Modelamiento predictivo del desempeño territorial en la implementación de las Rutas Integrales de Atención en Salud (RIAS) en Colombia mediante aprendizaje automático

Luis Edilson Cortes Lozano

Asesor

Luis Angel Anillo Arrieta

Universidad Nacional Abierta y a Distancia UNAD

Escuela de Ciencias Básicas, Tecnología e Ingeniería ECBTI

Especialización en Ciencia de Datos y Analítica

2025

Resumen

El proyecto desarrolla un modelo predictivo orientado a estimar el desempeño territorial en la implementación de las Rutas Integrales de Atención en Salud (RIAS) en Colombia, utilizando series históricas del Monitor RIAS del SISPRO y técnicas avanzadas de aprendizaje automático. La metodología CRISP-DM permitió estructurar el proceso analítico desde la depuración de datos hasta la evaluación de los modelos, garantizando rigor metodológico y coherencia en los distintos niveles de desagregación territorial.

Los algoritmos de ensamblaje —Random Forest, XGBoost y LightGBM— obtuvieron los mejores resultados, alcanzando valores de explicabilidad (R^2) entre 0,80 y 0,98, lo que evidencia una alta capacidad para modelar la variabilidad real de los indicadores y generar predicciones confiables. Con base en estos modelos se elaboró una proyección combinada para el periodo 2025–2027, que muestra estabilidad en los indicadores maternos y una progresión gradual en dimensiones como salud bucal, salud visual y detección temprana.

En conjunto, los resultados consolidan una herramienta predictiva robusta y aplicable para la planeación estratégica en salud pública, facilitando la identificación temprana de brechas territoriales y el fortalecimiento de la toma de decisiones basada en evidencia.

***Palabras claves*:** Salud pública, Rutas Integrales de Atención en Salud, desempeño territorial, ciencia de datos, modelamiento predictivo, gestión riesgo, aprendizaje automático.

Abstract

This project develops a predictive model aimed at estimating territorial performance in the implementation of the Comprehensive Health Care Routes (RIAS) in Colombia, using historical series from the SISPRO RIAS Monitor and advanced machine learning techniques. Following the CRISP-DM methodology, the analytical process encompassed data preparation, model training, and evaluation across national, departmental, municipal, and institutional levels.

Ensemble algorithms—Random Forest, XGBoost, and LightGBM—achieved the highest levels of explainability, with R^2 values ranging from 0.80 to 0.98, demonstrating a strong capacity to model the real variability of RIAS indicators and produce reliable predictions. Using these models, a combined projection for 2025–2027 was generated, revealing stability in maternal-health indicators and gradual improvements in oral health, visual health, and early detection metrics.

Overall, the results consolidate a robust predictive framework that supports strategic health-system planning, enhances early identification of territorial gaps, and strengthens evidence-based decision-making in public health.

Keywords**:** Public health, Comprehensive Health Care Routes, territorial performance, data science, predictive modeling, risk management, machine learning.

Tabla de Contenido

Introducción 10

Planteamiento del Problema 11

Justificación 12

Objetivos 13

Objetivo General 13

Objetivos Específicos 13

Marco Conceptual 14

Rutas Integrales de Atención en Salud 14

Finalidad de las RIAS 14

Tipos de Intervenciones 14

Enfoques Orientadores 14

Adaptabilidad Territorial y Gestión del Riesgo 15

Desempeño Territorial en Salud 15

Indicadores de las RIAS 15

Analítica de Datos en Salud 15

Aprendizaje Supervisado y Regresión 15

Regresión con Árboles de Decisión 16

Modelos Predictivos en Salud 16

Métricas de Evaluación de Regresión 16

Marco Teórico 17

Marco Normativo de las RIAS 17

Analítica de Datos en Salud y Toma de Decisiones 17

Aportes Metodológicos para el Análisis Territorial 17
Enfoques Institucionales, Desigualdad y Brechas Territoriales 18
Aprendizaje Automático y Comparación de Algoritmos 18
Modelos Multivariables y Dimensión Espacio -Temporal 18
Procesamiento Multivariable de Datos Territoriales 19
Aplicación de IA en Datos Agregados en Salud 19
Analítica Territorial Aplicada a Cobertura y Desempeño Institucional 19
Modelos Eficientes en Contextos de Baja Capacidad Tecnológica 20
Procesamiento Avanzado y Detección de Patrones 20
Variables Socioeconómicas y Ambientales como Predictoras 20
Diseño Metodológico Integral y Selección de Algoritmos 20
Metodología 21
Comprensión del Negocio 21
Comprensión de los Datos 22
Preparación de los Datos 22
Modelado 23
Evaluación 23
Despliegue 24
Cronograma de Actividades 25
Presupuesto del Proyecto 26
Resultados o Productos Esperados 27
Resultados 28
Análisis de los Datos Históricos y Tendencias Territoriales 28

Aplicación y Comparación de Modelos de Aprendizaje Supervisado 30

Evaluación del Desempeño de los Modelos Predictivos 34

Resultados de la Validación y Proyección 38

Conclusiones 41

Recomendaciones 43

Referencias Bibliográficas 44

Apéndices 49

Lista de Tablas

Tabla 1 *Cronograma de Trabajo del Proyecto Según la Metodología CRISP-DM* 25

Tabla 2 *Presupuesto del Proyecto* 26

Tabla 3 *Resultados o Productos Esperados* 27

Tabla 4 *Comparativo de Modelos Supervisados de Regresión – Nivel Nacional* 35

Tabla 5 *Comparativo de Modelos Supervisados de Regresión – Nivel Departamental* 36

Tabla 6 *Comparativo de Modelos Supervisados de Regresión – Nivel Municipal* 37

Tabla 7 *Comparativo de Modelos Supervisados de Regresión – Nivel Asegurador* 38

Lista de Figuras

Figura 1 *Fases del Proceso CRISP-DM para el Desarrollo de Modelos Analíticos* 21

Figura 2 *Diagrama Conceptual del Problema: Predicción del Desempeño Territorial* 22

Figura 3 *Indicador Agrupado en Seis Clusters de Cumplimiento* 28

Figura 4 *Dimensiones de Salud y Clústeres de Cumplimiento* 29

Figura 5 *Tendencias Territoriales por Clúster* 30

Figura 6 *Evaluación Nacional por Modelo* 31

Figura 7 *Evaluación de Modelos por Departamento* 32

Figura 8 *Evaluación de Modelos por Municipio* 33

Figura 9 *Evaluación de Modelos por Aseguradora* 34

Figura 10 *Proyección de Resultados del Monitor RIAS (2018–2027)* 40

Lista de Apéndices

Apéndice A *Análisis Exploratorio de Datos (EDA) – Monitor RIAS* 49

Apéndice B *Modelos Predictivos y Evaluación – Nivel Nacional* 62

Apéndice C *Modelos Predictivos y Evaluación – Nivel Departamental* 72

Apéndice D *Modelos Predictivos y Evaluación – Nivel Municipal* 83

Apéndice E *Modelos Predictivos y Evaluación – Nivel Institucional* 92

Apéndice F *Proyección de Resultados y Modelo Combinado por Votación* 103

Apéndice G *Fuentes de Datos y Repositorio de Bases Utilizadas* 108

Introducción

La implementación de las Rutas Integrales de Atención en Salud (RIAS) constituye uno de los mayores desafíos del sistema de salud colombiano, marcado por profundas diferencias territoriales en capacidad operativa, gestión institucional y cumplimiento de los indicadores. Aunque el Monitor RIAS del SISPRO ofrece una base sólida para el seguimiento, su carácter retrospectivo limita la anticipación de riesgos y dificulta orientar acciones oportunas en los diferentes niveles de gestión.

Este contexto evidencia la necesidad de metodologías que permitan transformar los datos históricos en capacidades predictivas, facilitando la identificación temprana de brechas, la proyección del desempeño y la toma de decisiones basada en evidencia. La analítica avanzada y el aprendizaje automático brindan herramientas idóneas para este propósito al permitir modelar tendencias, reconocer patrones y generar estimaciones confiables del comportamiento territorial.

En respuesta a esta necesidad, el presente proyecto desarrolla y evalúa modelos de aprendizaje supervisado orientados a predecir el cumplimiento de los indicadores RIAS a nivel nacional, departamental, municipal e institucional. El objetivo es ofrecer un insumo técnico que fortalezca la planificación estratégica, el monitoreo continuo y la gestión del riesgo en salud pública, contribuyendo a la equidad y efectividad de la implementación de las rutas en el país.

Planteamiento del Problema

La adopción de las Rutas Integrales de Atención en Salud (RIAS) busca garantizar un acceso oportuno y continuo a intervenciones esenciales en todo el territorio nacional. Sin embargo, su implementación presenta marcadas desigualdades entre departamentos, municipios y aseguradores, reflejadas en variaciones significativas en el cumplimiento de los indicadores. Aunque el Monitor RIAS del SISPRO ofrece información histórica para el seguimiento, su enfoque descriptivo no permite anticipar rezagos ni proyectar escenarios futuros, lo que limita la capacidad institucional para intervenir de manera temprana y focalizada.

La ausencia de metodologías analíticas que integren predicción, análisis multivariable y comprensión de patrones territoriales dificulta reconocer oportunamente brechas estructurales, identificar comportamientos atípicos y orientar recursos de manera equitativa. Esta limitación se acentúa en territorios con menor capacidad técnica y operativa, donde la variabilidad en el cumplimiento de los indicadores no suele detectarse a tiempo para ajustar las estrategias de gestión.

En este contexto, surge el interrogante central del proyecto: ¿cómo puede un modelo de aprendizaje automático predecir el desempeño territorial de los indicadores de las RIAS y fortalecer la toma de decisiones en salud pública? Abordar esta pregunta resulta esencial para avanzar hacia sistemas de monitoreo preventivo, orientar acciones diferenciales y mejorar la efectividad de la política pública en salud.

Justificación

El fortalecimiento de la gestión pública en salud requiere herramientas que permitan anticiparse a los riesgos territoriales asociados a la implementación de las Rutas Integrales de Atención en Salud (RIAS). Aunque el Monitor RIAS del SISPRO provee información histórica relevante, su carácter retrospectivo limita la capacidad institucional para actuar de manera proactiva, identificar brechas estructurales y orientar recursos hacia los territorios con mayores rezagos. Esta insuficiencia analítica se refleja en desigualdades persistentes en el cumplimiento de los indicadores, especialmente en regiones con menor capacidad operativa.

Desde la perspectiva técnica, la analítica avanzada y el aprendizaje automático ofrecen la posibilidad de transformar datos administrativos en modelos predictivos capaces de estimar el desempeño territorial, identificar patrones de cumplimiento y generar alertas tempranas. La construcción de un modelo robusto y replicable permite evolucionar del análisis descriptivo tradicional hacia un enfoque predictivo que fortalezca la toma de decisiones.

Desde la perspectiva académica, el proyecto materializa la aplicación práctica de los conocimientos adquiridos en ciencia de datos, aportando una metodología rigurosa basada en CRISP-DM y en la comparación de algoritmos de aprendizaje supervisado. Con ello, contribuye al campo de la analítica en salud, donde la evidencia sobre predicción del desempeño territorial de las RIAS sigue siendo limitada.

Desde la perspectiva social y de política pública, el modelo predictivo contribuye a la equidad en salud al facilitar la identificación de territorios con bajo cumplimiento, priorizar acciones de mejora y orientar decisiones estratégicas sustentadas en evidencia. De esta manera, la propuesta se inscribe en los principios de calidad, integralidad y equidad que guían las políticas nacionales de salud.

Objetivos

Objetivo General

Evaluar modelos de aprendizaje automático para interpretar y anticipar el comportamiento territorial de los indicadores de las Rutas Integrales de Atención en Salud (RIAS), con el fin de generar información útil que apoye la toma de decisiones y el fortalecimiento de la gestión en salud pública.

Objetivos Específicos

Analizar los datos históricos de los indicadores de las Rutas Integrales de Atención en Salud (RIAS) disponibles en SISPRO, con el propósito de reconocer tendencias de comportamiento y diferencias territoriales que sirvan como base para el modelamiento predictivo.

Aplicar y comparar modelos de aprendizaje supervisado para seleccionar aquellos con mayor pertinencia en la predicción del comportamiento de los indicadores de las Rutas Integrales de Atención en Salud (RIAS) en los diferentes niveles territoriales.

Determinar el desempeño de los modelos predictivos mediante métricas apropiadas, con el fin de identificar cuáles ofrecen mayor confiabilidad y precisión en la estimación del comportamiento de los indicadores de las Rutas Integrales de Atención en Salud (RIAS).

Validar la capacidad predictiva de los modelos seleccionados mediante la elaboración de proyecciones y la revisión de las tendencias esperadas en los indicadores de las Rutas Integrales de Atención en Salud (RIAS), garantizando su utilidad para la planificación y la toma de decisiones.

Marco Conceptual

Rutas Integrales de Atención en Salud

Las RIAS son herramientas fundamentales del SGSSS orientadas a organizar, articular y garantizar el acceso integral y continuo a los servicios de salud. Se estructuran para responder a necesidades específicas del curso de vida y condiciones del entorno, bajo principios de integralidad, gestión del riesgo y participación comunitaria. Su implementación obligatoria está definida en la Resolución 3280 de 2018 *(Ministerio de Salud y Protección Social, 2018)*.

Finalidad de las RIAS

Su objetivo es asegurar la atención efectiva en promoción, prevención, diagnóstico, tratamiento, rehabilitación y cuidados paliativos, considerando infraestructura, perfiles epidemiológicos y capacidad operativa territorial. La gestión del riesgo permite anticipar eventos adversos y orientar intervenciones oportunas *(Ministerio de Salud y Protección Social, 2018)*.

Tipos de Intervenciones

Las RIAS integran intervenciones individuales, colectivas, poblacionales y de gestión. Estas permiten estructurar acciones clínicas, comunitarias, diferenciales y administrativas, constituyendo la base para la definición y análisis de indicadores utilizados en modelos predictivos *(Ministerio de Salud y Protección Social, 2018)*.

Enfoques Orientadores

Su implementación se fundamenta en enfoques basados en derechos, enfoque diferencial, enfoque familiar y comunitario y atención primaria en salud. Estos garantizan pertinencia, equidad y culturalidad, elementos clave para interpretar los resultados y contextualizar predicciones territoriales *(Ministerio de Salud y Protección Social, 2018)*.

Adaptabilidad Territorial y Gestión del Riesgo

La adaptabilidad territorial reconoce las diferencias en capacidad operativa, infraestructura y perfil epidemiológico. La gestión del riesgo permite priorizar acciones y optimizar recursos, lo que se refleja en los indicadores históricos que sirven como insumo para modelos de predicción *(Ministerio de Salud y Protección Social, 2018)*.

Desempeño Territorial en Salud

El desempeño territorial se evalúa mediante indicadores de cobertura, oportunidad, calidad y continuidad. El Monitor RIAS del SISPRO permite identificar brechas entre departamentos y municipios, especialmente en áreas como salud materna, visual, bucal y detección temprana *(Ministerio de Salud y Protección Social, 2024)*.

Indicadores de las RIAS

Los indicadores permiten medir el avance territorial en zonas y grupos poblacionales, facilitando la identificación de tendencias y brechas. Su desagregación anual y territorial permite construir modelos predictivos que anticipen resultados y apoyen decisiones estratégicas en salud pública.

Analítica de Datos en Salud

La analítica transforma datos clínicos, epidemiológicos y administrativos en información útil mediante minería de datos, aprendizaje automático y métodos computacionales. Permite reconocer patrones, anticipar comportamientos y fortalecer la gestión basada en evidencia *(J Med Internet Res, 2021)*.

Aprendizaje Supervisado y Regresión

El aprendizaje supervisado utiliza datos históricos con salidas conocidas para entrenar modelos. La regresión permite estimar valores continuos y predecir el desempeño territorial de

los indicadores RIAS en función del tiempo y la geografía *(Hastie, Tibshirani & Friedman, 2017; Breiman, 2001)*.

Regresión con Árboles de Decisión

Los árboles de decisión permiten predecir valores continuos a partir de divisiones lógicas de los datos. Los ensamblajes como Random Forest o Gradient Boosting aumentan la precisión y reducen el sobreajuste, siendo ampliamente utilizados en predicción en salud *(Breiman, 2001; Hastie, Tibshirani & Friedman, 2017; Katuwal & Suganthan, 2018)*.

Modelos Predictivos en Salud

Los modelos predictivos permiten anticipar el desempeño de los territorios, generar alertas tempranas y apoyar intervenciones basadas en datos. Su valor radica en fortalecer la planeación y la asignación eficiente de recursos *(J Med Internet Res, 2021; Moran Pizarro et al., 2023)*.

Métricas de Evaluación de Regresión

Las métricas como MSE, MAE y R^2 permiten validar la precisión, estabilidad y utilidad de un modelo predictivo, asegurando su aplicabilidad real en la gestión en salud pública *(Barreto, 2024; Gaitán, 2022; Bonaccorso, 2018)*.

Marco Teórico

Marco Normativo de las RIAS

La Resolución 3280 de 2018 constituye el principal referente normativo para la organización y ejecución de las RIAS dentro del SGSSS. Establece un enfoque centrado en la persona, el curso de vida y la gestión del riesgo, definiendo intervenciones individuales, colectivas, poblacionales y de gestión, así como los principios de adaptabilidad territorial, equidad, calidad y participación comunitaria. Esta normativa delimita las responsabilidades de administradores, territorios y prestadores, y define los indicadores que alimentan herramientas de monitoreo como el Monitor RIAS, insumo central para este proyecto *(Ministerio de Salud y Protección Social, 2018)*.

Analítica de Datos en Salud y Toma de Decisiones

La analítica de datos se ha consolidado como un componente crucial para la planeación y el monitoreo en salud, permitiendo procesar grandes volúmenes de información clínica, administrativa y poblacional. El uso de técnicas estadísticas y de aprendizaje automático facilita identificar patrones, predecir comportamientos y evaluar el desempeño institucional y territorial, contribuyendo a decisiones basadas en evidencia *(J Med Internet Res, 2021)*. En el contexto de las RIAS, esta perspectiva metodológica respalda la necesidad de transformar datos históricos en insumos predictivos capaces de orientar la planificación territorial.

Aportes Metodológicos para el Análisis Territorial

La literatura muestra que los modelos predictivos basados en aprendizaje automático permiten interpretar datos administrativos, epidemiológicos y operativos, anticipando niveles de cumplimiento y apoyando la gestión del riesgo. Para este proyecto, estos referentes validan el uso de modelos supervisados y técnicas de regresión como mecanismos para estimar el

comportamiento futuro de los indicadores, integrando resultados en procesos de planeación y mejora continua del desempeño territorial *(Andrade-Girón et al., 2023)*.

Enfoques Institucionales, Desigualdad y Brechas Territoriales

Estudios recientes evidencian la fragmentación de la promoción de la salud en Colombia, orientada mayormente a acciones individuales y desconectada de enfoques comunitarios y determinantes sociales. Esta desarticulación genera brechas entre el diseño normativo y la implementación real, afectando el desempeño territorial de las RIAS *(Mosquera-Becerra et al., 2023)*. Estos hallazgos justifican incorporar en el modelamiento variables que reflejen diferencias operativas, capacidad institucional, continuidad de intervenciones y coherencia normativa-operativa, las cuales influyen directamente en el nivel de cumplimiento de los territorios.

Aprendizaje Automático y Comparación de Algoritmos

Investigaciones aplicadas en salud pública demuestran que comparar distintos algoritmos supervisados —como regresión logística, árboles de decisión, Random Forest, SVM y Naive Bayes— mejora la selección del modelo más adecuado según el tipo de dato y objetivo analítico. Asimismo, se destaca la importancia del balanceo de datos, validación cruzada y prevención del sobreajuste, garantizando predicciones confiables y replicables *(Andrade-Girón et al., 2023)*. Estos elementos fundamentan la estrategia metodológica adoptada, basada en la evaluación comparativa de modelos para identificar aquellos con mayor capacidad predictiva.

Modelos Multivariables y Dimensión Espacio -Temporal

Estudios recientes enfatizan la importancia de modelos multivariables y no lineales —como redes neuronales, SVM, Random Forest y ensamblajes de árboles— en contextos epidemiológicos complejos, donde múltiples factores interactúan de manera simultánea *(Polo-*

Triana et al., 2023). La consideración de estructuras espacio-temporales fortalece la capacidad del modelo para capturar cambios por municipio y por año, lo que resulta especialmente pertinente para analizar el desempeño territorial de las RIAS.

Procesamiento Multivariable de Datos Territoriales

La literatura destaca que la calidad y representatividad de los datos son más determinantes que la técnica predictiva utilizada. Estrategias como limpieza, normalización, selección de atributos y segmentación temporal son claves para construir bases territoriales robustas. En estudios aplicados, algoritmos como Random Forest han mostrado eficiencia, estabilidad y facilidad de interpretación para análisis institucionales y territoriales *(Muñoz-Ordóñez et al., 2020)*.

Aplicación de IA en Datos Agregados en Salud

El uso de IA en datos agregados permite anticipar resultados, generar indicadores automatizados y mejorar el monitoreo institucional. Su aplicación en unidades de análisis agregadas —como municipios o rutas de atención— demuestra la viabilidad de construir sistemas predictivos para la gestión territorial en salud, apoyados en métricas robustas como MAE, MSE y R^2 *(Singla et al., 2024)*.

Analítica Territorial Aplicada a Cobertura y Desempeño Institucional

Modelos basados en integración de datos administrativos, depuración, análisis por cohortes y visualización dinámica permiten anticipar variaciones en cobertura, desempeño institucional y continuidad del aseguramiento. Este enfoque respalda la viabilidad de construir un modelo predictivo territorial centrado en indicadores de las RIAS *(Dorado Daza, 2023)*.

Modelos Eficientes en Contextos de Baja Capacidad Tecnológica

Investigaciones muestran que modelos simples pero bien optimizados como SVM, KNN y árboles de decisión pueden alcanzar altos niveles de precisión operando incluso en entornos de bajo consumo computacional, lo que los hace aplicables a territorios con capacidades técnicas limitadas *(Niño Rondón, 2024)*.

Procesamiento Avanzado y Detección de Patrones

Estrategias como reducción de dimensionalidad, extracción de características temporales y normalización de datos permiten construir matrices multivariables para predicción territorial. Estas técnicas, junto con modelos multivariables como redes neuronales o regresión no lineal, favorecen la detección de patrones territoriales y la generación de alertas tempranas *(Barahona García & Jaramillo Marín, 2022)*.

Variables Socioeconómicas y Ambientales como Predictoras

Se ha demostrado que datos socioeconómicos, demográficos y ambientales pueden usarse con éxito para construir modelos predictivos robustos en salud pública, sin necesidad de información clínica detallada. Esto valida el uso de datos administrativos y poblacionales como insumos para anticipar el desempeño territorial en la implementación de las RIAS *(Mejía et al., 2023)*.

Diseño Metodológico Integral y Selección de Algoritmos

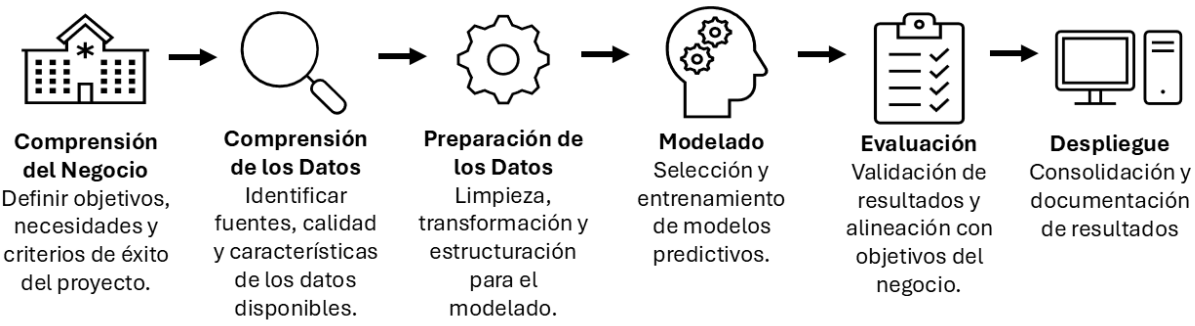
Diversos estudios refuerzan la necesidad de pipelines completos que incluyan extracción, limpieza, transformación, modelamiento comparativo, validación cruzada y evaluación mediante métricas robustas como sensibilidad, especificidad y AUC-ROC. Estas buenas prácticas guían el diseño de un modelo predictivo sostenible, escalable y ajustado a las particularidades territoriales *(López Rodríguez & Isaza Vides, 2023)*.

Metodología

La metodología del proyecto se fundamenta en el estándar CRISP–DM, reconocido por su carácter estructurado, iterativo y adaptable para el desarrollo de modelos predictivos basados en analítica avanzada. Este enfoque guía las seis fases del proceso mediante las cuales se construyó, evaluó y consolidó el modelo de aprendizaje automático diseñado para predecir el desempeño territorial de los indicadores de las Rutas Integrales de Atención en Salud (RIAS).

Figura 1

Fases del Proceso CRISP-DM para el Desarrollo de Modelos Analíticos

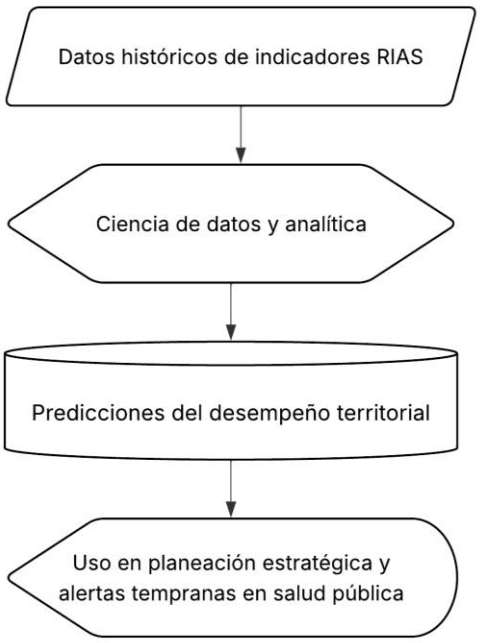


Comprensión del Negocio

El proyecto tiene como propósito desarrollar un modelo predictivo capaz de anticipar el nivel de cumplimiento de los indicadores RIAS en los niveles nacional, departamental, municipal y asegurador. El modelo utiliza datos históricos del SISPRO con periodicidad anual y se evalúa mediante métricas como MAE, MSE, RMSE y R^2. Esta fase permitió definir el alcance, los requerimientos analíticos y el flujo conceptual del problema, incluyendo la relación entre los datos de entrada, el procesamiento y las predicciones destinadas a apoyar la planificación estratégica en salud pública.

Figura 2

Diagrama Conceptual del Problema: Predicción del Desempeño Territorial



Comprensión de los Datos

Se realizó un análisis exploratorio (EDA) para evaluar la estructura, completitud y coherencia de los registros del Monitor RIAS. Se verificaron numeradores, denominadores y resultados porcentuales, y se validó la integridad territorial mediante la codificación DIVIPOLA y los identificadores de aseguradores. Se aplicaron técnicas estadísticas y visuales para identificar patrones de comportamiento, correlaciones, grupos de desempeño y tendencias temporales, permitiendo reconocer áreas con rezagos estructurales o avances consolidados.

Preparación de los Datos

La preparación de datos incluyó tareas de depuración, transformación y estandarización necesarias para el uso adecuado de modelos supervisados. Se eliminaron registros incompletos,

se normalizaron escalas, se aplicó Label Encoding a las variables categóricas y se excluyó el año 2025 por no corresponder a un corte anual cerrado.

La variable objetivo seleccionada fue el valor porcentual del indicador (Ind), mientras que las predictoras incluyeron año, identificador del indicador y variables territoriales/institucionales. Posteriormente, los datos fueron normalizados mediante StandardScaler para uniformar magnitudes entre variables.

De acuerdo con las buenas prácticas de modelamiento supervisado, el conjunto final depurado se dividió en 70 % para entrenamiento y 30 % para prueba, garantizando un adecuado balance entre aprendizaje del modelo y capacidad de generalización, manteniendo la coherencia temporal y territorial entre niveles.

Modelado

Se entrenaron diversos algoritmos de aprendizaje supervisado, desde técnicas lineales hasta modelos no lineales y de ensamblaje. Entre ellos: Linear Regression, Ridge, Lasso, Elastic Net, KNN, SVR, Decision Tree, Random Forest, Extra Trees, Gradient Boosting, MLP, XGBoost, LightGBM y CatBoost.

El análisis comparativo de los niveles nacional y departamental permitió seleccionar los seis modelos con mejor desempeño global —XGBoost, LightGBM, Random Forest, Decision Tree, CatBoost y Gradient Boosting— los cuales se aplicaron posteriormente a los niveles municipal e institucional debido a su mayor estabilidad, precisión y eficiencia computacional.

Evaluación

Los modelos fueron evaluados mediante métricas de regresión: R^2, MAE, MSE y RMSE, analizando la capacidad predictiva, la magnitud del error y la variabilidad explicada por cada algoritmo. Las métricas fueron presentadas en tablas comparativas y visualizadas mediante

gráficos normalizados para facilitar la interpretación del comportamiento de los modelos en cada nivel territorial.

Despliegue

Se consolidaron los modelos, parámetros, métricas y scripts en un entorno reproducible basado en Jupyter Notebook y archivos .csv, organizado según las fases CRISP–DM. Este repositorio analítico permite reentrenar los modelos con nuevas vigencias del Monitor RIAS, actualizar predicciones y recalibrar métricas de desempeño.

Cronograma de Actividades

Tabla 1

Cronograma de Trabajo del Proyecto Según la Metodología CRISP-DM

Actividad	Mes inicio	Mes final	Porcentaje
Comprensión del negocio	1	1	10%
Comprensión de los datos	1	2	15%
Preparación de los datos	2	2	20%
Modelado	2	3	20%
Evaluación	3	3	15%
Despliegue (prototipos y consolidación)	3	3	20%
Total			100%

Presupuesto del Proyecto

Tabla 2

Presupuesto del Proyecto

Tipo de recurso	Descripción	Presupuesto
Equipo humano	Actividades desarrolladas directamente por el estudiante bajo la tutoría académica asignada. No implica contratación ni costos adicionales.	$ 0
Equipos y software	Uso de computador personal, software de código abierto (Python) y recursos institucionales provistos por la universidad.	$ 0
Viajes y salidas de campo	No se requieren desplazamientos físicos; el proyecto se desarrolla con datos disponibles en línea.	$ 0
Materiales y suministros	No se contemplan insumos físicos; el trabajo se realiza en formato digital.	$ 0
Bibliografía	Se utilizarán bases de datos y literatura académica de acceso abierto y recursos bibliográficos de la universidad.	$ 0
Total		$ 0

Resultados o Productos Esperados

Tabla 3

Resultados o Productos Esperados

Resultado o producto esperado	Indicador	Beneficiario
Base de datos consolidada y depurada a partir de fuentes oficiales	Número de bases integradas y depuradas sobre el total de fuentes oficiales identificadas	Estudiante y Universidad
Aplicación modelo predictivo funcional para estimar el desempeño territorial en la implementación de las RIAS	Número de métricas de desempeño que alcanzan o superan el umbral definido sobre el total de métricas establecidas	Estudiante y Universidad
Documento técnico y académico con metodología, resultados y conclusiones	Número de entregables presentados y validados sobre el total de entregables exigidos por la universidad	Universidad

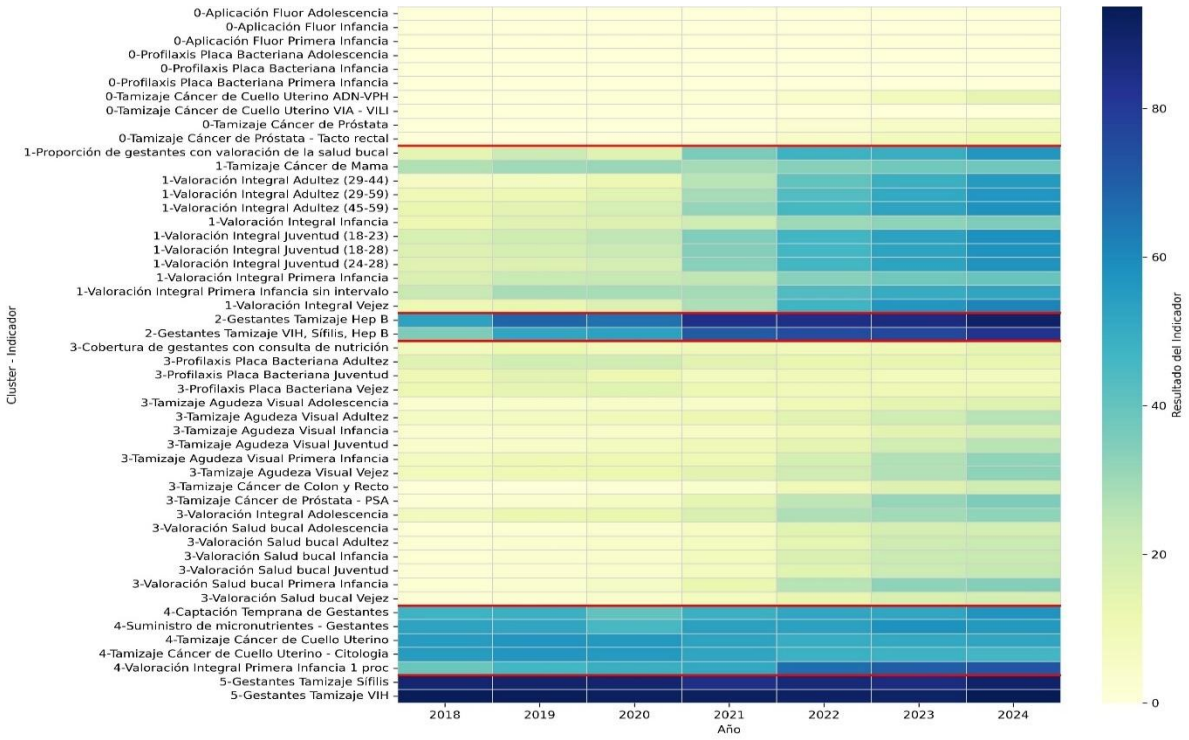
Resultados

Análisis de los Datos Históricos y Tendencias Territoriales

El análisis de los indicadores del Monitor RIAS evidencia patrones definidos en el comportamiento temporal y territorial del cumplimiento. Tal como se observa en la Figura 3, los indicadores se agrupan en seis trayectorias que reflejan distintos niveles de avance: algunos presentan cumplimiento alto y sostenido, mientras que otros muestran bajo desempeño persistente o alta variabilidad interanual, lo que sugiere inestabilidad operativa o inconsistencias en la capacidad de reporte.

Figura 3

Indicador Agrupado en Seis Clusters de Cumplimiento

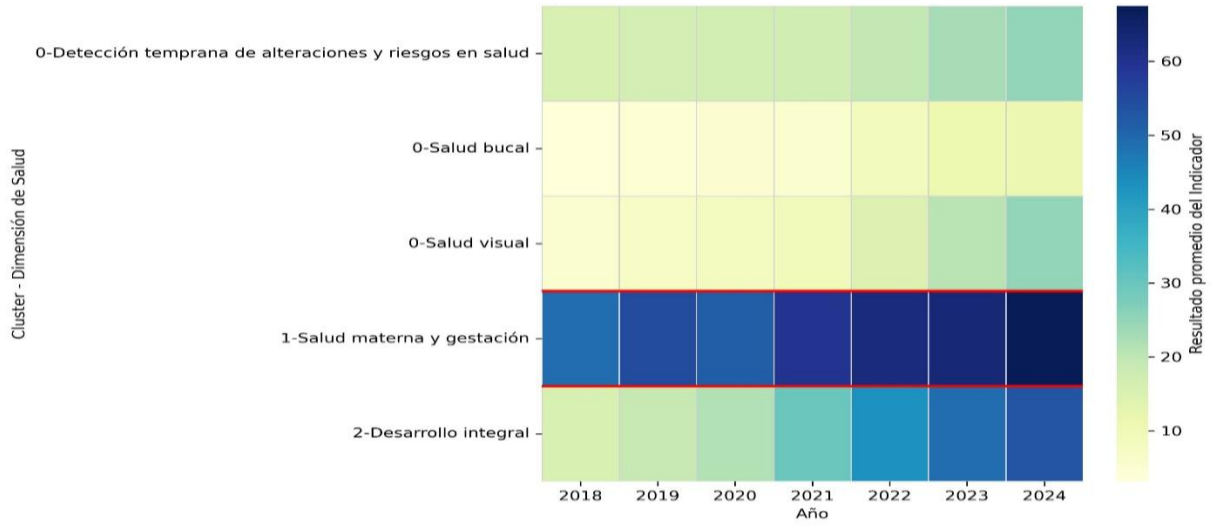


La Figura 4 muestra diferencias relevantes entre las agrupaciones de salud. La salud materna y gestación mantiene los niveles más altos y estables de cumplimiento, mientras que

salud bucal, salud visual y detección temprana registran valores bajos y escasos avances. En contraste, la dimensión de desarrollo integral presenta una tendencia creciente, reflejando un fortalecimiento progresivo de las acciones preventivas en el curso de vida.

Figura 4

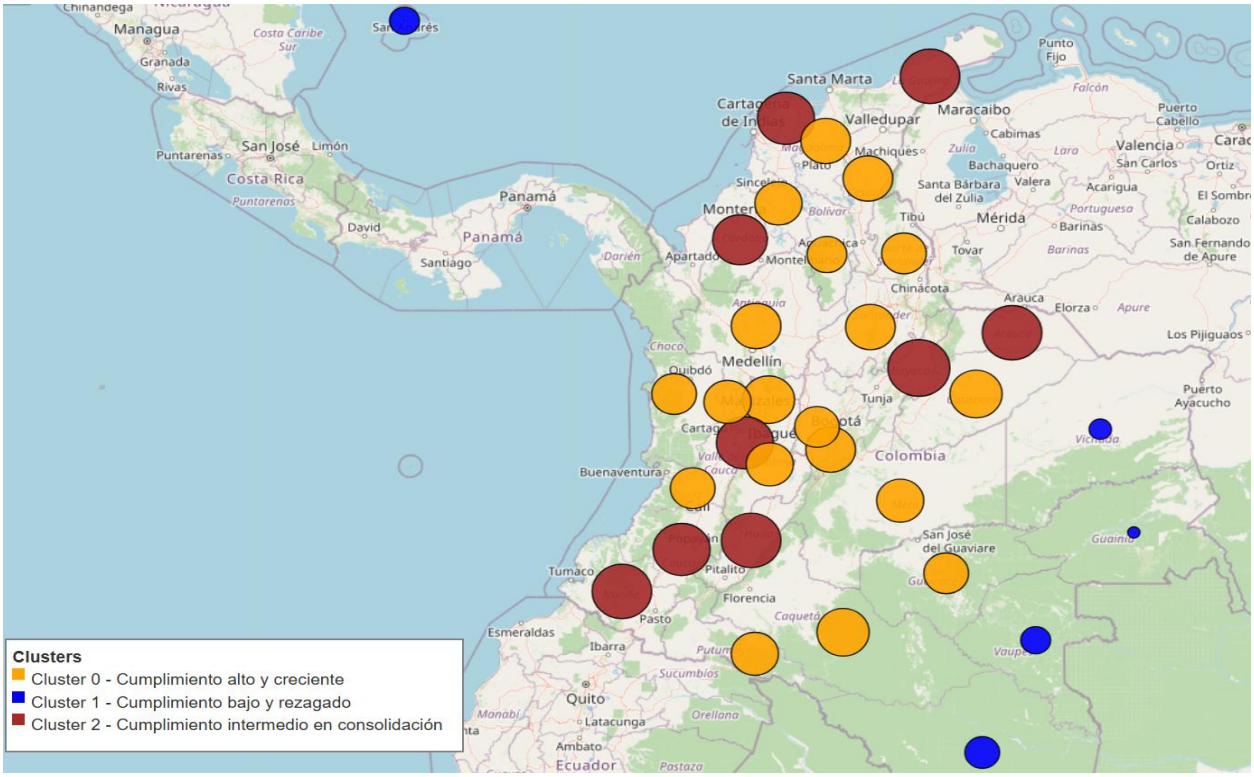
Dimensiones de Salud y Clústeres de Cumplimiento



Desde la perspectiva territorial, los resultados por departamento representados en la Figura 5 revelan contrastes marcados en el desempeño. Los territorios del centro y noroccidente conforman el grupo de mayor cumplimiento y progreso sostenido, mientras que las regiones del sur y oriente se ubican en los clústeres de bajo desempeño, con rezagos persistentes. Entre estos extremos se encuentran departamentos con mejoras graduales, que consolidan trayectorias intermedias.

Figura 5

Tendencias Territoriales por Clúster



En conjunto, los hallazgos indican una mejora general del cumplimiento, pero también una heterogeneidad significativa entre dimensiones e inequidades territoriales. Las líneas maternas se destacan como las más consolidadas, mientras que las acciones preventivas siguen siendo las más rezagadas, lo que resalta la necesidad de intervenciones diferenciadas para fortalecer la equidad en la implementación territorial de las RIAS.

Aplicación y Comparación de Modelos de Aprendizaje Supervisado

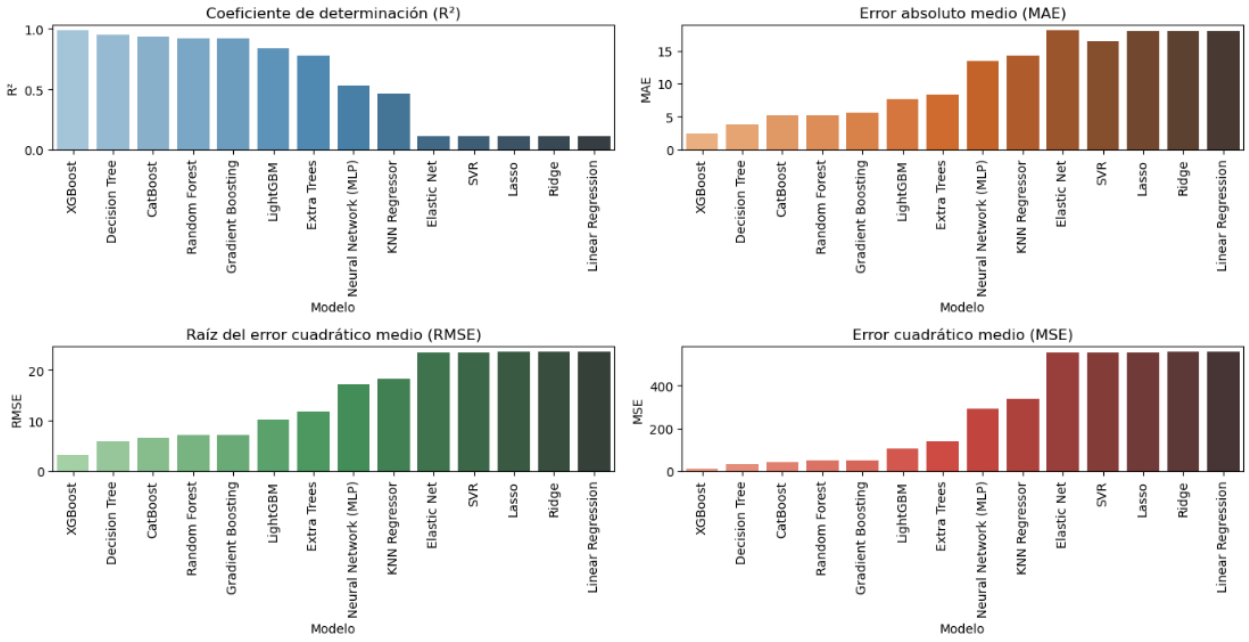
La aplicación de los modelos de aprendizaje supervisado permitió evaluar el desempeño predictivo en los cuatro niveles de análisis del Monitor RIAS: nacional, departamental, municipal e institucional. En todos los casos, la comparación se realizó bajo condiciones

homogéneas, empleando idénticas bases de entrenamiento y prueba, así como las mismas métricas de evaluación.

En el nivel nacional, la Figura 6 evidencia diferencias claras entre los modelos lineales y los métodos de ensamblaje. Algoritmos como XGBoost, Decision Tree, CatBoost y Random Forest registran los valores más altos de R^2 y los menores errores, demostrando una mejor capacidad de ajuste frente a la variabilidad temporal de los indicadores. Los modelos lineales, por el contrario, presentan un rendimiento limitado en la captura de relaciones no lineales.

Figura 6

Evaluación Nacional por Modelo

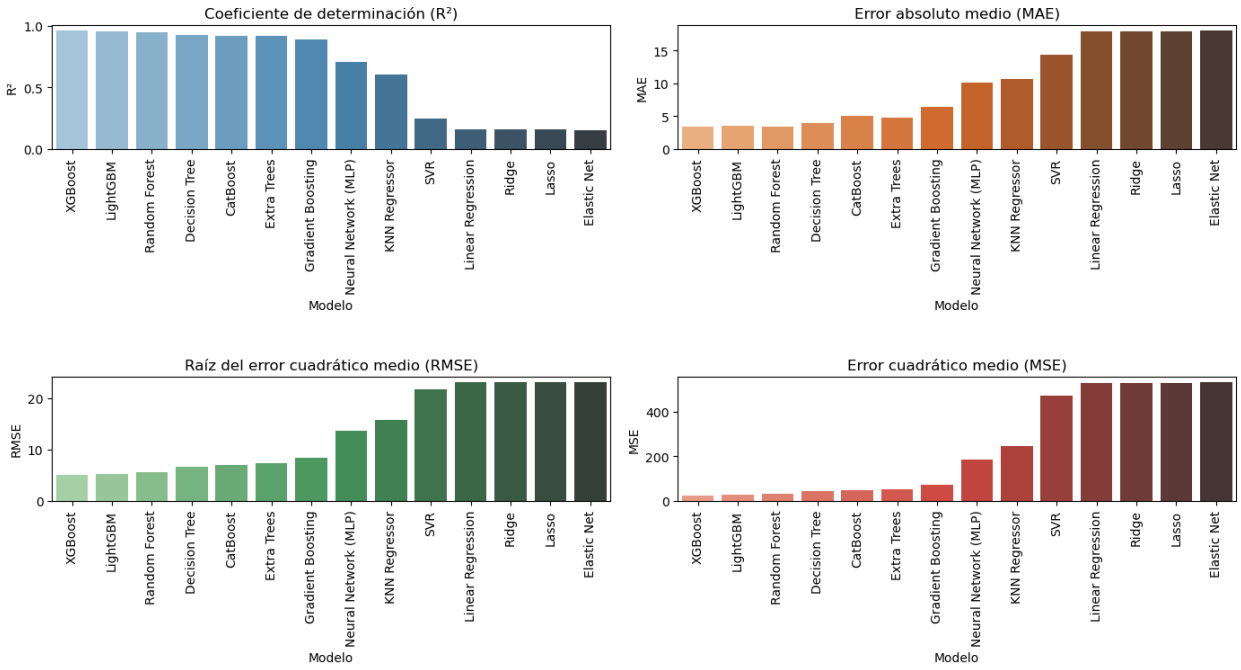


En el nivel departamental, la Figura 7 confirma nuevamente la superioridad de los modelos no lineales. La mayor densidad y variabilidad de los datos departamentales favorece el desempeño de XGBoost, LightGBM, Random Forest y Decision Tree, que alcanzan métricas

más altas de R^2 y errores reducidos. Los métodos lineales pierden precisión en este nivel por su menor capacidad para representar diferencias territoriales.

Figura 7

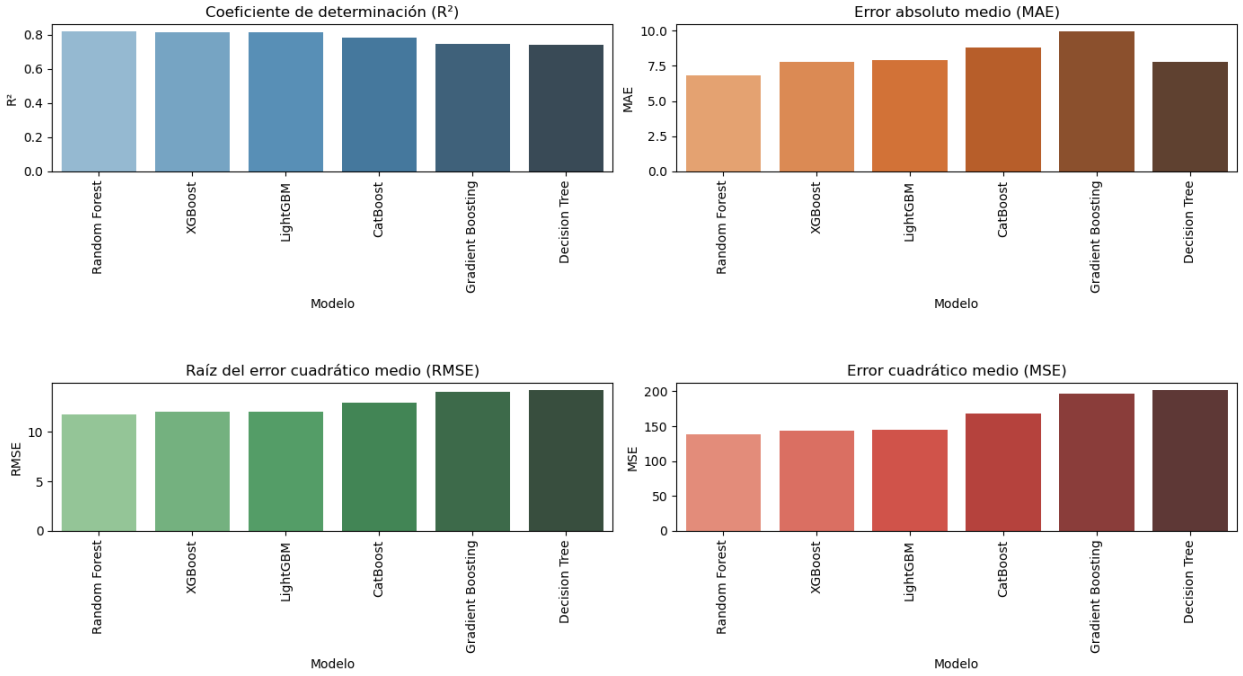
Evaluación de Modelos por Departamento



En el nivel municipal, la Figura 8 muestra una menor diferencia entre los modelos, aunque Random Forest, XGBoost y LightGBM continúan encabezando los mejores resultados. La alta heterogeneidad municipal disminuye la precisión global, pero los modelos de ensamblaje mantienen ventajas en estabilidad y ajuste, con errores moderados y coeficientes de determinación superiores respecto a las alternativas lineales.

Figura 8

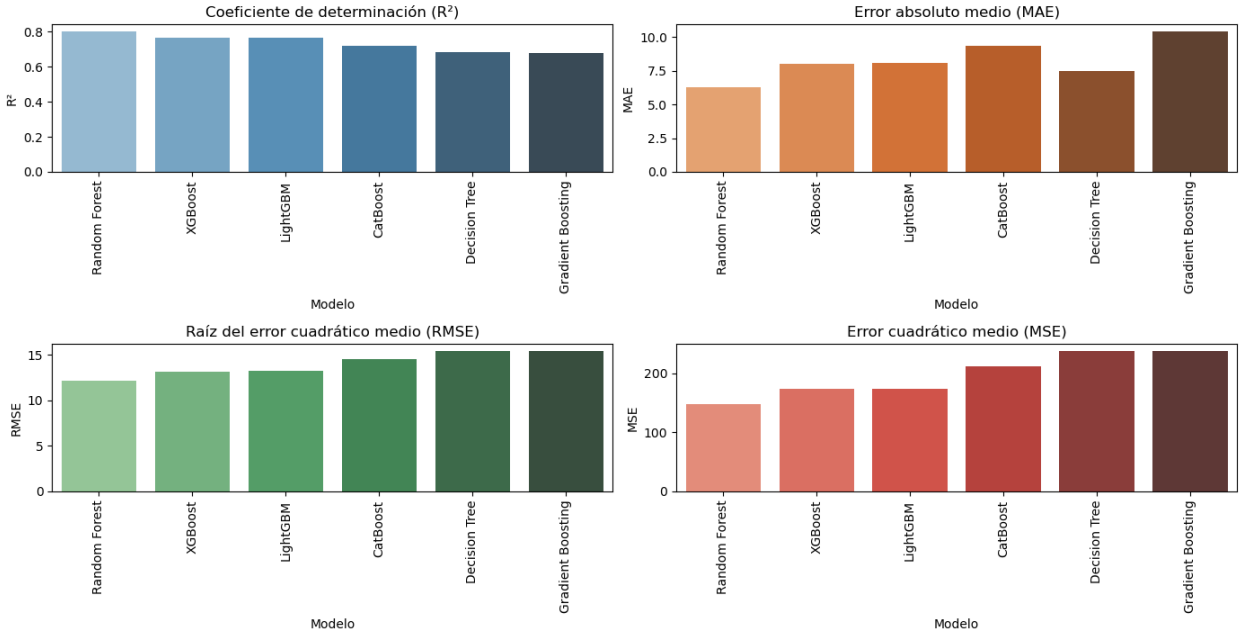
Evaluación de Modelos por Municipio



Finalmente, en el nivel institucional, la Figura 9 revela un comportamiento altamente consistente entre los modelos. Random Forest, XGBoost y LightGBM logran nuevamente los valores más altos de R^2 y los menores errores, reflejando un desempeño robusto en la predicción por asegurador. CatBoost y Decision Tree también presentan un rendimiento cercano, evidenciando estabilidad y capacidad de generalización.

Figura 9

Evaluación de Modelos por Aseguradora



En conjunto, los cuatro niveles muestran una tendencia uniforme: los modelos basados en árboles y gradiente (XGBoost, LightGBM, Random Forest, CatBoost, Decision Tree y Gradient Boosting) ofrecen el mejor equilibrio entre ajuste, precisión y estabilidad, logrando capturar relaciones no lineales y variaciones territoriales de los indicadores RIAS con mayor eficacia que los modelos lineales.

Evaluación del Desempeño de los Modelos Predictivos

La evaluación cuantitativa de los modelos supervisados permitió analizar el ajuste, la precisión y la estabilidad de las predicciones en los cuatro niveles de desagregación del Monitor RIAS. Para ello se emplearon las métricas R^2, MAE, MSE y RMSE, lo cual facilitó comparar el rendimiento de los algoritmos ante distintos grados de variabilidad territorial.

En el nivel nacional, la Tabla 4 muestra una superioridad marcada de los modelos basados en árboles y gradiente. XGBoost obtuvo el mejor desempeño ($R^2 = 0,98$, RMSE = 3,20), seguido por Decision Tree y CatBoost. Los modelos lineales presentaron un rendimiento significativamente menor ($R^2 = 0,11$), confirmando su incapacidad para capturar relaciones no lineales y la complejidad temporal del indicador.

Tabla 4

Comparativo de Modelos Supervisados de Regresión – Nivel Nacional

Modelo	R^2	MAE	MSE	RMSE
XGBoost	0,98	2,42	10,23	3,20
Decision Tree	0,95	3,76	33,86	5,82
CatBoost	0,93	5,22	43,19	6,57
Random Forest	0,92	5,13	51,68	7,19
Gradient Boosting	0,92	5,60	51,71	7,19
LightGBM	0,83	7,70	103,65	10,18
Extra Trees	0,78	8,33	138,62	11,77
Neural Network	0,53	13,45	292,86	17,11
KNN Regressor	0,46	14,26	335,92	18,33
Elastic Net	0,11	18,04	553,92	23,54
SVR	0,11	16,50	553,95	23,54
Lasso	0,11	17,97	554,97	23,56
Ridge	0,11	17,95	555,30	23,56
Linear Regression	0,11	17,95	555,42	23,57

En el nivel departamental, la Tabla 5 evidencia que XGBoost y LightGBM alcanzan los valores más altos de R^2 (0,96) y errores reducidos, manteniendo una alta consistencia frente a la variabilidad territorial. Los modelos lineales y SVR muestran un deterioro notable en su capacidad predictiva, registrando errores elevados y menor ajuste frente a las diferencias departamentales.

Tabla 5

Comparativo de Modelos Supervisados de Regresión – Nivel Departamental

Modelo	R^2	MAE	MSE	RMSE
XGBoost	0,96	3,37	25,48	5,05
LightGBM	0,96	3,50	27,21	5,22
Random Forest	0,95	3,45	31,73	5,63
Decision Tree	0,93	3,93	44,83	6,70
CatBoost	0,92	5,05	48,87	6,99
Extra Trees	0,92	4,70	52,52	7,25
Gradient Boosting	0,89	6,39	70,80	8,41
Neural Network	0,70	10,11	186,07	13,64
KNN Regressor	0,61	10,65	246,94	15,71
SVR	0,25	14,40	470,82	21,70
Linear Regression	0,16	17,89	529,45	23,01
Ridge	0,16	17,89	529,45	23,01
Lasso	0,16	17,91	529,55	23,01
Elastic Net	0,16	17,97	530,06	23,02

En el nivel municipal, la Tabla 6 indica una reducción general del ajuste debido a la heterogeneidad local. Aun así, Random Forest ($R^2 = 0,82$) lidera el desempeño, seguido de XGBoost y LightGBM ($R^2 = 0,81$). Aunque los errores son más altos que en niveles superiores, los modelos de ensamblaje mantienen estabilidad y logran reproducir tendencias municipales con aceptable precisión.

Tabla 6

Comparativo de Modelos Supervisados de Regresión – Nivel Municipal

Modelo	R^2	MAE	MSE	RMSE
Random Forest	0,82	6,81	137,95	11,75
XGBoost	0,81	7,78	143,33	11,97
LightGBM	0,81	7,88	144,66	12,03
CatBoost	0,78	8,79	168,01	12,96
Gradient Boosting	0,75	9,97	196,13	14,00
Decision Tree	0,74	7,79	202,32	14,22

En el nivel asegurador, la Tabla 7 confirma nuevamente a Random Forest como el modelo con mejor desempeño ($R^2 = 0,80$, RMSE = 12,14), seguido por XGBoost y LightGBM, que registran métricas cercanas. CatBoost, Decision Tree y Gradient Boosting completan el grupo de alto rendimiento, mientras que los modelos lineales y redes neuronales muestran un desempeño inferior y poco estable.

Tabla 7

Comparativo de Modelos Supervisados de Regresión – Nivel Asegurador

Modelo	R^2	MAE	MSE	RMSE
Random Forest	0,80	6,25	147,31	12,14
XGBoost	0,77	8,01	173,54	13,17
LightGBM	0,77	8,07	174,16	13,20
CatBoost	0,72	9,35	211,41	14,54
Decision Tree	0,68	7,45	237,95	15,43
Gradient Boosting	0,68	10,44	238,63	15,45

En conjunto, los cuatro niveles analizados muestran una conclusión consistente: los modelos basados en árboles y gradiente (Random Forest, XGBoost, LightGBM, CatBoost, Decision Tree y Gradient Boosting) ofrecen el mejor equilibrio entre ajuste, precisión y robustez, manteniendo un rendimiento superior frente a modelos lineales incluso en escenarios con alta variabilidad territorial. Estos resultados respaldan su elección para las fases finales de validación y proyección predictiva del modelo.

Resultados de la Validación y Proyección

La validación de los modelos seleccionados tuvo como objetivo comprobar su capacidad de generalización y la estabilidad de sus predicciones al aplicarse sobre valores proyectados del Monitor RIAS. Para este proceso se eligieron los tres algoritmos con mejor desempeño en la fase anterior —Random Forest, XGBoost y LightGBM—, los cuales fueron integrados mediante un esquema de votación ponderada, asignando a cada modelo un peso proporcional a su coeficiente de determinación (R^2). La predicción final resultó del promedio ponderado de los valores

estimados, combinando la robustez de Random Forest, la eficiencia de XGBoost y la capacidad de generalización de LightGBM.

La Figura 10 muestra el mapa de calor correspondiente a la proyección combinada de los indicadores del Monitor RIAS entre 2018 y 2027. En la visualización se distinguen patrones de evolución diferenciados por grupos de indicadores: las acciones maternas y de gestación mantienen niveles altos y estables de cumplimiento, representados en tonalidades azul oscuro; las valoraciones integrales del curso de vida exhiben una tendencia creciente, indicando una mejora sostenida en la cobertura poblacional; y las dimensiones de salud bucal, salud visual y detección temprana de cáncer presentan valores bajos pero con incrementos graduales hacia el final del horizonte proyectado, lo que refleja avances progresivos aunque aún insuficientes respecto a las metas de integralidad.

Figura 10

Proyección de Resultados del Monitor RIAS (2018–2027)

En conjunto, los resultados confirman que la estrategia de votación ponderada mejora la estabilidad y precisión de las predicciones, reduciendo la sensibilidad a la variabilidad de un único algoritmo. La coherencia entre los patrones históricos y las trayectorias proyectadas valida la aplicabilidad del modelo combinado como herramienta estratégica para la planificación, el seguimiento y la priorización de intervenciones territoriales en salud pública.

Conclusiones

El análisis histórico y la clusterización de los indicadores mostraron patrones estructurales robustos y estadísticamente consistentes en el desempeño territorial de las Rutas Integrales de Atención en Salud. La clasificación de los indicadores en seis trayectorias diferenciadas —resultado del análisis multianual y de técnicas de agrupamiento— permitió identificar comportamientos de alto, medio y bajo cumplimiento respaldados por medidas de tendencia central, dispersión y variabilidad interanual. Las figuras comparativas evidenciaron diferencias sostenidas entre dimensiones: mientras los indicadores maternos alcanzaron valores superiores al 80–90 % de cumplimiento en la mayor parte del periodo, los indicadores de salud bucal, salud visual y detección temprana permanecieron en rangos inferiores al 40 %, confirmando mediante evidencia estadística la existencia de brechas persistentes y estructurales entre territorios y líneas de acción.

La evaluación comparativa de algoritmos demostró, con base en métricas de desempeño como R^2, MAE, MSE y RMSE, que los modelos de ensamblaje (XGBoost, LightGBM y Random Forest) superan ampliamente a los enfoques lineales y a los métodos tradicionales. Estos modelos alcanzaron coeficientes de determinación entre 0,80 y 0,98, errores absolutos medios entre 2 y 8 puntos porcentuales según el nivel, y residuos centrados y simétricos, lo que valida estadísticamente su capacidad para capturar relaciones no lineales entre variables temporales, territoriales e institucionales. La consistencia de estas métricas en los niveles nacional, departamental, municipal e institucional constituye un aporte metodológico sólido, respaldado por análisis cuantitativos reproducibles y gráficas de ajuste que confirman la superioridad de estos modelos en escenarios de alta variabilidad.

La proyección de resultados a través del modelo combinado por votación ponderada —integrando el desempeño de los tres modelos con mayores valores de R^2— permitió anticipar comportamientos futuros con estabilidad y coherencia estadística. El heatmap proyectado para 2025–2027 evidenció tendencias de continuidad en los indicadores maternos y mejoras moderadas en salud visual, bucal y detección temprana, apoyadas en valores predichos consistentes entre modelos y sin presencia de sesgos sistemáticos en los residuos. Este resultado valida la capacidad del enfoque predictivo para generar escenarios verificables, útiles para anticipar brechas y orientar la planificación territorial desde un fundamento cuantitativo robusto y sustentado en modelos con alta capacidad explicativa.

Recomendaciones

A partir de los resultados obtenidos, se recomienda implementar un sistema institucional de monitoreo predictivo que permita actualizar periódicamente los datos del Monitor RIAS, recalibrar los modelos y validar la estabilidad del desempeño territorial en cada vigencia. Este sistema debe integrarse en los procesos de planeación estratégica para garantizar decisiones oportunas basadas en evidencia.

Es pertinente dirigir acciones focalizadas hacia los territorios ubicados de manera consistente en los clusters de bajo desempeño, especialmente en departamentos y municipios del sur y oriente del país. La evidencia sugiere que las brechas territoriales no evolucionan espontáneamente hacia mejores niveles, por lo que se requiere una intervención diferenciada y sostenida, alineada con las dimensiones de mayor rezago.

Este estudio presenta algunas limitaciones inherentes al uso exclusivo de datos históricos del Monitor RIAS, lo que restringe la capacidad del modelo para capturar factores estructurales y operativos que influyen en el desempeño territorial. En consecuencia, se recomienda ampliar el modelo incorporando variables adicionales que permitan mejorar la precisión predictiva en niveles locales. Variables como la disponibilidad de talento humano, la capacidad instalada, las condiciones socioeconómicas y la inversión pública pueden aumentar el poder explicativo del modelo y ofrecer una comprensión más completa de las dinámicas municipales e institucionales.

Finalmente, resulta conveniente desarrollar tableros interactivos que integren datos históricos, predicciones y alertas tempranas, facilitando la interpretación operativa por parte de los equipos técnicos y territoriales. Adicionalmente, se sugiere realizar análisis de sensibilidad y simulación de escenarios alternativos para evaluar el impacto de intervenciones específicas y fortalecer la toma de decisiones basada en proyección y gestión anticipada del riesgo.

Referencias Bibliográficas

ADRES. (2023). *Informe de Gestión 2023 – 2024*. https://www.adres.gov.co/rendicion-de-cuentas/Informe%20de%20Gestion/Informe_de_Gestion_2023_2024.pdf

Andrade-Girón, D., Carreño-Cisneros, E., Mejía-Domínguez, C., Marín-Rodriguez, W., & Villarreal-Torres, H. (2023). Comparación de algoritmos *machine learning* para la predicción de pacientes con sospecha de COVID-19. *Salud, Ciencia y Tecnología, 3*, 1–7. https://doi.org/10.56294/saludcyt2023336

Arnau-Sabatés, L., & Sala Roca, J. (2020). *La revisión de la literatura científica: pautas, procedimientos y criterios de calidad*. https://ddd.uab.cat/record/222109

Barahona García, M. A., & Jaramillo Marín, L. V. (2022). *Regresión y caracterización de señales electroencefalográficas empleando técnicas de inteligencia artificial* [Trabajo de grado, Universidad ECCI]. Repositorio Institucional ECCI. https://repositorio.ecci.edu.co/handle/001/3002

Bonaccorso, G. (2018). *Machine Learning Algorithms: Popular Algorithms for Data Science and Machine Learning* (2nd ed.). Packt Publishing. https://bibliotecavirtual.unad.edu.co/login?url=https://search.ebscohost.com/login.aspx?direct=true&db=nlebk&AN=1881497

Dorado Daza, D. J. (2023). *Modelo de analítica de datos para apoyar la cobertura del aseguramiento en salud en el departamento de Cundinamarca* [Tesis de maestría, Pontificia Universidad Javeriana Cali]. Vitela. https://vitela.javerianacali.edu.co/handle/11522/2023

Instituto Nacional de Salud. (2023). *Informe tablero de problemas, Colombia, 2023*. https://www.ins.gov.co/BibliotecaDigital/informe-tp-2023.pdf

Jansen, S. (2018). *Hands-on machine learning for algorithmic trading: Design and implement investment strategies based on smart algorithms that learn from data using Python*. Packt Publishing. https://research-ebsco-com.bibliotecavirtual.unad.edu.co/c/qcagk4/ebook-viewer/pdf/kjf2kt3kdr/page/pp_iii

Kane, F. (2017). *Hands-On Data Science and Python Machine Learning*. Packt Publishing. https://bibliotecavirtual.unad.edu.co/login?url=https://search.ebscohost.com/login.aspx?direct=true&db=nlebk&AN=1566405

Kyriakides, G., & Margaritis, K. G. (2019). *Hands-On Ensemble Learning with Python*. Packt Publishing. https://bibliotecavirtual.unad.edu.co/login?url=https://search.ebscohost.com/login.aspx?direct=true&db=nlebk&AN=2204655

Lantz, B. (2019). *Machine Learning with R: Expert Techniques for Predictive Modeling* (3rd ed.). Packt Publishing. https://bibliotecavirtual.unad.edu.co/login?url=https://search.ebscohost.com/login.aspx?direct=true&db=nlebk&AN=2106304

Liu, Y., Chen, P. H. C., Krause, J., & Peng, L. (2021). How artificial intelligence will change the future of healthcare. *Journal of Medical Internet Research, 23*(2), e26221. https://doi.org/10.2196/26221

López, J. D., & Angarita, F. A. (2021). Analítica de datos aplicada al sector salud: una herramienta para la toma de decisiones en la gestión pública. *Revista LOGOS Ciencia & Tecnología, 13*(1), 7–20. https://revistas.sena.edu.co/index.php/LOG/article/view/2635/3578

López Rodríguez, D. A., & Isaza Vides, S. A. (2023). *Modelos de Aprendizaje de Máquina para la detección de crisis epilépticas* [Trabajo de grado, Pontificia Universidad Javeriana]. Repositorio Universidad Javeriana. http://hdl.handle.net/10554/66804

Mejía, J. A., Oviedo-Benalcázar, M. A., Ordoñez, J. A., & Valencia, J. F. (2023). Aprendizaje automático aplicado a la predicción de diabetes mellitus, utilizando información socioeconómica y ambiental de usuarios del sistema de salud. *Revista Facultad Nacional de Salud Pública, 41*(2), 1–13. https://doi.org/10.17533/udea.rfnsp.471222

Ministerio de Salud y Protección Social. (2018). *Por la cual se adopta el Modelo de Acción Integral Territorial (MAITE) y se definen las Rutas Integrales de Atención en Salud (RIAS)*. Diario Oficial No. 50.748. https://www.minsalud.gov.co/sites/rid/lists/bibliotecadigital/ride/de/dij/resolucion-3280-de-2018.pdf

Ministerio de Salud y Protección Social. (2023). *Monitor RIAS - SISPRO*. https://web.sispro.gov.co/WebPublico/Consultas/MonitorRIAS.aspx

Ministerio de Salud y Protección Social. (2024). *Informe de Gestión 2024*. https://www.minsalud.gov.co/sites/rid/Lists/BibliotecaDigital/RIDE/DE/PES/informe-gestion-msps-2024.pdf

Molina Arévalo, N., Tovar Perilla, N. J., & Sánchez Echeverri, L. A. (2022). *Proceso de formulación y gestión de proyectos de investigación, desarrollo e innovación de acuerdo con los requisitos de la norma técnica colombiana 5802 del 2008*. Sello Editorial UNAD. https://doi.org/10.22490/9789586518581

Moran Pizarro, D. S., Dominguez Bonilla, S. J., Castaño Gutierrez, C., & Martinez Bez, C. E. (2023). De Hilbert a los algoritmos cuánticos: el rol del álgebra en el desarrollo de la

computación. *Publicaciones e Investigación, 17*(4). https://doi.org/10.22490/25394088.7503

Mosquera-Becerra, J., Pérez-Bustos, A. H., Díaz-Grajales, C., Quiroz-Arias, C., Salcedo-Cifuentes, M., Mejía-López, J., ... Barrera-Vergara, L. I. (2023). Promoción de la salud en América Latina: Coherencia en su implementación. *Hacia la Promoción de la Salud, 28*(2), 141–159. https://doi.org/10.17151/hpsal.2023.28.2.10

Muñoz-Ordóñez, C., Figueroa, A., Pencue-Fierro, L., & Muñoz-Ordóñez, J. (2020). Mapeo de cobertura terrestre utilizando aprendizaje máquina. *Investigación e Innovación en Ingenierías, 8*(Special Issue), 85–101. https://doi.org/10.17081/invinno.8.3.4706

Niño Rondón, C. V. (2024). *Sistema de detección de cáncer de piel con aprendizaje de máquina para dispositivo de bajo consumo* [Tesis de maestría, Pontificia Universidad Javeriana Cali]. Vitela. https://vitela.javerianacali.edu.co/handle/11522/2077

Observatorio Nacional de Calidad en Salud. (2023). *Informe de acciones de tutela en salud*. https://www.sispro.gov.co/observatorios/oncalidadsalud/Paginas/tutelas-en-salud.aspx

Pal, A., & Prakash, P. K. S. (2017). *Practical time series analysis: Master time series data processing, visualization, and modeling using Python*. Packt Publishing. https://research-ebsco-com.bibliotecavirtual.unad.edu.co/c/qcagk4/ebook-viewer/pdf/vgupyjs26v/page/pp_Cover

Parra Méndez, C. A., Santos Méndez, D. J., & Pineda Romero, M. M. (2021). Big data, educación y post-acuerdo. Cultura de paz en redes sociales. *Publicaciones e Investigación, 14*(3). https://doi.org/10.22490/25394088.4486

Perez, L., Perez, R., & Seca, M. V. (2020). *Metodología de la investigación científica*. Editorial Maipue. https://elibro-net.bibliotecavirtual.unad.edu.co/es/ereader/unad/138497

Polo-Triana, S. I., Ramírez-Sierra, Y. A., Arias-Osorio, J. E., Martínez-Vega, R. A., & Lamos-Díaz, H. (2023). Métodos de aprendizaje automático para predecir el comportamiento epidemiológico de enfermedades arbovirales: Revisión estructurada de literatura. *Revista Salud UIS, 55*, e23017. https://doi.org/10.18273/saluduis.55.e:23017

Raschka, S., & Mirjalili, V. (2017). *Python Machine Learning – Second Edition* (Vol. 2). Packt Publishing. https://bibliotecavirtual.unad.edu.co/login?url=https://search.ebscohost.com/login.aspx?direct=true&db=nlebk&AN=1606531

Rodríguez, E., Rodríguez, L. M., & Romero, D. (2022). Aplicación de herramientas de análisis de datos para la evaluación del servicio de atención en salud en Colombia. *Revista Publicaciones e Investigación, 16*(1), 33–50. https://hemeroteca.unad.edu.co/index.php/publicaciones-e-investigacion/article/view/6446/6079

Siddaway, A. P., Wood, A. M., & Hedges, L. V. (2019). *How to do a systematic review: A best practice guide for conducting and reporting narrative reviews, meta-analyses, and meta-syntheses*. *Annual Review of Psychology*, 70, 747–770. https://doi.org/10.1146/annurev-psych-010418-102803

Singla, S., Howitt, L., Medeiros, C., Grinspun, D., & Naik, S. (2024). O uso de técnicas de inteligência artificial nos sistemas de dados de enfermagem: Scoping Review. Revista da Faculdade de Ciências da Saúde, 26(3), 512–521. https://doi.org/10.29375/01237047.4634

Vishwas, B. V., & Patel, A. (2020). *Time series analysis and forecasting using Python*. Apress. https://link-springer-com.bibliotecavirtual.unad.edu.co/book/10.1007/978-1-4842-5992-4

Apéndices

Apéndice A

Análisis Exploratorio de Datos (EDA) – Monitor RIAS

```
# ============================================================
# APÉNDICE A. ANÁLISIS EXPLORATORIO DE DATOS (EDA) - MONITOR RIAS
# ============================================================


# --- Importación de librerías necesarias ---
import pandas as pd
import numpy as np
import re
import os
import folium
import warnings
import matplotlib.pyplot as plt
import seaborn as sns
from prettytable import PrettyTable
from sklearn.preprocessing import StandardScaler
from sklearn.cluster import KMeans
from sklearn.decomposition import PCA


# Configuración general
warnings.filterwarnings("ignore")
```

```
os.environ["LOKY_MAX_CPU_COUNT"] = "12"
os.environ["OMP_NUM_THREADS"] = "1"

# ============================================================
# 1. CARGA Y NORMALIZACIÓN DE LOS DATOS
# ============================================================

# Rutas de los archivos por nivel de desagregación
archivos = {
    "general": "BaseDatosMonitorRIAS/General.csv",
    "departamento": "BaseDatosMonitorRIAS/Departamento.csv",
    "municipio": "BaseDatosMonitorRIAS/Municipio.csv",
    "depto_asegurador": "BaseDatosMonitorRIAS/Depto_Asegurador.csv"
}

# Cargar los DataFrames y tipificar variables
dfs = {}
for nombre, ruta in archivos.items():
    df = pd.read_csv(ruta, sep=';', low_memory=False)

    if "FechaCorte" in df.columns:
        df["FechaCorte"] = pd.to_datetime(df["FechaCorte"], format="%Y-%m-%d",
errors="coerce")
```

```
    if "Anual" in df.columns:
        df["Anual"] = pd.to_numeric(df["Anual"], errors="coerce").astype(float)
    if "CodDepartamento" in df.columns:
        df["CodDepartamento"] = df["CodDepartamento"].astype("category")
    if "CodMunicipio" in df.columns:
        df["CodMunicipio"] = df["CodMunicipio"].astype("category")

    dfs[nombre] = df

# Renombrar claves para consistencia
df_general = dfs["general"]
df_departamento = dfs["departamento"]
df_municipio = dfs["municipio"]
df_depto_asegurador = dfs["depto_asegurador"]

dfs = {
    "General": df_general,
    "Departamento": df_departamento,
    "Municipio": df_municipio,
    "Depto_Asegurador": df_depto_asegurador
}

# Mostrar estructura general
```

```
tabla = PrettyTable()
tabla.field_names = ["DataFrame", "Filas", "Columnas"]
for nombre, df in dfs.items():
    filas, columnas = df.shape
    tabla.add_row([nombre, filas, columnas])
print(tabla)


# ============================================================
# 2. DICCIONARIO DE VARIABLES
# ============================================================

variables = [
    ("IdIndicador", "Identificador único asignado al indicador.", "Numérica", "Todos"),
    ("NomIndicador", "Nombre del indicador.", "Texto", "Todos"),
    ("FechaCorte", "Periodo de referencia del cálculo del indicador.", "Fecha", "Todos"),
    ("Anual", "Identifica si el dato corresponde a consolidado anual.", "Booleana", "Todos"),
    ("CodDepartamento", "Código DANE del departamento.", "Texto", "Depto, Mun,
Asegurador"),
    ("NomDepartamento", "Nombre del departamento.", "Texto", "Depto, Mun, Asegurador"),
    ("CodMunicipio", "Código DANE del municipio.", "Texto", "Municipio"),
    ("NomMunicipio", "Nombre del municipio.", "Texto", "Municipio"),
    ("CodAsegurador", "Código del asegurador en salud.", "Texto", "Asegurador"),
    ("NomAsegurador", "Nombre del asegurador en salud.", "Texto", "Asegurador"),
```

```
    ("Numerador", "Total de registros que cumplen la condición establecida.", "Numérica",
"Todos"),
    ("Denominador", "Población o universo de referencia para el cálculo.", "Numérica", "Todos"),
    ("Ind", "Valor porcentual resultante del cálculo del indicador.", "Decimal", "Todos")
]

tabla = PrettyTable()
tabla.title = "Diccionario resumido de variables Monitor RIAS"
tabla.field_names = ["Variable", "Definición", "Tipo de dato", "DataFrames"]

for var, defin, tipo, df_str in variables:
    tabla.add_row([var, defin, tipo, df_str])

print(tabla)

# ============================================================
# 3. VALIDACIÓN DE CALIDAD DE DATOS
# ============================================================

# --- Conteo de registros por FechaCorte ---
conteos_list = []
for nombre, df in dfs.items():
    if "FechaCorte" in df.columns:
```

```
        conteo = df.groupby("FechaCorte").size().reset_index(name="Conteo")
        conteo["DataFrame"] = nombre
        conteos_list.append(conteo)
conteos_df = pd.concat(conteos_list, ignore_index=True)

tabla = PrettyTable()
tabla.title = "Conteo por FechaCorte y DataFrame"
tabla.field_names = ["DataFrame", "FechaCorte", "Conteo"]
for _, row in conteos_df.iterrows():
    tabla.add_row([row["DataFrame"], row["FechaCorte"], row["Conteo"]])
print(tabla)

# --- Variables con valores nulos ---
resumen_list = []
for nombre, df in dfs.items():
    total = len(df)
    for col in df.columns:
        nulos = df[col].isna().sum()
        if nulos > 0:
            resumen_list.append({"DataFrame": nombre, "Variable": col, "Nulos": nulos, "Total":
total})
resumen_df = pd.DataFrame(resumen_list)
```

```
tabla = PrettyTable()
tabla.title = "Columnas con valores nulos"
tabla.field_names = ["DataFrame", "Variable", "Nulos", "Total"]
for _, row in resumen_df.iterrows():
    tabla.add_row([row["DataFrame"], row["Variable"], row["Nulos"], row["Total"]])
print(tabla)

# --- Variables duplicadas ---
claves = {
    "General": ["IdIndicador", "FechaCorte"],
    "Departamento": ["IdIndicador", "FechaCorte", "CodDepartamento"],
    "Municipio": ["IdIndicador", "FechaCorte", "CodDepartamento", "CodMunicipio"],
    "Depto_Asegurador": ["IdIndicador", "FechaCorte", "CodDepartamento", "CodAsegurador"]
}

tabla = PrettyTable()
tabla.title = "Resumen de duplicados por DataFrame"
tabla.field_names = ["DataFrame", "Columnas clave", "Duplicados"]

for nombre, cols in claves.items():
    df = dfs[nombre]
    dup = df.duplicated(subset=cols).sum()
    tabla.add_row([nombre, ", ".join(cols), dup])
```

```
print(tabla)


# ============================================================
# 4. VALIDACIÓN DE INTEGRIDAD ENTRE NIVELES
# ============================================================


def validar_agregaciones(df_general, df_departamento, df_municipio, df_depto_asegurador):
    claves = ["IdIndicador", "FechaCorte", "Anual"]
    resumenes = []


    def comparar(df1, df2, nombre1, nombre2):
        agg = df2.groupby(claves)[["Numerador", "Denominador"]].sum().reset_index()
        comp = df1.merge(agg, on=claves, suffixes=("", "_agg"), how="inner")
        comp["Dif_Num"] = comp["Numerador"] - comp["Numerador_agg"]
        iguales = (comp["Dif_Num"] == 0).sum()
        resumenes.append([f"{nombre1} vs {nombre2}", len(comp), iguales, len(comp) - iguales])


    comparar(df_general, df_departamento, "General", "Departamentos")
    comparar(df_general, df_municipio, "General", "Municipios")
    comparar(df_general, df_depto_asegurador, "General", "Aseguradores")


    tabla = PrettyTable()
```

```
    tabla.title = "Validación de agregaciones jerárquicas"
    tabla.field_names = ["Comparación", "Total", "Coinciden", "Difieren"]
    for r in resumenes:
        tabla.add_row(r)
    print(tabla)

validar_agregaciones(df_general, df_departamento, df_municipio, df_depto_asegurador)

# ============================================================
# 5. VERIFICACIÓN GEOGRÁFICA
# ============================================================

geo_depto = pd.read_csv("BaseDatosMonitorRIAS/GeoDepartamentos.csv", sep=";", dtype=str)
geo_mun = pd.read_csv("BaseDatosMonitorRIAS/GeoMunicipios.csv", sep=";", dtype=str)

def norm_code(x, width):
    if pd.isna(x): return None
    s = re.sub(r"\D", "", str(x)).zfill(width)
    return s

# Normalizar códigos DANE
df_departamento["CodDepartamento"] = df_departamento["CodDepartamento"].map(lambda x:
norm_code(x, 2))
```

```
df_municipio["CodMunicipio"] = df_municipio["CodMunicipio"].map(lambda x: norm_code(x,
5))
geo_depto["DepartamentoId"] = geo_depto["DepartamentoId"].map(lambda x: norm_code(x, 2))
geo_mun["MunicipioId"] = geo_mun["MunicipioId"].map(lambda x: norm_code(x, 5))

# Cruce y validación
cods_no_depto = sorted(set(df_departamento["CodDepartamento"]) -
set(geo_depto["DepartamentoId"]))
cods_no_mun = sorted(set(df_municipio["CodMunicipio"]) - set(geo_mun["MunicipioId"]))

print(f"Departamentos sin cruce: {len(cods_no_depto)}")
print(f"Municipios sin cruce: {len(cods_no_mun)}")

# Generar mapas
def generar_mapa(df_geo, lat_col, lon_col, nombre, archivo):
    m = folium.Map(location=[4.6, -74.1], zoom_start=5)
    for _, row in df_geo.iterrows():
        folium.CircleMarker(
            location=[float(row[lat_col]), float(row[lon_col])],
            radius=5, color="darkgreen", fill=True, fill_color="lime", fill_opacity=0.8
        ).add_to(m)
    m.save(archivo)
    print(f"Mapa guardado como {archivo}")
```

```
generar_mapa(geo_depto, "LatitudDepartamento", "LongitudDepartamento", "Departamentos",
"Departamentos_Cruzados.html")
generar_mapa(geo_mun, "LatitudMunicipio", "LongitudMunicipio", "Municipios",
"Municipios_Cruzados.html")

# ============================================================
# 6. ANÁLISIS ESTADÍSTICO DEL INDICADOR (Ind)
# ============================================================

resumen_list = []
for nombre, df in dfs.items():
    df["Ind"] = pd.to_numeric(df["Ind"], errors="coerce")
    desc = df["Ind"].describe(percentiles=[0.25, 0.5, 0.75])
    resumen_list.append({
        "DataFrame": nombre,
        "Mínimo": desc["min"],
        "Q1": desc["25%"],
        "Mediana": desc["50%"],
        "Q3": desc["75%"],
        "Máximo": desc["max"]
    })
```

```
print(pd.DataFrame(resumen_list))


# Boxplot y violín comparativo
frames = []
for nombre, df in dfs.items():
    temp = df.copy()
    temp["DataFrame"] = nombre
    temp["Ind"] = pd.to_numeric(temp["Ind"], errors="coerce")
    frames.append(temp[["DataFrame", "Ind"]])
df_all = pd.concat(frames)


fig, axes = plt.subplots(2, 1, figsize=(12, 10), sharex=True)
sns.boxplot(x="DataFrame", y="Ind", data=df_all, ax=axes[0], palette="Set3")
sns.violinplot(x="DataFrame", y="Ind", data=df_all, ax=axes[1], palette="Set2",
inner="quartile")
plt.xticks(rotation=15)
plt.tight_layout()
plt.savefig("comparacion_indicadores.jpg", dpi=300)
plt.show()


# ============================================================
# 7. CLUSTERIZACIÓN Y ANÁLISIS TEMPORAL
# ============================================================
```

```
df_filt = df_general[df_general["Anual"] == 0].copy()
df_filt["Anio"] = pd.to_datetime(df_filt["FechaCorte"]).dt.year
df_filt = df_filt[df_filt["Anio"] != 2025]

pivot = df_filt.groupby(["NomIndicador", "Anio"])["Ind"].mean().unstack().fillna(0)
X_scaled = StandardScaler().fit_transform(pivot)

kmeans = KMeans(n_clusters=6, random_state=42, n_init=10)
clusters = kmeans.fit_predict(X_scaled)
pivot["Cluster"] = clusters

# Visualización Heatmap
plt.figure(figsize=(14, 10))
sns.heatmap(pivot.drop(columns="Cluster"), cmap="YlGnBu", linewidths=0.3)
plt.title("Tendencia de Indicadores por Año y Clusters", weight="bold")
plt.tight_layout()
plt.savefig("heatmap_clusters.jpg", dpi=300)
plt.show()
```

Apéndice B

Modelos Predictivos y Evaluación – Nivel Nacional

```
# ============================================================
# COMPRENSIÓN Y ANÁLISIS DE LOS DATOS - MONITOR RIAS
# ============================================================


# Se importan las librerías necesarias
import warnings
warnings.filterwarnings('ignore')

import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns

from sklearn.model_selection import train_test_split
from sklearn.preprocessing import LabelEncoder, StandardScaler
from sklearn.linear_model import LinearRegression, Ridge, Lasso, ElasticNet
from sklearn.tree import DecisionTreeRegressor
from sklearn.ensemble import (
    RandomForestRegressor, GradientBoostingRegressor, ExtraTreesRegressor
)
from sklearn.neighbors import KNeighborsRegressor
```

```
from sklearn.svm import SVR
from sklearn.neural_network import MLPRegressor
from sklearn.metrics import r2_score, mean_absolute_error, mean_squared_error
from xgboost import XGBRegressor
from lightgbm import LGBMRegressor
from catboost import CatBoostRegressor

# ============================================================
# COMPRENSIÓN DE LOS DATOS
# ============================================================

# Cargar el conjunto de datos general
df = pd.read_csv("BaseDatosMonitorRIAS/General.csv", sep=';', low_memory=False)

# ============================================================
# PREPARACIÓN DE LOS DATOS
# ============================================================

# Conversión de fechas al formato de año
df["FechaCorte"] = pd.to_datetime(df["FechaCorte"], errors="coerce").dt.year

# Ajuste de tipos de datos
df["IdIndicador"] = df["IdIndicador"].astype(str)
```

```
df["Anual"] = df["Anual"].astype(str)

# Eliminación de filas con valores faltantes en variables clave
df = df.dropna(subset=["Ind", "FechaCorte", "IdIndicador", "Anual"])

# Filtrado de registros parciales o del año no consolidado
df = df[(df["Anual"] != "1") & (df["FechaCorte"] != 2025)]

# Definición de variables predictoras (X) y variable objetivo (y)
X = df[["FechaCorte", "IdIndicador"]].copy()
y = df["Ind"]

# Codificación de variable categórica
le_indicador = LabelEncoder()
X["IdIndicador"] = le_indicador.fit_transform(X["IdIndicador"])

# Escalamiento de variables predictoras
scaler = StandardScaler()
X_scaled = scaler.fit_transform(X)

# División del conjunto en entrenamiento (80%) y prueba (20%)
X_train, X_test, y_train, y_test = train_test_split(
    X_scaled, y, test_size=0.2, random_state=42
```

```
)

# ============================================================
# MODELADO
# ============================================================

# Implementación de modelos de regresión supervisada
modelos = {
    "Linear Regression": LinearRegression(),
    "Ridge": Ridge(alpha=1.0),
    "Lasso": Lasso(alpha=0.1),
    "Elastic Net": ElasticNet(alpha=0.1, l1_ratio=0.5),
    "KNN Regressor": KNeighborsRegressor(n_neighbors=5),
    "SVR": SVR(kernel='rbf'),
    "Decision Tree": DecisionTreeRegressor(random_state=42),
    "Random Forest": RandomForestRegressor(random_state=42),
    "Extra Trees": ExtraTreesRegressor(random_state=42),
    "Gradient Boosting": GradientBoostingRegressor(random_state=42),
    "Neural Network (MLP)": MLPRegressor(
        hidden_layer_sizes=(64, 32), max_iter=3000, random_state=42
    ),
    "XGBoost": XGBRegressor(
        random_state=42,
```

```
        n_estimators=300,
        learning_rate=0.05,
        max_depth=6,
        objective='reg:squarederror',
        verbosity=0
    ),
    "LightGBM": LGBMRegressor(
        random_state=42,
        n_estimators=300,
        learning_rate=0.05,
        num_leaves=31,
        verbose=-1
    ),
    "CatBoost": CatBoostRegressor(
        verbose=0,
        random_state=42,
        iterations=300,
        learning_rate=0.05,
        depth=6
    )
}

# Entrenamiento y evaluación de modelos
```

```
resultados = []


for nombre, modelo in modelos.items():
    modelo.fit(X_train, y_train)
    y_pred = modelo.predict(X_test)


    r2 = r2_score(y_test, y_pred)
    mae = mean_absolute_error(y_test, y_pred)
    mse = mean_squared_error(y_test, y_pred)
    rmse = np.sqrt(mse)


    resultados.append({
        "Modelo": nombre,
        "R²": r2,
        "MAE": mae,
        "MSE": mse,
        "RMSE": rmse
    })


# ============================================================
# EVALUACIÓN
# ============================================================
```

```
# Consolidación y orden de resultados
resultados_df = pd.DataFrame(resultados).sort_values(by="R²",
ascending=False).reset_index(drop=True)

print("\nComparativo de modelos supervisados de regresión:")
print(resultados_df.to_string(index=False))

# Normalización de métricas para visualización comparativa
resultados_norm = resultados_df.copy()
resultados_norm["MAE_norm"] = resultados_norm["MAE"] / resultados_norm["MAE"].max()
resultados_norm["RMSE_norm"] = resultados_norm["RMSE"] /
resultados_norm["RMSE"].max()
resultados_norm["MSE_norm"] = resultados_norm["MSE"] / resultados_norm["MSE"].max()

# ------------------------------------------------------------
# Visualización de métricas comparativas
# ------------------------------------------------------------
plt.figure(figsize=(12,6))

sns.lineplot(data=resultados_norm, x="Modelo", y="R²", label="R²", marker='o')
sns.lineplot(data=resultados_norm, x="Modelo", y="MAE_norm", label="MAE", marker='o')
sns.lineplot(data=resultados_norm, x="Modelo", y="RMSE_norm", label="RMSE", marker='o')
sns.lineplot(data=resultados_norm, x="Modelo", y="MSE_norm", label="MSE", marker='o')
```

```
plt.xticks(rotation=45, ha='right')
plt.ylabel("Valor (normalizado)")
plt.title("Comparativo de métricas de error por modelo")
plt.legend()
plt.grid(True, linestyle='--', alpha=0.5)
plt.tight_layout()
plt.show()

# ----------------------------------------------------------
# Visualización en barras por métrica
# ----------------------------------------------------------
fig, axes = plt.subplots(2, 2, figsize=(14, 8))
fig.suptitle("Comparativo general de métricas por modelo - Nivel Nacional", fontsize=14,
fontweight='bold')

sns.barplot(ax=axes[0,0], data=resultados_df, x="Modelo", y="R²", palette="Blues_d")
axes[0,0].set_title("Coeficiente de determinación (R²)")
axes[0,0].tick_params(axis='x', rotation=90)

sns.barplot(ax=axes[0,1], data=resultados_df, x="Modelo", y="MAE", palette="Oranges_d")
axes[0,1].set_title("Error absoluto medio (MAE)")
axes[0,1].tick_params(axis='x', rotation=90)
```

```
sns.barplot(ax=axes[1,0], data=resultados_df, x="Modelo", y="RMSE", palette="Greens_d")
axes[1,0].set_title("Raíz del error cuadrático medio (RMSE)")
axes[1,0].tick_params(axis='x', rotation=90)

sns.barplot(ax=axes[1,1], data=resultados_df, x="Modelo", y="MSE", palette="Reds_d")
axes[1,1].set_title("Error cuadrático medio (MSE)")
axes[1,1].tick_params(axis='x', rotation=90)

plt.tight_layout(rect=[0, 0, 1, 0.95])
plt.show()

# ------------------------------------------------------------
# Comparación de valores reales vs predichos
# ------------------------------------------------------------
fig, axes = plt.subplots(4, 4, figsize=(18, 18))
fig.suptitle("Comparación de valores reales vs predichos", fontsize=15, fontweight='bold')
axes = axes.flatten()

for i, (nombre, modelo) in enumerate(modelos.items()):
    modelo.fit(X_train, y_train)
    y_pred = modelo.predict(X_test)
```

```
    sns.scatterplot(x=y_test, y=y_pred, ax=axes[i], s=15, alpha=0.6)
    axes[i].plot([y_test.min(), y_test.max()], [y_test.min(), y_test.max()], 'r--')
    axes[i].set_title(nombre, fontsize=9)
    axes[i].set_xlabel("Real")
    axes[i].set_ylabel("Predicho")

plt.tight_layout(rect=[0, 0, 1, 0.96])
plt.show()

# ------------------------------------------------------------
# Distribución de residuos por modelo
# ------------------------------------------------------------
fig, axes = plt.subplots(4, 4, figsize=(18, 18))
fig.suptitle("Distribución de residuos por modelo", fontsize=15, fontweight='bold')
axes = axes.flatten()

for i, (nombre, modelo) in enumerate(modelos.items()):
    modelo.fit(X_train, y_train)
    y_pred = modelo.predict(X_test)
    residuos = y_test - y_pred

    sns.histplot(residuos, bins=20, kde=True, ax=axes[i], color="steelblue")
    axes[i].axvline(0, color='red', linestyle='--')
```

```
    axes[i].set_title(nombre, fontsize=9)
    axes[i].set_xlabel("Residuo")

plt.tight_layout(rect=[0, 0, 1, 0.96])
plt.show()
```

Apéndice C

Modelos Predictivos y Evaluación – Nivel Departamental

```
# ============================================================
# COMPRENSIÓN Y ANÁLISIS DE LOS DATOS - NIVEL DEPARTAMENTAL
# ============================================================

# Se importan las librerías necesarias
import warnings
warnings.filterwarnings('ignore')

import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns

from sklearn.model_selection import train_test_split
from sklearn.preprocessing import LabelEncoder, StandardScaler
```

```
from sklearn.linear_model import LinearRegression, Ridge, Lasso, ElasticNet
from sklearn.tree import DecisionTreeRegressor
from sklearn.ensemble import (
    RandomForestRegressor, GradientBoostingRegressor, ExtraTreesRegressor
)
from sklearn.neighbors import KNeighborsRegressor
from sklearn.svm import SVR
from sklearn.neural_network import MLPRegressor
from sklearn.metrics import r2_score, mean_absolute_error, mean_squared_error
from xgboost import XGBRegressor
from lightgbm import LGBMRegressor
from catboost import CatBoostRegressor


# ============================================================
# COMPRENSIÓN DE LOS DATOS
# ============================================================


# Cargar el conjunto de datos departamental
df = pd.read_csv("BaseDatosMonitorRIAS/Departamento.csv", sep=';', low_memory=False)


# ============================================================
# PREPARACIÓN DE LOS DATOS
# ============================================================
```

```
# Conversión de fechas al formato de año
df["FechaCorte"] = pd.to_datetime(df["FechaCorte"], errors="coerce").dt.year

# Ajuste de tipos de datos
df["CodDepartamento"] = df["CodDepartamento"].astype(str)
df["IdIndicador"] = df["IdIndicador"].astype(str)
df["Anual"] = df["Anual"].astype(str)

# Eliminación de filas con valores faltantes en variables clave
df = df.dropna(subset=["Ind", "CodDepartamento", "FechaCorte", "IdIndicador", "Anual"])

# Filtrado de registros inválidos o no consolidados
df = df[
    (df["CodDepartamento"] != "-1") &
    (df["Anual"] != "1") &
    (df["FechaCorte"] != 2025)
]

# Definición de variables predictoras (X) y objetivo (y)
X = df[["CodDepartamento", "FechaCorte", "IdIndicador"]].copy()
y = df["Ind"]
```

```
# Codificación de variables categóricas
le_depto = LabelEncoder()
le_indicador = LabelEncoder()
X["CodDepartamento"] = le_depto.fit_transform(X["CodDepartamento"])
X["IdIndicador"] = le_indicador.fit_transform(X["IdIndicador"])

# Escalamiento de variables predictoras
scaler = StandardScaler()
X_scaled = scaler.fit_transform(X)

# División del conjunto en entrenamiento (80%) y prueba (20%)
X_train, X_test, y_train, y_test = train_test_split(
    X_scaled, y, test_size=0.2, random_state=42
)

# ============================================================
# MODELADO
# ============================================================

# Implementación de modelos de regresión supervisada
modelos = {
    "Linear Regression": LinearRegression(),
    "Ridge": Ridge(alpha=1.0),
```

```
"Lasso": Lasso(alpha=0.1),
"Elastic Net": ElasticNet(alpha=0.1, l1_ratio=0.5),
"KNN Regressor": KNeighborsRegressor(n_neighbors=5),
"SVR": SVR(kernel='rbf'),
"Decision Tree": DecisionTreeRegressor(random_state=42),
"Random Forest": RandomForestRegressor(random_state=42),
"Extra Trees": ExtraTreesRegressor(random_state=42),
"Gradient Boosting": GradientBoostingRegressor(random_state=42),
"Neural Network (MLP)": MLPRegressor(
    hidden_layer_sizes=(64, 32), max_iter=3000, random_state=42
),
"XGBoost": XGBRegressor(
    random_state=42,
    n_estimators=300,
    learning_rate=0.05,
    max_depth=6,
    objective='reg:squarederror',
    verbosity=0
),
"LightGBM": LGBMRegressor(
    random_state=42,
    n_estimators=300,
    learning_rate=0.05,
```

```
        num_leaves=31,
        verbose=-1
    ),
    "CatBoost": CatBoostRegressor(
        verbose=0,
        random_state=42,
        iterations=300,
        learning_rate=0.05,
        depth=6
    )
}

# Entrenamiento y evaluación de cada modelo
resultados = []

for nombre, modelo in modelos.items():
    modelo.fit(X_train, y_train)
    y_pred = modelo.predict(X_test)

    r2 = r2_score(y_test, y_pred)
    mae = mean_absolute_error(y_test, y_pred)
    mse = mean_squared_error(y_test, y_pred)
    rmse = np.sqrt(mse)
```

```
    resultados.append({
        "Modelo": nombre,
        "R²": r2,
        "MAE": mae,
        "MSE": mse,
        "RMSE": rmse
    })

# ============================================================
# EVALUACIÓN
# ============================================================

# Consolidación y orden de resultados
resultados_df = pd.DataFrame(resultados).sort_values(by="R²",
ascending=False).reset_index(drop=True)

print("\nComparativo de modelos supervisados de regresión (Nivel Departamental):")
print(resultados_df.to_string(index=False))

# Normalización de métricas para visualización
resultados_norm = resultados_df.copy()
resultados_norm["MAE_norm"] = resultados_norm["MAE"] / resultados_norm["MAE"].max()
```

```
resultados_norm["RMSE_norm"] = resultados_norm["RMSE"] /
resultados_norm["RMSE"].max()
resultados_norm["MSE_norm"] = resultados_norm["MSE"] / resultados_norm["MSE"].max()

# ------------------------------------------------------------
# Visualización de métricas comparativas
# ------------------------------------------------------------
plt.figure(figsize=(12,6))
sns.lineplot(data=resultados_norm, x="Modelo", y="R²", label="R²", marker='o')
sns.lineplot(data=resultados_norm, x="Modelo", y="MAE_norm", label="MAE", marker='o')
sns.lineplot(data=resultados_norm, x="Modelo", y="RMSE_norm", label="RMSE", marker='o')
sns.lineplot(data=resultados_norm, x="Modelo", y="MSE_norm", label="MSE", marker='o')

plt.xticks(rotation=45, ha='right')
plt.ylabel("Valor (normalizado)")
plt.title("Comparativo de métricas de error por modelo - Nivel Departamental")
plt.legend()
plt.grid(True, linestyle='--', alpha=0.5)
plt.tight_layout()
plt.show()

# ------------------------------------------------------------
# Gráficos de barras comparativos de métricas
```

```
# ----------------------------------------------------------
fig, axes = plt.subplots(2, 2, figsize=(14, 8))
fig.suptitle("Comparativo general de métricas por modelo - Nivel Departamental", fontsize=14,
fontweight='bold')

sns.barplot(ax=axes[0,0], data=resultados_df, x="Modelo", y="R²", palette="Blues_d")
axes[0,0].set_title("Coeficiente de determinación (R²)")
axes[0,0].tick_params(axis='x', rotation=90)

sns.barplot(ax=axes[0,1], data=resultados_df, x="Modelo", y="MAE", palette="Oranges_d")
axes[0,1].set_title("Error absoluto medio (MAE)")
axes[0,1].tick_params(axis='x', rotation=90)

sns.barplot(ax=axes[1,0], data=resultados_df, x="Modelo", y="RMSE", palette="Greens_d")
axes[1,0].set_title("Raíz del error cuadrático medio (RMSE)")
axes[1,0].tick_params(axis='x', rotation=90)

sns.barplot(ax=axes[1,1], data=resultados_df, x="Modelo", y="MSE", palette="Reds_d")
axes[1,1].set_title("Error cuadrático medio (MSE)")
axes[1,1].tick_params(axis='x', rotation=90)

plt.tight_layout(rect=[0, 0, 1, 0.95])
plt.show()
```

```
# ------------------------------------------------------------
# Comparación de valores reales vs predichos
# ------------------------------------------------------------
fig, axes = plt.subplots(4, 4, figsize=(18, 18))
fig.suptitle("Comparación de valores reales vs predichos - Nivel Departamental", fontsize=15,
fontweight='bold')
axes = axes.flatten()

for i, (nombre, modelo) in enumerate(modelos.items()):
    modelo.fit(X_train, y_train)
    y_pred = modelo.predict(X_test)

    sns.scatterplot(x=y_test, y=y_pred, ax=axes[i], s=15, alpha=0.6)
    axes[i].plot([y_test.min(), y_test.max()], [y_test.min(), y_test.max()], 'r--')
    axes[i].set_title(nombre, fontsize=9)
    axes[i].set_xlabel("Real")
    axes[i].set_ylabel("Predicho")
plt.tight_layout(rect=[0, 0, 1, 0.96])
plt.show()

# ------------------------------------------------------------
# Distribución de residuos por modelo
```

```
# ------------------------------------------------------------
fig, axes = plt.subplots(4, 4, figsize=(18, 18))
fig.suptitle("Distribución de residuos por modelo - Nivel Departamental", fontsize=15,
fontweight='bold')
axes = axes.flatten()
for i, (nombre, modelo) in enumerate(modelos.items()):
    modelo.fit(X_train, y_train)
    y_pred = modelo.predict(X_test)
    residuos = y_test - y_pred

    sns.histplot(residuos, bins=20, kde=True, ax=axes[i], color="steelblue")
    axes[i].axvline(0, color='red', linestyle='--')
    axes[i].set_title(nombre, fontsize=9)
    axes[i].set_xlabel("Residuo")

plt.tight_layout(rect=[0, 0, 1, 0.96])
plt.show()
```

Apéndice D

Modelos Predictivos y Evaluación – Nivel Municipal

```
# ============================================================

# COMPRENSIÓN Y ANÁLISIS DE LOS DATOS - NIVEL MUNICIPAL

# ============================================================


import warnings

warnings.filterwarnings('ignore')


import pandas as pd

import numpy as np

import matplotlib.pyplot as plt

import seaborn as sns


from sklearn.model_selection import train_test_split

from sklearn.preprocessing import LabelEncoder, StandardScaler

from sklearn.tree import DecisionTreeRegressor

from sklearn.ensemble import RandomForestRegressor, GradientBoostingRegressor

from sklearn.metrics import r2_score, mean_absolute_error, mean_squared_error

from xgboost import XGBRegressor

from lightgbm import LGBMRegressor

from catboost import CatBoostRegressor
```

```
# ============================================================

# COMPRENSIÓN DE LOS DATOS

# ============================================================


# Cargar el conjunto de datos municipal

df = pd.read_csv("BaseDatosMonitorRIAS/Municipio.csv", sep=';', low_memory=False)


# ============================================================

# PREPARACIÓN DE LOS DATOS

# ============================================================


# Conversión de fechas al formato de año

df["FechaCorte"] = pd.to_datetime(df["FechaCorte"], errors="coerce").dt.year


# Ajuste de tipos de datos

df["CodDepartamento"] = df["CodDepartamento"].astype(str)

df["CodMunicipio"] = df["CodMunicipio"].astype(str)

df["IdIndicador"] = df["IdIndicador"].astype(str)

df["Anual"] = df["Anual"].astype(str)


# Eliminación de valores faltantes

df = df.dropna(subset=["Ind", "CodDepartamento", "CodMunicipio", "FechaCorte",

"IdIndicador", "Anual"])
```

```
# Filtrado de registros inválidos
df = df[
    (df["CodDepartamento"] != "-1") &
    (df["Anual"] != "1") &
    (df["FechaCorte"] != 2025)
]

# Definición de variables predictoras (X) y objetivo (y)
X = df[["CodDepartamento", "CodMunicipio", "FechaCorte", "IdIndicador"]].copy()
y = df["Ind"]

# Codificación de variables categóricas
le_depto = LabelEncoder()
le_muni = LabelEncoder()
le_indicador = LabelEncoder()
X["CodDepartamento"] = le_depto.fit_transform(X["CodDepartamento"])
X["CodMunicipio"] = le_muni.fit_transform(X["CodMunicipio"])
X["IdIndicador"] = le_indicador.fit_transform(X["IdIndicador"])

# Escalamiento de variables
scaler = StandardScaler()
X_scaled = scaler.fit_transform(X)
```

```
# División del conjunto en entrenamiento (80%) y prueba (20%)
X_train, X_test, y_train, y_test = train_test_split(
    X_scaled, y, test_size=0.2, random_state=42
)

# ============================================================
# MODELADO
# ============================================================

# Se emplean los seis modelos con mejor desempeño general
modelos = {
    "Decision Tree": DecisionTreeRegressor(random_state=42),
    "Random Forest": RandomForestRegressor(random_state=42),
    "Gradient Boosting": GradientBoostingRegressor(random_state=42),
    "XGBoost": XGBRegressor(
        random_state=42,
        n_estimators=300,
        learning_rate=0.05,
        max_depth=6,
        objective='reg:squarederror',
        verbosity=0
    ),
```

```
    "LightGBM": LGBMRegressor(
        random_state=42,
        n_estimators=300,
        learning_rate=0.05,
        num_leaves=31,
        verbose=-1
    ),
    "CatBoost": CatBoostRegressor(
        verbose=0,
        random_state=42,
        iterations=300,
        learning_rate=0.05,
        depth=6
    )
}

# Entrenamiento y evaluación
resultados = []

for nombre, modelo in modelos.items():
    modelo.fit(X_train, y_train)
    y_pred = modelo.predict(X_test)
```

```
    r2 = r2_score(y_test, y_pred)
    mae = mean_absolute_error(y_test, y_pred)
    mse = mean_squared_error(y_test, y_pred)
    rmse = np.sqrt(mse)

    resultados.append({
        "Modelo": nombre,
        "R²": r2,
        "MAE": mae,
        "MSE": mse,
        "RMSE": rmse
    })

# ============================================================
# EVALUACIÓN
# ============================================================

# Consolidación y orden de resultados
resultados_df = pd.DataFrame(resultados).sort_values(by="R²",
ascending=False).reset_index(drop=True)

print("\nComparativo de modelos supervisados de regresión - Nivel Municipal:")
print(resultados_df.to_string(index=False))
```

```
# Normalización de métricas para visualización
resultados_norm = resultados_df.copy()
resultados_norm["MAE_norm"] = resultados_norm["MAE"] / resultados_norm["MAE"].max()
resultados_norm["RMSE_norm"] = resultados_norm["RMSE"] /
resultados_norm["RMSE"].max()
resultados_norm["MSE_norm"] = resultados_norm["MSE"] / resultados_norm["MSE"].max()

# ------------------------------------------------------------
# Visualización de métricas comparativas
# ------------------------------------------------------------
plt.figure(figsize=(12,6))
sns.lineplot(data=resultados_norm, x="Modelo", y="R²", label="R²", marker='o')
sns.lineplot(data=resultados_norm, x="Modelo", y="MAE_norm", label="MAE", marker='o')
sns.lineplot(data=resultados_norm, x="Modelo", y="RMSE_norm", label="RMSE", marker='o')
sns.lineplot(data=resultados_norm, x="Modelo", y="MSE_norm", label="MSE", marker='o')

plt.xticks(rotation=45, ha='right')
plt.ylabel("Valor (normalizado)")
plt.title("Comparativo de métricas de error por modelo - Nivel Municipal")
plt.legend()
plt.grid(True, linestyle='--', alpha=0.5)
plt.tight_layout()
```

```
plt.show()

# ------------------------------------------------------------
# Gráficos de barras comparativos de métricas
# ------------------------------------------------------------
fig, axes = plt.subplots(2, 2, figsize=(14, 8))
fig.suptitle("Comparativo general de métricas por modelo - Nivel Municipal", fontsize=14,
fontweight='bold')

sns.barplot(ax=axes[0,0], data=resultados_df, x="Modelo", y="R²", palette="Blues_d")
axes[0,0].set_title("Coeficiente de determinación (R²)")
axes[0,0].tick_params(axis='x', rotation=90)

sns.barplot(ax=axes[0,1], data=resultados_df, x="Modelo", y="MAE", palette="Oranges_d")
axes[0,1].set_title("Error absoluto medio (MAE)")
axes[0,1].tick_params(axis='x', rotation=90)

sns.barplot(ax=axes[1,0], data=resultados_df, x="Modelo", y="RMSE", palette="Greens_d")
axes[1,0].set_title("Raíz del error cuadrático medio (RMSE)")
axes[1,0].tick_params(axis='x', rotation=90)

sns.barplot(ax=axes[1,1], data=resultados_df, x="Modelo", y="MSE", palette="Reds_d")
axes[1,1].set_title("Error cuadrático medio (MSE)")
```

```
axes[1,1].tick_params(axis='x', rotation=90)


plt.tight_layout(rect=[0, 0, 1, 0.95])
plt.show()


# ------------------------------------------------------------
# Comparación de valores reales vs predichos
# ------------------------------------------------------------
fig, axes = plt.subplots(2, 3, figsize=(18, 10))
fig.suptitle("Comparación de valores reales vs predichos - Nivel Municipal", fontsize=15,
fontweight='bold')
axes = axes.flatten()


for i, (nombre, modelo) in enumerate(modelos.items()):
    modelo.fit(X_train, y_train)
    y_pred = modelo.predict(X_test)
    sns.scatterplot(x=y_test, y=y_pred, ax=axes[i], s=15, alpha=0.6)
    axes[i].plot([y_test.min(), y_test.max()], [y_test.min(), y_test.max()], 'r--')
    axes[i].set_title(nombre, fontsize=9)
    axes[i].set_xlabel("Valor Real")
    axes[i].set_ylabel("Valor Predicho")


plt.tight_layout(rect=[0, 0, 1, 0.96])
```

```
plt.show()


# ------------------------------------------------------------
# Distribución de residuos
# ------------------------------------------------------------
fig, axes = plt.subplots(2, 3, figsize=(18, 10))
fig.suptitle("Distribución de residuos por modelo - Nivel Municipal", fontsize=15,
fontweight='bold')
axes = axes.flatten()

for i, (nombre, modelo) in enumerate(modelos.items()):
    modelo.fit(X_train, y_train)
    y_pred = modelo.predict(X_test)
    residuos = y_test - y_pred
    sns.histplot(residuos, bins=20, kde=True, ax=axes[i], color="steelblue")
    axes[i].axvline(0, color='red', linestyle='--')
    axes[i].set_title(nombre, fontsize=9)
    axes[i].set_xlabel("Residuo")

plt.tight_layout(rect=[0, 0, 1, 0.96])
plt.show()
```

Apéndice E *Modelos Predictivos y Evaluación – Nivel Institucional*

```
# ============================================================
# COMPRENSIÓN Y ANÁLISIS DE LOS DATOS - NIVEL INSTITUCIONAL
(ASEGURADOR)
# ============================================================

import warnings
warnings.filterwarnings('ignore')

import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns

from sklearn.model_selection import train_test_split
from sklearn.preprocessing import LabelEncoder, StandardScaler
from sklearn.tree import DecisionTreeRegressor
from sklearn.ensemble import RandomForestRegressor, GradientBoostingRegressor
from sklearn.metrics import r2_score, mean_absolute_error, mean_squared_error
from xgboost import XGBRegressor
from lightgbm import LGBMRegressor
from catboost import CatBoostRegressor

# ============================================================
```

```
# COMPRENSIÓN DE LOS DATOS
# ============================================================


# Cargar el conjunto de datos institucional (asegurador)
df = pd.read_csv("BaseDatosMonitorRIAS/Depto_Asegurador.csv", sep=';',
low_memory=False)


# ============================================================
# PREPARACIÓN DE LOS DATOS
# ============================================================


# Conversión de fechas al formato de año
df["FechaCorte"] = pd.to_datetime(df["FechaCorte"], errors="coerce").dt.year


# Ajuste de tipos de datos
df["CodDepartamento"] = df["CodDepartamento"].astype(str)
df["CodAsegurador"] = df["CodAsegurador"].astype(str)
df["IdIndicador"] = df["IdIndicador"].astype(str)
df["Anual"] = df["Anual"].astype(str)


# Eliminación de valores faltantes
df = df.dropna(subset=["Ind", "CodDepartamento", "CodAsegurador", "FechaCorte",
"IdIndicador", "Anual"])
```

```
# Filtrado de registros inválidos
df = df[
    (df["CodDepartamento"] != "-1") &
    (df["Anual"] != "1") &
    (df["FechaCorte"] != 2025)
]

# Definición de variables predictoras (X) y objetivo (y)
X = df[["CodDepartamento", "CodAsegurador", "FechaCorte", "IdIndicador"]].copy()
y = df["Ind"]

# Codificación de variables categóricas
le_depto = LabelEncoder()
le_aseg = LabelEncoder()
le_indicador = LabelEncoder()
X["CodDepartamento"] = le_depto.fit_transform(X["CodDepartamento"])
X["CodAsegurador"] = le_aseg.fit_transform(X["CodAsegurador"])
X["IdIndicador"] = le_indicador.fit_transform(X["IdIndicador"])

# Escalamiento de variables
scaler = StandardScaler()
X_scaled = scaler.fit_transform(X)
```

```
# División del conjunto en entrenamiento (80%) y prueba (20%)
X_train, X_test, y_train, y_test = train_test_split(
    X_scaled, y, test_size=0.2, random_state=42
)

# ============================================================
# MODELADO
# ============================================================

# Se emplean los seis modelos con mejor desempeño general
modelos = {
    "Decision Tree": DecisionTreeRegressor(random_state=42),
    "Random Forest": RandomForestRegressor(random_state=42),
    "Gradient Boosting": GradientBoostingRegressor(random_state=42),
    "XGBoost": XGBRegressor(
        random_state=42,
        n_estimators=300,
        learning_rate=0.05,
        max_depth=6,
        objective='reg:squarederror',
        verbosity=0
    ),
```

```
    "LightGBM": LGBMRegressor(
        random_state=42,
        n_estimators=300,
        learning_rate=0.05,
        num_leaves=31,
        verbose=-1
    ),
    "CatBoost": CatBoostRegressor(
        verbose=0,
        random_state=42,
        iterations=300,
        learning_rate=0.05,
        depth=6
    )
}

# Entrenamiento y evaluación
resultados = []

for nombre, modelo in modelos.items():
    modelo.fit(X_train, y_train)
    y_pred = modelo.predict(X_test)
```

```
    r2 = r2_score(y_test, y_pred)
    mae = mean_absolute_error(y_test, y_pred)
    mse = mean_squared_error(y_test, y_pred)
    rmse = np.sqrt(mse)

    resultados.append({
        "Modelo": nombre,
        "R²": r2,
        "MAE": mae,
        "MSE": mse,
        "RMSE": rmse
    })

# ============================================================
# EVALUACIÓN
# ============================================================

# Consolidación y orden de resultados
resultados_df = pd.DataFrame(resultados).sort_values(by="R²",
ascending=False).reset_index(drop=True)

print("\nComparativo de modelos supervisados de regresión - Nivel Asegurador:")
print(resultados_df.to_string(index=False))
```

```
# Normalización de métricas para visualización
resultados_norm = resultados_df.copy()
resultados_norm["MAE_norm"] = resultados_norm["MAE"] / resultados_norm["MAE"].max()
resultados_norm["RMSE_norm"] = resultados_norm["RMSE"] /
resultados_norm["RMSE"].max()
resultados_norm["MSE_norm"] = resultados_norm["MSE"] / resultados_norm["MSE"].max()

# ------------------------------------------------------------
# Visualización de métricas comparativas
# ------------------------------------------------------------
plt.figure(figsize=(12,6))
sns.lineplot(data=resultados_norm, x="Modelo", y="R²", label="R²", marker='o')
sns.lineplot(data=resultados_norm, x="Modelo", y="MAE_norm", label="MAE", marker='o')
sns.lineplot(data=resultados_norm, x="Modelo", y="RMSE_norm", label="RMSE", marker='o')
sns.lineplot(data=resultados_norm, x="Modelo", y="MSE_norm", label="MSE", marker='o')

plt.xticks(rotation=45, ha='right')
plt.ylabel("Valor (normalizado)")
plt.title("Comparativo de métricas de error por modelo - Nivel Asegurador")
plt.legend()
plt.grid(True, linestyle='--', alpha=0.5)
plt.tight_layout()
```

```
plt.show()

# ------------------------------------------------------------
# Gráficos de barras comparativos de métricas
# ------------------------------------------------------------
fig, axes = plt.subplots(2, 2, figsize=(14, 8))
fig.suptitle("Comparativo general de métricas por modelo - Nivel Asegurador", fontsize=14,
fontweight='bold')

sns.barplot(ax=axes[0,0], data=resultados_df, x="Modelo", y="R²", palette="Blues_d")
axes[0,0].set_title("Coeficiente de determinación (R²)")
axes[0,0].tick_params(axis='x', rotation=90)

sns.barplot(ax=axes[0,1], data=resultados_df, x="Modelo", y="MAE", palette="Oranges_d")
axes[0,1].set_title("Error absoluto medio (MAE)")
axes[0,1].tick_params(axis='x', rotation=90)

sns.barplot(ax=axes[1,0], data=resultados_df, x="Modelo", y="RMSE", palette="Greens_d")
axes[1,0].set_title("Raíz del error cuadrático medio (RMSE)")
axes[1,0].tick_params(axis='x', rotation=90)

sns.barplot(ax=axes[1,1], data=resultados_df, x="Modelo", y="MSE", palette="Reds_d")
axes[1,1].set_title("Error cuadrático medio (MSE)")
```

```
axes[1,1].tick_params(axis='x', rotation=90)


plt.tight_layout(rect=[0, 0, 1, 0.95])
plt.show()


# ------------------------------------------------------------
# Comparación de valores reales vs predichos
# ------------------------------------------------------------
fig, axes = plt.subplots(2, 3, figsize=(18, 10))
fig.suptitle("Comparación de valores reales vs predichos - Modelos Aseguradores", fontsize=15,
fontweight='bold')
axes = axes.flatten()


for i, (nombre, modelo) in enumerate(modelos.items()):
    modelo.fit(X_train, y_train)
    y_pred = modelo.predict(X_test)
    sns.scatterplot(x=y_test, y=y_pred, ax=axes[i], s=15, alpha=0.6)
    axes[i].plot([y_test.min(), y_test.max()], [y_test.min(), y_test.max()], 'r--')
    axes[i].set_title(nombre, fontsize=9)
    axes[i].set_xlabel("Valor Real (Ind)")
    axes[i].set_ylabel("Valor Predicho")


plt.tight_layout(rect=[0, 0, 1, 0.96])
```

```
plt.show()


# ------------------------------------------------------------
# Distribución de residuos
# ------------------------------------------------------------
fig, axes = plt.subplots(2, 3, figsize=(18, 10))
fig.suptitle("Distribución de residuos por modelo - Nivel Asegurador", fontsize=15,
fontweight='bold')
axes = axes.flatten()

for i, (nombre, modelo) in enumerate(modelos.items()):
    modelo.fit(X_train, y_train)
    y_pred = modelo.predict(X_test)
    residuos = y_test - y_pred
    sns.histplot(residuos, bins=20, kde=True, ax=axes[i], color="steelblue")
    axes[i].axvline(0, color='red', linestyle='--')
    axes[i].set_title(nombre, fontsize=9)
    axes[i].set_xlabel("Residuo")

plt.tight_layout(rect=[0, 0, 1, 0.96])
plt.show()
```

Apéndice F

Proyección de Resultados y Modelo Combinado por Votación

```
# ============================================================
# PREDICCIÓN DE AÑOS PRÓXIMOS (2025–2027) CON MODELO COMBINADO
# ============================================================


# Seleccionar los tres mejores modelos según R²
mejores_modelos = resultados_df.nlargest(3, "R²")["Modelo"].tolist()
print(f"\nModelos seleccionados para la votación combinada: {mejores_modelos}")


# Crear diccionario con pesos proporcionales a su R²
pesos = resultados_df.set_index("Modelo").loc[mejores_modelos]["R²"]
pesos = pesos / pesos.sum()
print("\nPesos asignados según R²:\n", pesos)


# Instanciar los modelos seleccionados
modelos_votacion = {nombre: modelos[nombre] for nombre in mejores_modelos}


# Entrenar los modelos seleccionados
for nombre, modelo in modelos_votacion.items():
    modelo.fit(X_train, y_train)


# Crear combinaciones futuras (años 2025–2027)
```

```
anios_futuros = [2025, 2026, 2027]
indicadores_cod = le_indicador.transform(le_indicador.classes_)

futuro = pd.DataFrame(
    [(anio, ind) for anio in anios_futuros for ind in indicadores_cod],
    columns=["FechaCorte", "IdIndicador"]
)

# Escalar variables futuras
futuro_scaled = scaler.transform(futuro)

# Obtener predicciones individuales de cada modelo
predicciones_modelos = {}
for nombre, modelo in modelos_votacion.items():
    predicciones_modelos[nombre] = modelo.predict(futuro_scaled)

# Calcular predicción final mediante votación ponderada
futuro["Prediccion_Ind"] = sum(
    predicciones_modelos[nombre] * pesos[nombre] for nombre in mejores_modelos
)

# Decodificar IdIndicador a nombres originales
futuro["IdIndicador"] = le_indicador.inverse_transform(futuro["IdIndicador"])
```

```
# Incorporar nombres desde el dataset original
mapa_nombres = (
    df[["IdIndicador", "NomIndicador"]]
    .drop_duplicates(subset=["IdIndicador"])
    .dropna(subset=["NomIndicador"])
)
futuro = futuro.merge(mapa_nombres, on="IdIndicador", how="left")

# Combinar datos históricos y proyecciones
historico = df[["FechaCorte", "IdIndicador", "NomIndicador", "Ind"]].copy()
historico.rename(columns={"Ind": "Resultado"}, inplace=True)
futuro.rename(columns={"Prediccion_Ind": "Resultado"}, inplace=True)
df_total = pd.concat([historico, futuro], ignore_index=True)

# ============================================================
# ASEGURAR COBERTURA COMPLETA DE INDICADORES Y AÑOS
# ============================================================
indicadores_unicos = df_total["NomIndicador"].unique()
anios_completos = sorted(df_total["FechaCorte"].unique())

# Crear malla completa Indicador × Año
malla = pd.MultiIndex.from_product(
```

```
    [indicadores_unicos, anios_completos],
    names=["NomIndicador", "FechaCorte"]
).to_frame(index=False)

# Unir con datos reales/predichos
df_completo = malla.merge(df_total, on=["NomIndicador", "FechaCorte"], how="left")

# Crear tabla pivote
pivot = df_completo.pivot_table(
    index="NomIndicador",
    columns="FechaCorte",
    values="Resultado",
    aggfunc="mean"
).sort_index()
# Ordenar por valor proyectado en 2027
if 2027 in pivot.columns:
    orden_2027 = pivot[2027].sort_values(ascending=False)
    pivot = pivot.loc[orden_2027.index]

# ============================================================
# VISUALIZACIÓN: HEATMAP HISTÓRICO Y PROYECTADO (2018–2027)
# ============================================================
plt.figure(figsize=(13, 10))
```

```
sns.heatmap(
    pivot,
    cmap="YlGnBu",
    linewidths=0.3,
    cbar_kws={'label': 'Resultado del indicador'},
    vmin=0, vmax=100
)
# Línea separadora visual entre datos históricos y proyecciones
if 2024 in pivot.columns:
    plt.axvline(x=list(pivot.columns).index(2024) + 0.5, color='red', linestyle='--', lw=1.2)
plt.title("Proyección combinada de resultados del Monitor RIAS (2018–2027)")
plt.xlabel("Año")
plt.ylabel("Indicador")
plt.tight_layout()
plt.show()


# ============================================================
# TABLA DE VALORES PROYECTADOS 2025–2027
# ============================================================
proyecciones = (
    pivot[[2025, 2026, 2027]]
    .reset_index()
    .sort_values(by=2027, ascending=False)
```

```
)

print("\nPredicciones promedio por indicador (votación ponderada, años 2025–2027):")
print(proyecciones.round(2).to_string(index=False))
```

Apéndice G

Fuentes de Datos y Repositorio de Bases Utilizadas

BaseDatosMonitorRIAS. (2025). Repositorio de datos para análisis y modelado del Monitor RIAS. Recuperado de: https://unadvirtualedu-my.sharepoint.com/:f:/g/personal/lecortesl_unadvirtual_edu_co/EnjH_EZ4EYZMnFOV4zxw9PkBTDiJOM3foI4-p8YOm9gvRA?e=yt7Dsz

Ministerio de Salud y Protección Social – SISPRO. (2025). Monitor RIAS. Recuperado de: https://web.sispro.gov.co/WebPublico/Consultas/MonitorRIAS.aspx