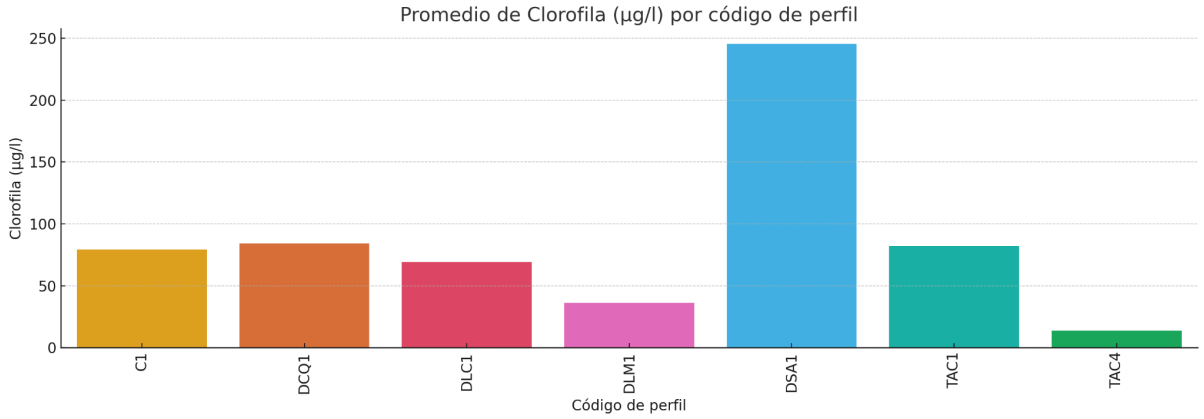

Promedio estacional de N-NO2 (µg/l)
N-NO2 (µg/l)
17.5
15.0
12.5
10.0
7.5
5.0
2.5
0.0
Verano
Otoño
Invierno
Primavera
Estación

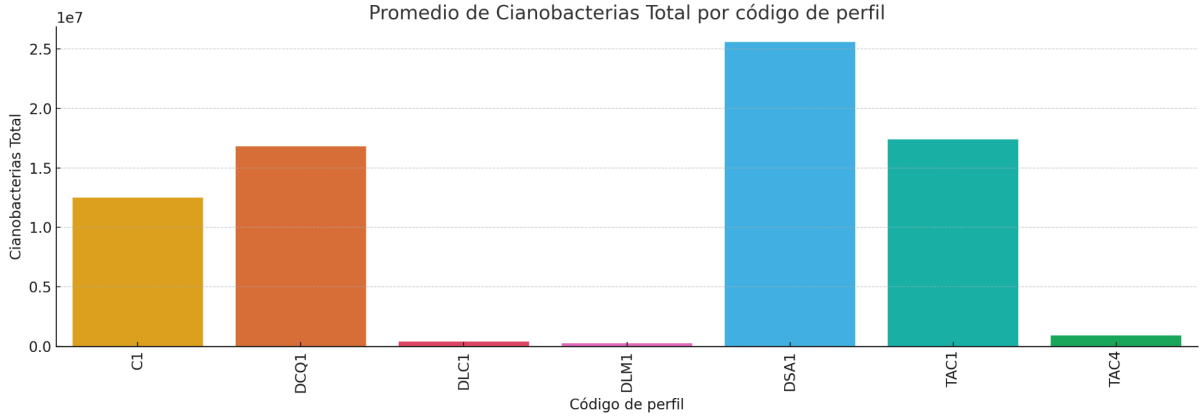

Promedio estacional de N-NO3 (mg/l)
N-NO3 (mg/l)
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0.0
Verano
Otoño
Invierno
Primavera
Estación

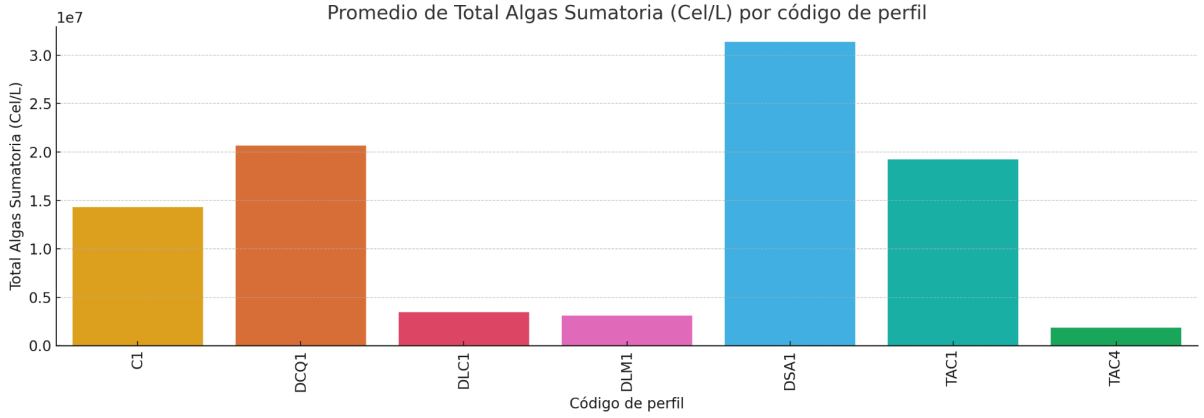


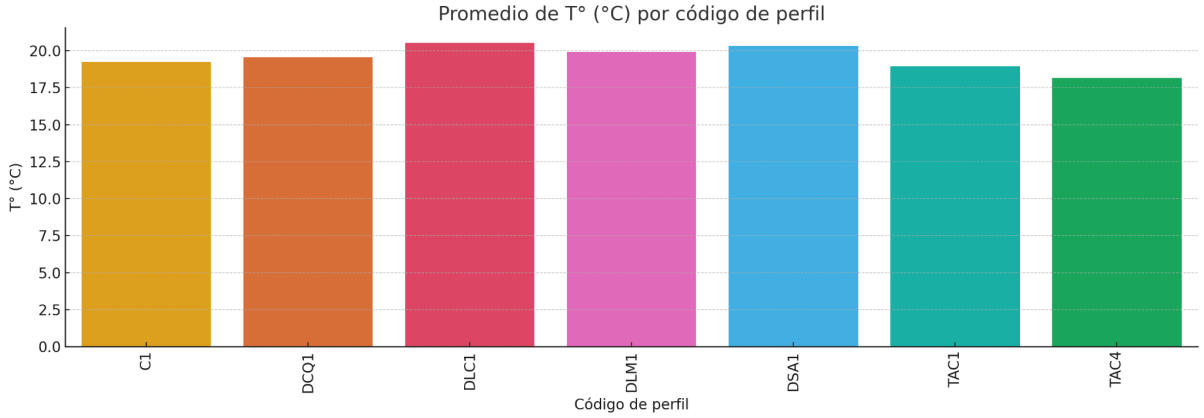


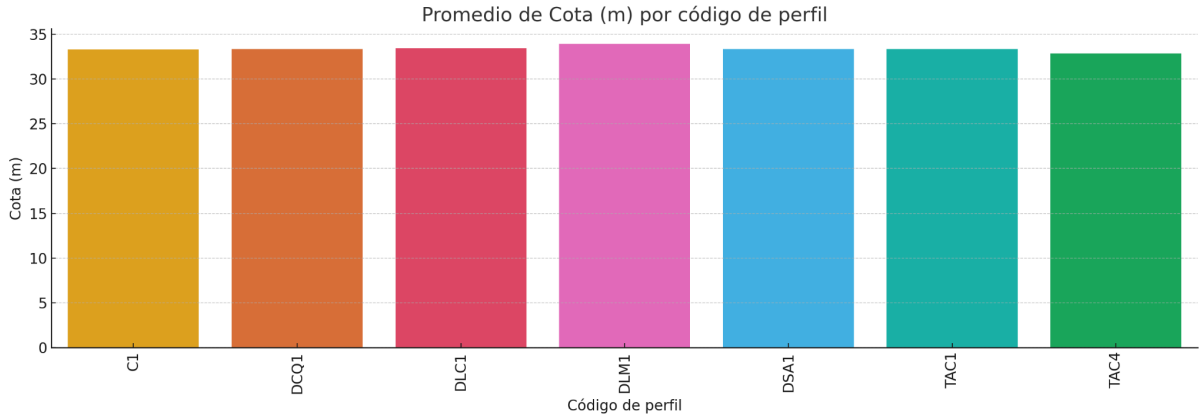
Análisis espacial: Promedio por sitios de medición


Promedio de Clorofila (µg/l) por código de perfil
Clorofila (µg/l)
0
50
100
150
200
250
C1
DCQ1
DLC1
DLM1
DSA1
TAC1
TAC4
Código de perfil

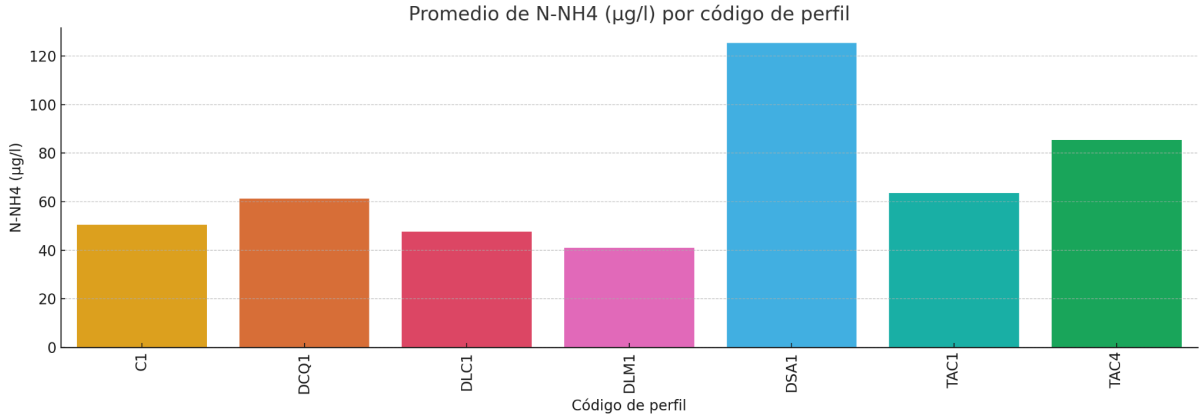

Promedio de Cianobacterias Total por código de perfil
1e7
Cianobacterias Total
0.0
0.5
1.0
1.5
2.0
2.5
C1
DCQ1
DLC1
DLM1
DSA1
TAC1
TAC4
Código de perfil

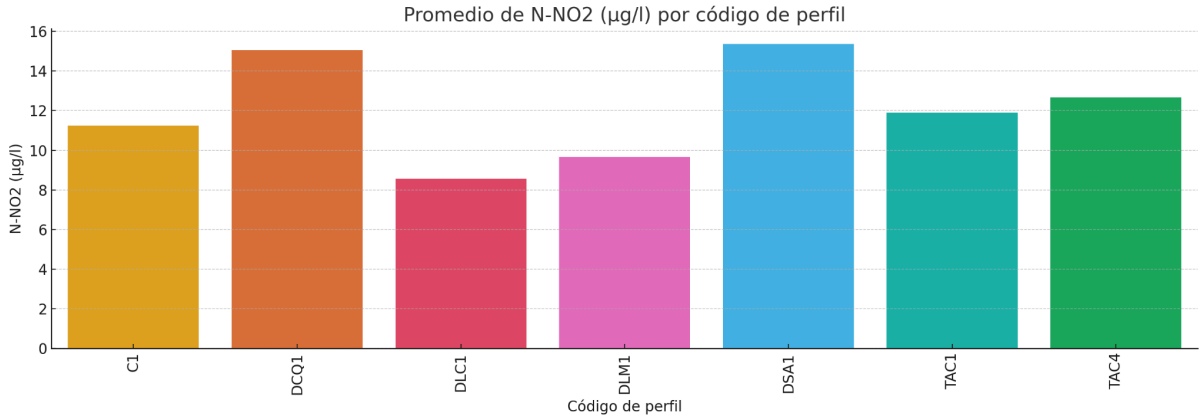

Promedio de Total Algas Sumatoria (Cel/L) por código de perfil
1e7
Total Algas Sumatoria (Cel/L)
0.0
0.5
1.0
1.5
2.0
2.5
3.0
C1
DCQ1
DLC1
DLM1
DSA1
TAC1
TAC4
Código de perfil

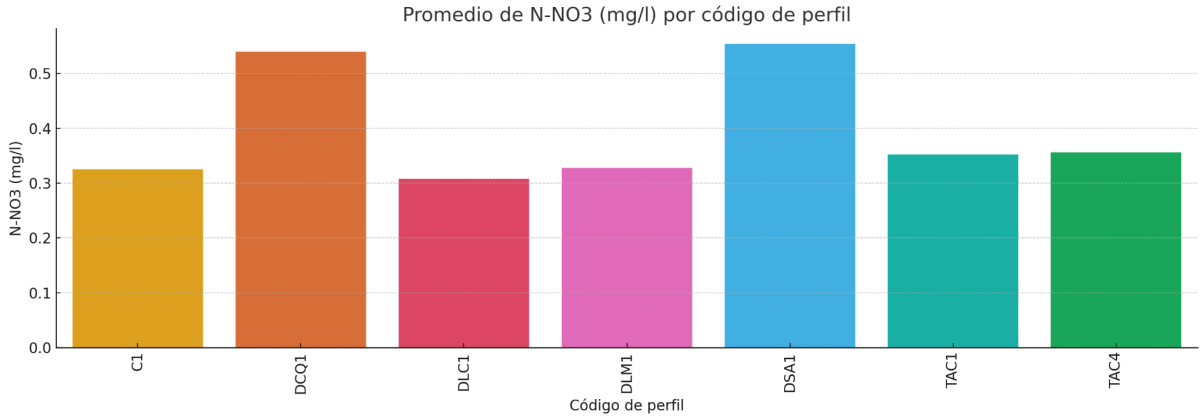

Promedio de T° (°C) por código de perfil
T° (°C)
0.0
2.5
5.0
7.5
10.0
12.5
15.0
17.5
20.0
C1
DCQ1
DLC1
DLM1
DSA1
TAC1
TAC4
Código de perfil

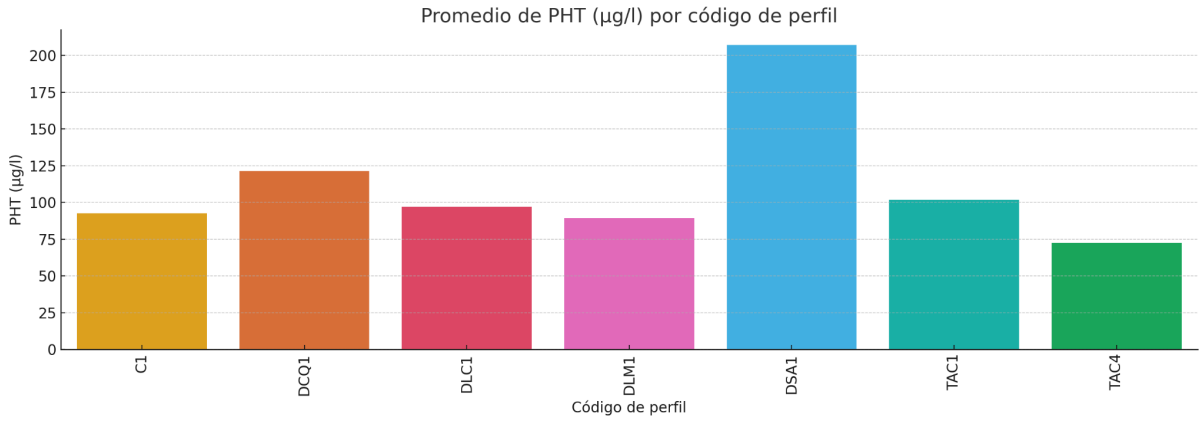


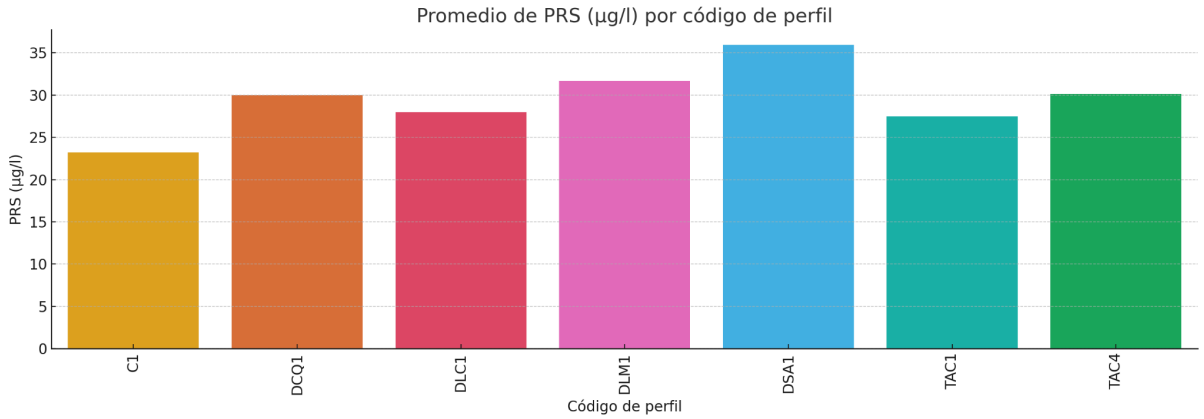
Mismo nivel del lago para todos los sitios






Promedio de N-NO3 (mg/l) por código de perfil
N-NO3 (mg/l)
0.0
0.1
0.2
0.3
0.4
0.5
C1
DCQ1
DLC1
DLM1
DSA1
TAC1
TAC4
Código de perfil


Promedio de PHT (µg/l) por código de perfil
PHT (µg/l)
0
25
50
75
100
125
150
175
200
C1
DCQ1
DLC1
DLM1
DSA1
TAC1
TAC4
Código de perfil


Promedio de PRS (µg/l) por código de perfil
PRS (µg/l)
0
5
10
15
20
25
30
35
C1
DCQ1
DLC1
DLM1
DSA1
TAC1
TAC4
Código de perfil

ANEXO II

Arquitectura e Hiperparámetros de los Modelos

A continuación, se presentan las configuraciones de los modelos de aprendizaje automático utilizados en este trabajo. Se detallan sus arquitecturas y los principales hiperparámetros considerados durante el proceso de ajuste. Para una explicación más extensa sobre la lógica de selección y entrenamiento de cada modelo, véase la sección **IMPLEMENTACIÓN** del cuerpo principal de la tesis.

Regresión Logística (Logistic Regression)

Tipo de modelo: Clasificación lineal.

Hiperparámetro ajustado:

$C \in \{0.1, 1.0, 10.0\}$

Random Forest

Tipo de modelo: Ensamble de árboles de decisión.

Hiperparámetros ajustados:

$n\_estimators \in \{100, 200\}$

$max\_depth \in \{5, 10\}$

Perceptrón Multicapa (MLP – Multi-Layer Perceptron)

Tipo de modelo: Red neuronal artificial supervisada.

Arquitectura definida:

- Capa de entrada: número de neuronas = número de variables seleccionadas.
- Capa oculta (1): 1 capa densa con entre 32 y 256 neuronas, activación ReLU.
  Con BatchNormalization y Dropout (0.2 – 0.5) para normalización y regularización.
- Capa de salida: número de clases, activación Softmax.

Parámetros de entrenamiento:

- Optimizador: Adam
- Función de pérdida: sparse_categorical_crossentropy.
- Métrica: accuracy.
- Épocas: hasta 50 con EarlyStopping.
- Batch size: 16.