MODELO PARA PREDECIR SI UN ASPIRANTE ADMITIDO SE MATRICULARÁ EN UN PROGRAMA DE PREGRADO DE UNA UNIVERSIDAD COLOMBIANA, APLICANDO TÉCNICAS DE CIENCIA DE DATOS

Carlos Rodrigo Piñeros Castro
Código 9015411

Proyecto Aplicado para optar al título de
Magister en Ciencia de Datos

Director
Diego Luis Linares Ospina

Codirectora
Gloria Inés Álvarez

FACULTAD DE INGENIERÍA Y CIENCIAS
MAESTRÍA EN CIENCIA DE DATOS
SANTIAGO DE CALI, DICIEMBRE DE 2025

TABLA DE CONTENIDO

LISTA DE FIGURAS

LISTA DE TABLAS

LISTA DE ANEXOS

INTRODUCCIÓN

La disminución de matrículas en las universidades colombianas representa un desafío significativo. Este proyecto aborda esta problemática mediante el desarrollo de un modelo predictivo que, a partir de datos sociodemográficos y contextuales, permite estimar si un aspirante se matriculará o no.

En este documento se presenta el proceso de preparación y análisis de datos para el entrenamiento de seis modelos de aprendizaje automático de clasificación binaria: Random Forest, XGBoost, Regresión Logística, MLP Classifier, Maquina de vectores SVM y TabTransformer.

Se evaluó el rendimiento de cada uno de los modelos para seis programas académicos ofertados por la institución (Administración de Empresas, Comercio Internacional, Comunicación Social, Contaduría Pública, Derecho y Psicología), considerando que cada programa presenta un comportamiento distinto en términos de matrícula y además difieren en cuanto a características y cantidad de registros para el procesamiento.

Para cada programa académico, se seleccionó el modelo con el mejor rendimiento obtenido durante el proceso de entrenamiento y de ajuste de hiperparámetros, y se construyó un prototipo para realizar predicciones con datos de nuevos aspirantes. A través de una interfaz sencilla, el usuario puede introducir la información relevante del aspirante y recibir una predicción basada en los patrones aprendidos por el modelo.

Esta predicción le permitirá a la institución enfocar sus esfuerzos en orientar la toma de decisiones del aspirante asegurando la matrícula. No obstante, es importante señalar que los modelos alcanzaron un rendimiento moderado, debido a que fueron construidos con la información que la universidad recopila al momento de la inscripción. Uno de los principales hallazgos del proyecto es que esta información resulta insuficiente para capturar adecuadamente la intención real de matrícula del aspirante. Recolectar datos adicionales relacionados con su nivel de interés y comportamiento previo, permitirá desarrollar modelos más precisos y representativos en futuras implementaciones.

1 DEFINICIÓN DEL PROBLEMA

Este capítulo presenta el contexto institucional y las condiciones que motivan la formulación del proyecto aplicado. Se aborda la importancia de fortalecer los procesos de admisión y matrícula en universidades privadas, así como los desafíos que implica identificar oportunamente a los aspirantes con mayor probabilidad de convertirse en estudiantes activos.

1.1 PLANTEAMIENTO DEL PROBLEMA

Las universidades privadas, como cualquier otra empresa, buscan garantizar los recursos financieros para el cumplimiento de su objeto social que les permita alcanzar su desarrollo, posicionamiento y consolidación [1]. Su sostenibilidad financiera depende en gran medida de la capacidad para captar estudiantes y de la efectividad de los procesos de admisión y matrícula.

El comportamiento de las variables socioeconómicas y demográficas les plantea la necesidad de tomar decisiones estratégicas que garanticen su sostenibilidad, así como de identificar con precisión la población a la cual dirige su oferta en aras de garantizar el ingreso de cohortes completas.

El incremento en la oferta de instituciones y programas académicos [2] se suma a la incidencia de las variables mencionadas y representa un reto en la intención de captar estudiantes, diseñar propuestas atractivas y enfocarse en aquellos inscritos que tiene mayor probabilidad de matricularse según el cumplimiento de ciertas condiciones que les pueden facilitar el desarrollo de la carrera universitaria.

Los departamentos de mercadeo de otras universidades se han enfocado en el diseño de publicidad dirigida a personas cuyas características indican una alta probabilidad de efectuar el proceso de matrícula, lo cual propicia una mayor eficiencia en el proceso [3]. Esta estrategia permite además focalizar los recursos y ofrecer opciones a los estudiantes para facilitar su ingreso a la universidad.

Sin embargo, este tipo de estrategia demanda la identificación de a qué estudiantes dedicar los esfuerzos desde el momento en el que realizan la inscripción mostrando intención de ingreso a la universidad y de quiénes, pese a haber realizado la inscripción y ser admitidos no culminarán el proceso de ingreso. Para esto, se requiere el estudio de las variables soportado en datos y la aplicación de modelos que permitan integrar dichas variables facilitando el reconocimiento o no de patrones y del comportamiento de los aspirantes en el proceso de matrícula.

La universidad a la que hace referencia este estudio enfrenta el desafío de adaptarse a los nuevos escenarios sociodemográficos y tiene el reto de estudiar detalladamente el comportamiento de su matrícula en un entorno cada vez más dinámico, exigente y de alta competitividad. Para

afrontar este reto, cuenta con datos históricos de los aspirantes inscritos y matriculados en los diferentes programas académicos, para los últimos cinco años, pero no contaba con un modelo de análisis de datos que le permita generar predicciones basadas en los mismos.

1.2 FORMULACIÓN DEL PROBLEMA

¿Cómo desarrollar un modelo para predecir si un aspirante admitido a un programa de pregrado de una universidad colombiana se matriculará?

2 OBJETIVOS DEL PROYECTO

Este capítulo presenta los objetivos que orientaron el desarrollo del proyecto, estableciendo la meta general y las acciones específicas necesarias para la construcción, evaluación e implementación del modelo predictivo para estimar la matrícula de los aspirantes.

2.1 OBJETIVO GENERAL

Desarrollar un modelo para predecir si un aspirante admitido se matriculará en un programa de pregrado de una universidad colombiana, aplicando técnicas de ciencia de datos.

2.2 OBJETIVOS ESPECÍFICOS

- Preparar los datos para el entrenamiento de diferentes modelos.
- Entrenar diferentes modelos de clasificación para predecir si el aspirante se va a matricular.
- Utilizar métricas de evaluación para verificar el rendimiento del modelo.
- Desarrollar un prototipo que permita introducir nuevos datos de aspirantes y obtener predicciones.

3 MARCO TEÓRICO Y ANTECEDENTES

Este capítulo presenta los fundamentos conceptuales y las referencias previas que sustentan el desarrollo del proyecto, contextualizando el fenómeno de la matrícula universitaria y el uso de técnicas de aprendizaje automático para su predicción.

En primer lugar, se abordan los conceptos esenciales relacionados con la matrícula en instituciones universitarias y las variables que pueden influir en la decisión de un aspirante. Posteriormente, se introducen los conceptos del aprendizaje automático, enfatizando el aprendizaje supervisado como enfoque central para el desarrollo de modelos de clasificación. Así mismo, se describen las métricas utilizadas para evaluar el desempeño de los modelos y su importancia en la validación de resultados.

Finalmente, se presentan antecedentes relevantes relacionados con modelos de predicción de matrícula o permanencia estudiantil, que muestran la pertinencia del desarrollo del modelo predictivo y del prototipo implementado.

3.1 MARCO TEÓRICO

3.1.1 MATRÍCULA UNVERSITARIA Y VARIABLES QUE INCIDEN

El concepto de matrícula es ampliamente utilizado en las instituciones educativas, puesto que hace referencia concretamente al público a quien va dirigido el servicio educativo. Se trata del proceso mediante el cual una persona formaliza su relación con la institución permitiéndole acceder a los programas y servicios que ha contratado en un periodo determinado. Algunas instituciones conciben el proceso de matrícula desde el momento en el que la persona otorga sus datos y acredita el cumplimiento de requisitos, hasta el momento en el que realiza el pago y adquiere la condición de estudiante.

El sistema SNIES (Sistema Nacional de Información de la Educación Superior) del Ministerio de Educación Nacional, define como matriculados a los “estudiantes de todas las cohortes en todos los programas académicos en educación superior” y realiza una diferenciación con aquellos estudiantes que ingresan por primera vez a la institución en calidad de estudiantes, denominándolos “Matriculados en Primer Curso” [4].

Cada periodo, la universidad recibe cierto número de inscripciones de aspirantes a cada uno de sus programas. De estos inscritos, algunos acreditan el cumplimiento de los requisitos establecidos por el Ministerio de Educación y por la universidad alcanzando el estatus de admitido. Finalmente, sólo una parte de los admitidos formaliza su matrícula realizando el respectivo contrato como estudiante, como matriculado en primer curso según los términos del

SNIES. A esta matrícula es a la que hace referencia este estudio descartando la matrícula por permanencia en la institución.

La cantidad de matriculados en primer curso determina el comportamiento de la matrícula posterior, que es vital para las proyecciones que realiza la universidad en la asignación de sus recursos como infraestructura y capacidad docente. Se entiende que, si son pocos los matriculados en primer curso en uno o varios periodos, la totalidad de la matrícula disminuye y por tanto los ingresos con los que se sostiene la institución.

La predicción de matrícula en instituciones universitarias ha sido un tema de creciente interés en los últimos años. Estudios previos han demostrado que factores como el rendimiento académico, los factores socioeconómicos y la oferta académica influyen significativamente en la decisión de los estudiantes de matricularse [5]. Sin embargo, la mayoría de estas investigaciones se han centrado en instituciones de otros países y con sistemas educativos diferentes al nuestro. Estos estudios han identificado variables como el rendimiento académico, los factores socioeconómicos, la oferta académica y la percepción de la calidad institucional como factores clave.

En términos generales, los estudios consideran entre las variables de ingreso datos demográficos como: nacionalidad, sexo, edad y estado civil [5] y tienen en cuenta consideraciones sobre la ubicación de la vivienda, composición de la familia y cercanía al núcleo familiar [6], a la vez que se plantean la posibilidad de realizar proyecciones de adaptación a la vida universitaria partiendo de variables académicas y de rendimiento escolar previo.

Por otra parte, algunos estudios se enfocan en la consideración del perfil socioeconómico [7] para determinar la posibilidad de permanencia del estudiante, particularmente si se trata de una universidad de carácter privado. Este perfil socioeconómico considera: Características de la vivienda de origen y las personas con quienes vive el estudiante en su lugar de origen, edad, género, número de hijos del estudiante, estado civil, religión, nivel de escolaridad de los padres del estudiante, origen de los recursos con los que cuenta el estudiante, situación laboral del estudiante, capacidad de pago de la matrícula y gastos de sostenimiento, modalidad de vinculación laboral del estudiante trabajador, tiempo de dedicación laboral e ingreso.

Variables como la motivación, vocación, disciplina, organización, responsabilidad, curiosidad, capacidad de adaptación, comunicación efectiva, trabajo en equipo, iniciativa, resiliencia y pasión por el aprendizaje [8], también han sido puestas en consideración, pero las dificultades para su conceptualización y medición representa todavía un reto para la investigación. Estudios que tienen en cuenta este tipo de variables, realizan apreciaciones de tipo cualitativo orientadas a contextos específicos.

3.1.2 APRENDIZAJE AUTOMATICO (MACHINE LEARNING)

El aprendizaje automático es una rama de la inteligencia artificial que se centra en el análisis y la interpretación de patrones y estructuras de datos que permiten a un sistema aprender y tomar decisiones de acuerdo con la aplicación de ciertos algoritmos o de ciertas reglas de operación. Se entrena el algoritmo con un volumen gigantesco de datos para que aprenda [9] y sea capaz de tomar decisiones y hacer recomendaciones basándose únicamente en los datos introducidos, pudiendo incorporar correcciones para tomar decisiones futuras. Existen tres principales tipos de modelos de aprendizaje automático:

Aprendizaje supervisado
El aprendizaje supervisado es una técnica donde se utiliza un conjunto de datos etiquetados para entrenar a un algoritmo. Es decir, se le proporcionan al algoritmo ejemplos de entrada (datos) y sus correspondientes salidas correctas (etiquetas) [10]. A partir de estos ejemplos, el algoritmo aprende a asociar las entradas con las salidas y, de esta manera, puede realizar predicciones sobre nuevos datos.

Aprendizaje no supervisado
En el aprendizaje no supervisado los algoritmos buscan patrones y estructuras inherentes en los datos sin la necesidad de etiquetas o salidas predefinidas [9]. A diferencia del aprendizaje supervisado, donde se le dice al modelo cuál es la respuesta correcta, en el aprendizaje no supervisado el modelo debe descubrir por sí mismo las relaciones y agrupaciones subyacentes en los datos.

Aprendizaje por refuerzo
El aprendizaje por refuerzo se basa en la interacción entre un agente y un entorno. El agente realiza acciones en el entorno, y en respuesta, el entorno proporciona una señal de recompensa al agente [10]. El agente utiliza esta señal para ajustar su comportamiento y tomar mejores decisiones en el futuro.

3.1.3 ALGORITMOS DE CLASIFICACIÓN

Los algoritmos de clasificación son métodos de aprendizaje supervisado que se utilizan para categorizar datos en diferentes clases o grupos [9]. Estos modelos son ampliamente utilizados en tareas donde se requiere predecir una clase o estado, siendo los más comunes la Regresión logística, Random Forest, XGBoost, MLPClassifier, SVM y Tab Transformer.

Regresión Logística
La regresión logística es uno de los modelos fundamentales en el aprendizaje supervisado para resolver problemas de clasificación binaria. Este modelo estima la probabilidad de que un dato

pertenezca a una clase a partir de una combinación lineal de sus características. Para convertir dicha combinación en una probabilidad (Fig. 1), aplica la función logística o sigmoide, que transforma cualquier valor real en un rango entre 0 y 1. Russell y Norvig [10] destacan su simplicidad, interpretabilidad y eficiencia computacional, lo que la convierte en una técnica ampliamente utilizada en dominios como el análisis crediticio, la medicina y la evaluación de riesgo.

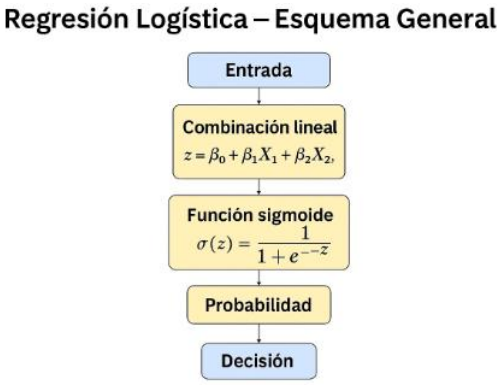

Fig. 1. Modelo Regresión Logística

De acuerdo con la documentación disponible, una de sus principales ventajas es la facilidad para interpretar los coeficientes asociados a cada variable, ya que indican la dirección y magnitud de su influencia sobre la clase positiva. Esto permite comprender el impacto real de cada atributo en la predicción, aspecto importante en contextos donde la transparencia del modelo es un requisito institucional. También es robusta frente a datos moderadamente ruidosos y puede extenderse a problemas multiclase mediante técnicas como softmax (regresión logística multinomial).

Sin embargo, su rendimiento disminuye cuando las relaciones entre las variables y la clase son altamente no lineales, o cuando existen fuertes interacciones entre características que el modelo no puede capturar sin transformaciones adicionales. Aun así, su capacidad de generalización en conjuntos de datos pequeños y medianos, junto con su bajo riesgo de sobreajuste, hacen de la regresión logística una herramienta fundamental en la modelación predictiva.

Bosques Aleatorios (Random Forest):
Se trata de un método de ensamble descrito por Russell y Norvig [10], donde múltiples árboles son entrenados sobre diferentes subconjuntos del conjunto de datos y características. Cada árbol se construye de forma independiente utilizando técnicas como bootstrap sampling (muestreo aleatorio con reemplazo) y la selección aleatoria de variables en cada división del árbol.

El principio fundamental de Random Forest es que la combinación de muchos modelos débiles o moderados puede producir un modelo final más robusto y preciso. El proceso de votación entre árboles reduce considerablemente la varianza inherente a los árboles de decisión individuales y mejora la capacidad de generalización, incluso en escenarios con datos ruidosos o estructuras complejas (Ver figura 2).

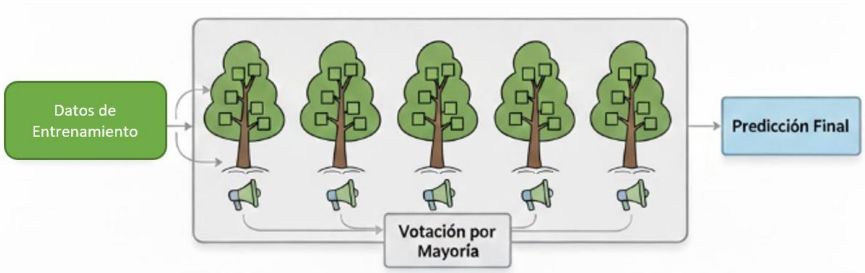

Fig. 2. Modelo Random Forest

Entre sus ventajas destacan su alto rendimiento, estabilidad y capacidad para manejar un gran número de variables sin riesgo significativo de sobreajuste. Además, permite calcular medidas de importancia de las características, lo que facilita la interpretación del modelo y la selección de atributos relevantes. Sin embargo, su estructura compuesta lo hace menos interpretable que un árbol de decisión único, y su entrenamiento puede requerir más recursos computacionales. Aun así, sigue siendo uno de los algoritmos más utilizados en la práctica debido a su eficacia en una amplia variedad de problemas reales.

XGBoost (Extreme Gradient Boosting)

XGBoost es un algoritmo de ensamble basado en gradient boosting, diseñado para optimizar tanto el rendimiento como la eficiencia computacional en tareas de clasificación y regresión. Propuesto inicialmente por Chen y Guestrin (2016) [11], este modelo ha sido ampliamente reconocido por su capacidad para manejar grandes conjuntos de datos, capturar relaciones no lineales y ofrecer un desempeño superior en tareas de machine learning.

El funcionamiento de XGBoost se basa en entrenar una secuencia de árboles de decisión, donde cada nuevo árbol intenta corregir los errores cometidos por el conjunto de árboles previos. La optimización se logra mediante el cálculo del gradiente de la función de pérdida y la incorporación de regularización tanto en los pesos de los árboles como en su complejidad (fig. 3.). Esto reduce el riesgo de sobreajuste y mejora la capacidad de generalización.

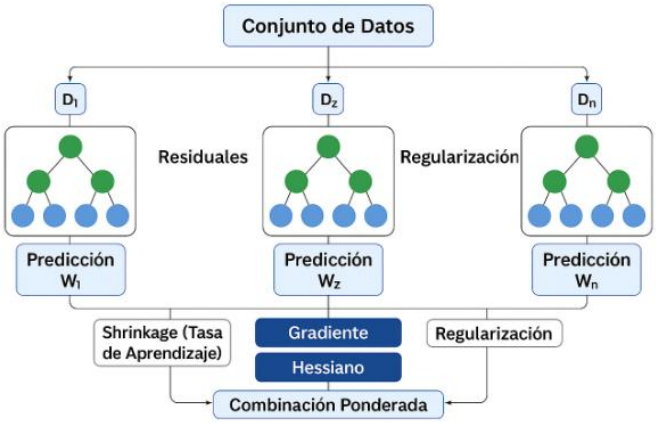

Fig. 3. Modelo XGBoost

Entre las principales características de XGBoost se destacan:

- Regularización avanzada (L1 y L2), que lo diferencia de otros métodos de boosting.
- Uso eficiente de memoria, mediante estructuras comprimidas.
- Soporte para manejo de valores faltantes, aprendiendo dinámicamente la mejor dirección para cada división.
- Paralelización del proceso de construcción de árboles, lo que acelera el entrenamiento.

XGBoost es particularmente efectivo en problemas con variables mixtas y estructuras complejas de interacción al interior de los datos. Por su robustez, es uno de los algoritmos más utilizados en entornos empresariales y académicos.

MLPClassifier (Perceptrón Multicapa)

El modelo MLPClassifier, implementado en la librería scikit-learn, es una red neuronal artificial de tipo feed-forward entrenada mediante el algoritmo de retropropagación. Su estructura se compone de capas densamente conectadas: una capa de entrada, una o más capas ocultas y una capa de salida con activación apropiada para tareas de clasificación binaria o multiclase.

Según Géron (2019) [12], los perceptrones multicapa pueden aproximar cualquier función continua no lineal, lo que los convierte en herramientas potentes para la clasificación. Cada neurona aplica una transformación no lineal a sus entradas, permitiendo al modelo aprender patrones complejos e interacciones entre características que los modelos lineales no pueden capturar. El rendimiento de un MLP depende de varios factores (Fig. 4):

- Número de capas y neuronas, que determinan la capacidad de aprendizaje del modelo.
- Funciones de activación como ReLU, tanh o logistic.
- Técnicas de regularización, como early stopping, dropout o penalización L2.
- Escalado de características, requisito fundamental para redes neuronales entrenadas con gradiente descendente.

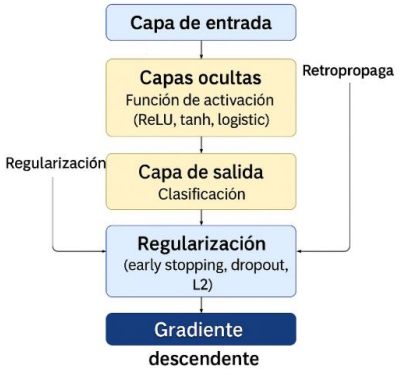

Fig. 4. Modelo MLP Classifier

A diferencia de los modelos basados en árboles, los MLP requieren datos escalados y tienden a necesitar más datos para evitar el sobreajuste. Sin embargo, cuando están bien configurados, pueden lograr rendimientos comparables o superiores a métodos tradicionales en conjuntos de datos estructurados.

Support Vector Machine (SVM):
Es uno de los algoritmos más representativos y robustos dentro del aprendizaje supervisado, ampliamente utilizado en problemas de clasificación debido a su capacidad para manejar relaciones lineales y no lineales de manera eficiente [13]. Su objetivo principal consiste en encontrar el hiperplano óptimo que separe las clases en el espacio de características, maximizando el margen, es decir, la distancia entre el hiperplano y los puntos más cercanos de cada clase. Estos puntos cercanos se denominan vectores de soporte y desempeñan un papel fundamental en la construcción de la frontera de decisión, ya que determinan su posición y orientación.

Una de las mayores fortalezas del SVM radica en su capacidad para abordar problemas donde las clases no son linealmente separables. Para lograrlo, implementa el conocido kernel trick (Fig. 5), una técnica que transforma implícitamente los datos a un espacio de mayor dimensión donde es posible encontrar un hiperplano lineal separador [13]. Esta transformación se realiza de manera eficiente mediante funciones kernel, sin necesidad de calcular explícitamente las nuevas dimensiones. Entre los kernels más utilizados se encuentran el lineal, el polinomial y el Radial Basis Function (RBF), este último especialmente eficaz para capturar patrones complejos y fronteras no lineales. Gracias a este mecanismo, SVM puede modelar problemas de clasificación con estructuras altamente complejas sin requerir grandes cantidades de datos, lo que lo vuelve especialmente útil en escenarios donde el tamaño de muestra es limitado.

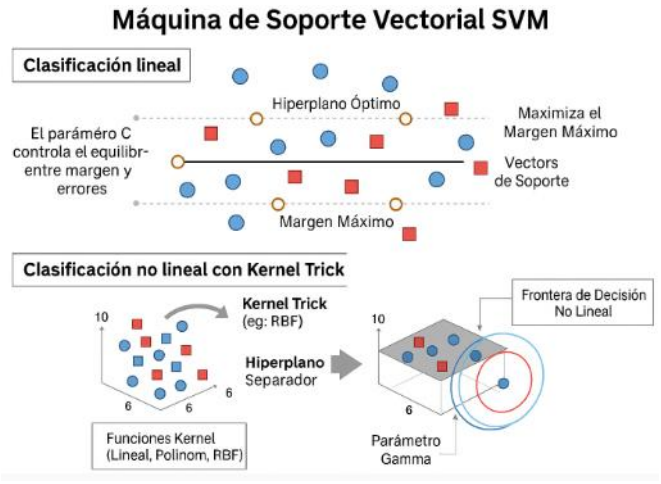

Fig. 5. Máquina de soporte SVM

De acuerdo con la documentación disponible, el SVM incorpora un parámetro de regularización, C, que controla el equilibrio entre maximizar el margen y minimizar los errores de clasificación.

Valores altos de C priorizan la clasificación correcta de los datos de entrenamiento, mientras que valores más bajos favorecen márgenes más amplios y una mayor capacidad de generalización. Otro hiperparámetro relevante es el parámetro del kernel (como gamma en el caso del RBF), que determina el grado de influencia de cada punto sobre la frontera de decisión. La combinación adecuada de estos parámetros es crucial para obtener un rendimiento óptimo y evitar tanto el sobreajuste como el subajuste.

Según Pedregosa et al. (2011) [13], su implementación en scikit-learn ha contribuido significativamente a su popularización en entornos académicos y profesionales, facilitando su uso mediante una interfaz intuitiva y optimizaciones internas que mejoran su tiempo de ejecución. En resumen, el modelo Support Vector Machine constituye una herramienta poderosa y flexible, capaz de ofrecer resultados altamente competitivos incluso cuando los datos presentan relaciones no lineales o se dispone de un número limitado de muestras.

TabTransformer:

Es un modelo de aprendizaje profundo para datos tabulares que aborda las limitaciones de las redes neuronales tradicionales en este dominio. Se basa en la arquitectura Transformer y su mecanismo de autoatención (self-attention) para mejorar el modelado de las características.

La innovación clave radica en aplicar capas Transformer exclusivamente a los embeddings de las características categóricas. Este proceso genera embeddings contextuales y robustos (Fig. 6), ya que el valor de cada característica categórica se ajusta y enriquece al considerar el contexto de todas las demás características de la fila. Finalmente, estos embeddings contextuales se concatenan con las características numéricas originales y se pasan a una red Multi-Layer Perceptron (MLP) para la predicción, lo que resulta en una arquitectura que ofrece una precisión comparable e incluso superior a los métodos basados en árboles en muchas tareas de clasificación y regresión [14].

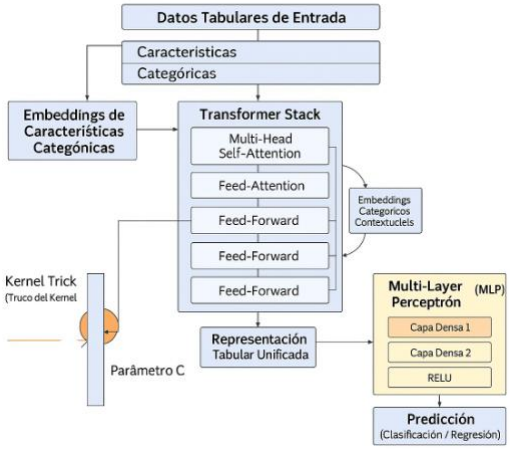

Fig. 6. Modelo Tab Transformer

En términos prácticos, esta arquitectura ofrece varias ventajas:

- Captura relaciones complejas entre variables categóricas sin necesidad de ingeniería manual;
- Mejora la generalización al aprender embeddings contextualizados;
- Se adapta bien a escenarios con alta cardinalidad categórica, donde modelos basados en árboles suelen degradar su rendimiento.

3.1.4 EVALUACIÓN DE MODELOS DE APRENDIZAJE AUTOMÁTICO

La evaluación de modelos de aprendizaje automático es un paso fundamental en cualquier proyecto de ciencia de datos, ya que permite determinar la calidad del modelo, identificar áreas de mejora y tomar decisiones informadas sobre su implementación.
Mediante el uso de métricas de rendimiento, es posible analizar las fortalezas y debilidades del modelo, garantizando no solo un buen desempeño durante el entrenamiento, sino también su capacidad para generalizar adecuadamente a nuevos datos [15]. Asimismo, la evaluación hace posible comparar diferentes modelos y seleccionar aquel que mejor se ajuste al problema planteado. Entre las métricas empleadas en este proyecto se encuentran la Precisión, el Recall, el F1-score, el ROC AUC, la Matriz de Confusión y la Validación Cruzada.

Precisión y Recall
la precisión mide la proporción de positivos y negativos correctamente identificados, mientras que el recall mide la proporción de casos positivos correctamente identificados [16]. En la ecuación 1 vemos su fórmula:

$$Precision = \frac{TP}{TP+FP} \; Recall = \frac{TP}{TP+FN} \tag{1}$$

donde: **TP** = Verdaderos positivos
FP = Falsos positivos
FN = Falsos negativos

F1 score
Es una métrica que combina de manera equilibrada la precisión y el recall, proporcionando una medida más completa del desempeño de un modelo de clasificación [16], como vemos en la ecuación 2:

$$F1 = \frac{Precision \; x \; Recall}{Precision+Recall} \tag{2}$$

Es especialmente útil cuando se trabaja con conjuntos de datos desbalanceados, donde una clase está representada de forma significativamente mayor que otra.

Matriz de confusión

La matriz de confusión permite visualizar el desempeño del modelo al mostrar la cantidad de predicciones correctas e incorrectas según cada clase. Se estructura de la siguiente manera:

	Predicho Positivo	Predicho Negativo
Real Positivo	TP	FN
Real Negativo	FP	TN

Es una tabla que nos permite visualizar de forma clara y concisa el desempeño de un algoritmo de clasificación, mostrando la cantidad de verdaderos positivos, verdaderos negativos, falsos positivos y falsos negativos [16].

Validación cruzada

La validación cruzada es un procedimiento que permite evaluar con mayor robustez el desempeño de un modelo y mitigar el riesgo de sobreajuste. En lugar de realizar una única partición de entrenamiento y prueba, este método divide los datos en k subconjuntos y realiza múltiples evaluaciones, generando una estimación más precisa del error de generalización [16].

En su forma más común, k-fold cross-validation, el error final se calcula como se ve en la ecuación 3:

$$\text{Error}_{CV} = \frac{1}{k}\sum_{i=1}^{k}\text{Error}_i \tag{3}$$

donde $Error_i$ es el error obtenido en la i-ésima iteración.

ROC AUC (Área bajo la curva ROC)

La métrica ROC AUC (Receiver Operating Characteristic – Area Under the Curve) evalúa la capacidad de un modelo para distinguir entre las clases positiva y negativa. La curva ROC representa gráficamente la relación entre la tasa de verdaderos positivos (TPR) y la tasa de falsos positivos (FPR) para distintos umbrales de decisión del modelo.

La tasa de verdaderos positivos (TPR) o Recall se define como se ve en la ecuación 4:

$$TPR = \frac{TP}{TP + FN} \tag{4}$$

La tasa de falsos positivos (FPR) se calcula como se ve en la ecuación 5:

$$FPR = \frac{FP}{FP + TN} \tag{5}$$

El AUC corresponde al área bajo la curva ROC como se ve en la ecuación 6:

$$AUC = \int_0^1 TPR(FPR)\, d(FPR) \tag{6}$$

En términos prácticos, el valor de AUC puede interpretarse como la probabilidad de que el modelo asigne una puntuación más alta a un ejemplo positivo que a uno negativo. Un valor de AUC:

- 0.5 indica un rendimiento equivalente al azar,
- entre 0.6 y 0.7 sugiere un modelo aceptable,
- entre 0.7 y 0.8 indica buen desempeño,
- mayor a 0.8 refleja una capacidad discriminante fuerte.

Esta métrica es especialmente útil cuando las clases están desbalanceadas, ya que no depende de un umbral específico y evalúa la calidad general del ordenamiento que hace el modelo sobre las probabilidades predichas.

3.2 ANTECEDENTES

Existen varios proyectos y estudios que han abordado la predicción de matrícula universitaria en diferentes contextos. Aquí se presentan algunos antecedentes relevantes:

Modelo Matemático para Predecir la Intención de Matrícula:
Este estudio de la Universidad del Norte utiliza redes neuronales artificiales y regresión logística para predecir la intención de matrícula de estudiantes admitidos. El modelo se basa en variables como edad, género, forma de financiación, ICFES, departamento de proveniencia, tipo de colegio, semestre académico, carrera y estrato [17].
Los autores evaluaron distintos enfoques y concluyeron que la regresión logística fue el modelo con mejor desempeño, alcanzando una capacidad predictiva superior a la obtenida por la red neuronal utilizada en el estudio. En particular, la regresión logística logró un nivel de precisión y consistencia adecuado para apoyar la toma de decisiones institucionales, mientras que la red neuronal presentó un rendimiento más inestable debido al tamaño y la naturaleza de los datos.
Este modelo es un ejemplo de cómo las técnicas de aprendizaje automático y análisis estadístico pueden ser aplicadas para mejorar la eficiencia y efectividad de los procesos de admisión en las universidades. La propuesta actual se enfoca en una universidad colombiana diferente, lo que implica características y datos específicos de los aspirantes.

Modelo de Predicción de Deserción Estudiantil, Apoyado en Tecnologías de Data Mining, en un Curso de Primera Matrícula de la Escuela ECBTI de la UNAD:
Este proyecto, desarrollado en la Universidad Nacional Abierta y a Distancia (UNAD), utiliza técnicas de minería de datos y aprendizaje automático como árboles de decisión y redes neuronales para predecir la deserción de estudiantes en cursos de primera matrícula. El objetivo es identificar estudiantes en riesgo de deserción y tomar medidas preventivas [18].

El estudio emplea métricas como la matriz de confusión, sensibilidad, especificidad y curvas ROC para evaluar el desempeño de los modelos. De acuerdo con los análisis presentados, los árboles de decisión se destacaron por su interpretabilidad y por ofrecer un desempeño competitivo al momento de clasificar estudiantes desertores y no desertores. Aunque el trabajo compara diferentes modelos, no se reporta explícitamente un valor único de precisión o un modelo claramente superior, sino que se enfatiza el aporte de las reglas generadas por los árboles para comprender los factores asociados a la deserción. Entre estos factores resaltan la interacción con la plataforma, la entrega de actividades y el seguimiento académico temprano.

Aunque este modelo se centra en la predicción de la deserción estudiantil, utiliza métricas de evaluación aplicables a la propuesta actual como la matriz de confusión y el análisis ROC para validar el desempeño de los modelos.

Modelo Analítico para la Predicción de la Deserción Estudiantil a Nivel de Pregrado en la Universidad Autónoma del Caribe:
Este estudio propone un modelo analítico para predecir la deserción estudiantil en programas de pregrado de la Universidad Autónoma del Caribe. Para ello, se emplearon modelos como árboles de decisión y regresión logística, las cuales permitieron analizar una variedad de factores académicos, sociodemográficos y administrativos asociados al riesgo de abandono [19].

El estudio desarrolló un proceso estructurado que incluyó la recopilación y depuración de datos, la selección de variables relevantes y la construcción de modelos de clasificación orientados a detectar patrones que distinguen a los estudiantes propensos a desertar. Un aspecto destacado del trabajo es la integración de herramientas de análisis visual como Power BI, mediante las cuales se segmentó a la población estudiantil en tres niveles de riesgo: bajo, medio y alto. Esta segmentación facilitó la interpretación de los resultados y permitió a la institución contar con indicadores accionables para la toma de decisiones.

Si bien el estudio utiliza modelos como árboles de decisión y regresión logística —también empleados en la presente investigación— su aporte principal radica en demostrar cómo estas técnicas pueden apoyar la gestión institucional mediante la identificación temprana de estudiantes vulnerables, la priorización de intervenciones y la optimización de recursos en estrategias de permanencia. La relevancia de este antecedente teórico-metodológico se refleja en la aplicabilidad de sus enfoques al contexto del presente proyecto, en el cual los modelos de clasificación también buscan apoyar procesos institucionales críticos, como la predicción de matrícula.

4 ANÁLISIS EXPLORATORIO Y PREPARACIÓN DE LOS DATOS

El presente capítulo describe el proceso de preparación y análisis exploratorio de los datos utilizado como base para el desarrollo de los modelos predictivos. En esta etapa se llevaron a cabo tareas como la depuración, transformación y codificación de las variables, así como la revisión de la estructura, distribución y coherencia del conjunto de datos para cada programa académico. El propósito de este apartado es garantizar que la información utilizada para el entrenamiento de los modelos sea consistente, de calidad y adecuada para los métodos de aprendizaje supervisado, permitiendo identificar patrones iniciales, posibles sesgos y relaciones relevantes entre las variables que influyen en el estado de matrícula de los aspirantes.

4.1 ANALISIS EXPLORATORIO DE LOS DATOS

Se consolidaron los datos de aspirantes de los últimos cinco años de los programas de Derecho, Comunicación Social, Contaduría Pública, Comercio Internacional, Administración de Empresas y Psicología, discriminados por periodo académico e incluyendo datos demográficos, socioeconómicos y resultados de pruebas de estado saber 11 de los aspirantes. En la figura 7 se muestra el total de registros disponibles por programa académico. El número de aspirantes varía significativamente entre los programas. Derecho tiene el mayor número de aspirantes (1530), seguido de Psicología (1224) y Contaduría Pública (989). Comunicación Social tiene el menor número de aspirantes (359).

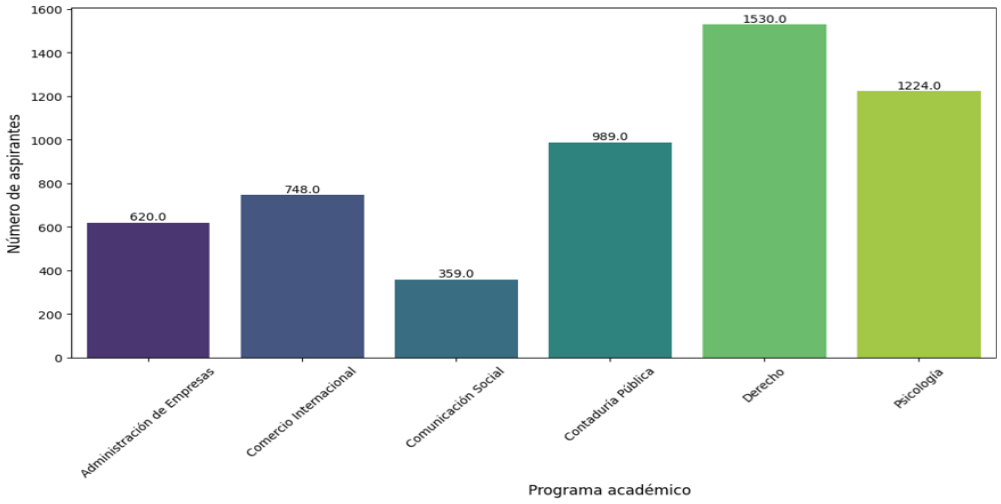

Fig. 7. Distribución de aspirantes por programa académico

Estos datos fueron adquiridos a partir de los registros suministrados por la universidad, específicamente de las bases de datos del sistema transaccional de información académica institucional. Posteriormente, los datos fueron extraídos, consolidados y anonimizados para proteger la identidad de los estudiantes, cumpliendo con las normas de confidencialidad y ética en el manejo de la información. Este procedimiento permitió disponer de un conjunto de datos estructurado y confiable para el desarrollo del análisis predictivo.

El conjunto de datos contiene 15 variables, que fueron preparadas según tipo y características, tal como se describe en adelante:

- **Período:** El período académico al que corresponde la solicitud (por ejemplo, 202402).
- **Programa Académico:** El programa académico al que aspira el estudiante.
- **Modalidad:** La modalidad del programa (presencial o virtual).
- **ID Estudiante:** Un identificador único para cada aspirante.
- **Estado:** El estado de la solicitud (Admisión o Matrícula Financiera).
- **Género:** El género del aspirante (masculino o femenino).
- **Edad:** La edad del aspirante.
- **Estado Civil:** El estado civil del aspirante.
- **Estrato:** El estrato socioeconómico del aspirante.
- **Nivel Estudios:** El nivel de estudios previo del aspirante.
- **Trabaja Actualmente:** Indica si el aspirante trabaja actualmente (sí o no).
- **Fuente Referencia:** La fuente a través de la cual el aspirante se enteró del programa.
- **Forma de Pago:** La posible forma de pago del aspirante.
- **Ciencias Inglés, Lectura Crítica, Matemáticas, Sociales:** Puntuaciones en las áreas de la prueba de estado saber 11.
- **Distancia a la universidad (km):** distancia desde el lugar de residencia del aspirante.

Valores Faltantes

Se analizó el porcentaje de valores faltantes para cada variable por programa académico, con el fin de evaluar la completitud de la información. La figura 8 muestra que las variables con mayor proporción de valores faltantes corresponden a Zona de residencia, Nivel de estudios e Inglés, especialmente en los programas de Derecho y Contaduría Pública, donde los porcentajes superan el 50% en algunos casos.

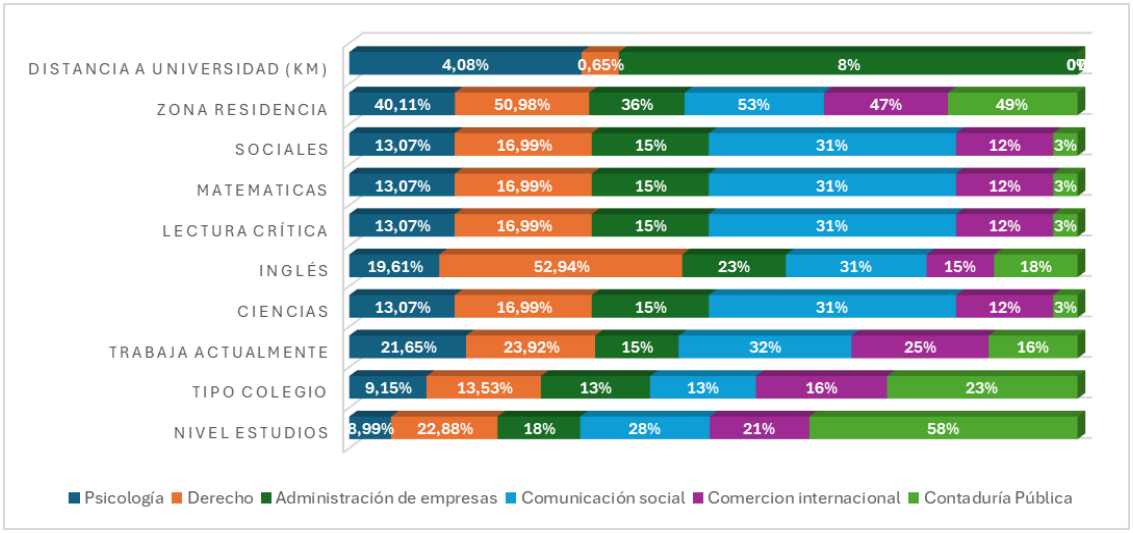

Fig. 8.Valores faltantes por variable y programa académico

Otras variables como Matemáticas, Lectura Crítica, Sociales y Ciencias presentan porcentajes

similares (entre 13% y 17%) en casi todos los programas, mostrando un patrón consistente de información incompleta. En contraste, variables como Distancia a la universidad registran muy pocos valores faltantes, lo que sugiere una mayor confiabilidad en estos datos.

Distribución de los Datos y Valores Atípicos

Se emplearon diagramas de caja (boxplots) para visualizar la distribución de las variables numéricas y detectar valores atípicos. A continuación, se presenta un análisis de los valores atípicos identificados en las variables numéricas para cada programa académico. Cabe resaltar que, dado el dominio trabajado, estos datos atípicos no son descartados en el desarrollo del modelo (por ejemplo: una persona de más de 40 o 50 años inscrita a un programa académico, seguramente se va a matricular, aunque puede representar un dato atípico entre los aspirantes).

Psicología:

- Edad inscripción: En la figura 9 se observan varios valores atípicos en el extremo superior, indicando estudiantes con edades significativamente mayores al promedio de inscripción.
- Ciencias, Inglés, Lectura Crítica, Matemáticas, Sociales: Se aprecian valores atípicos en ambos extremos para las puntuaciones de las pruebas, lo que indica tanto puntajes excepcionalmente bajos como altos.
- Distancia a Universidad (km): Hay una cantidad considerable de valores atípicos en el extremo superior, sugiriendo que algunos estudiantes de Psicología residen a distancias muy lejanas de la universidad.

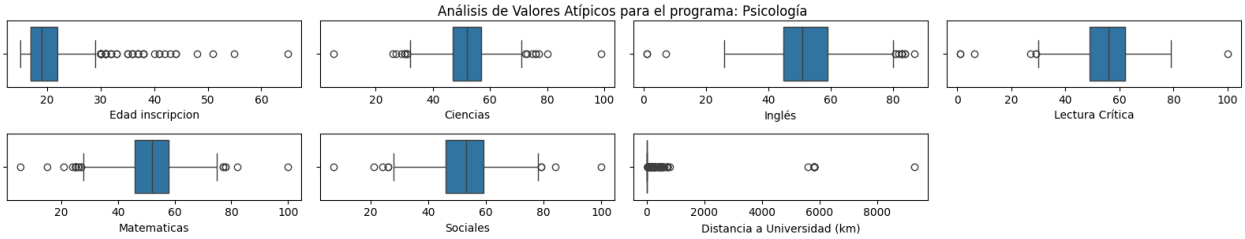

Fig. 9. Valores atípicos Psicología

Derecho:

- Edad inscripción: En la figura 10 se observan varios valores atípicos en el extremo superior.
- Ciencias, Inglés, Lectura Crítica, Matemáticas, Sociales: Valores atípicos en ambos extremos de las puntuaciones.
- Distancia a Universidad (km): Una cantidad notable de valores atípicos en el extremo superior, similar al programa de Psicología.

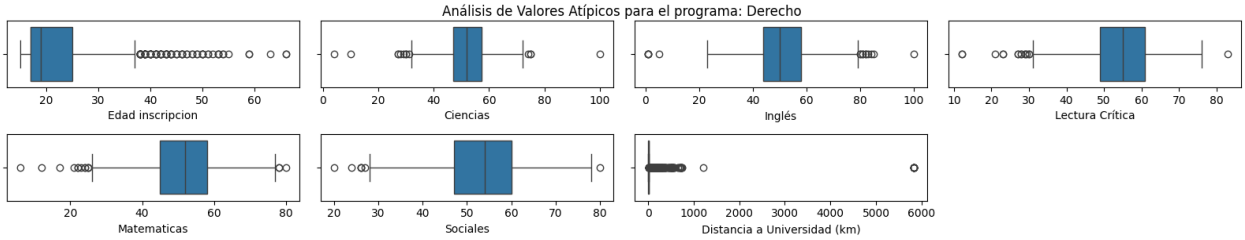

Fig. 10. Valores atípicos Derecho

Administración de Empresas:

- Edad inscripción y Distancia a Universidad (km): En la figura 11 se observan valores atípicos en el extremo superior.
- Ciencias, Inglés, Lectura Crítica, Matemáticas, Sociales: Se aprecian valores atípicos en ambos extremos de las puntuaciones.

Comunicación Social:

- Edad inscripción y Distancia a Universidad (km): En la figura 11 se observan algunos valores atípicos en el extremo superior.
- Ciencias, Inglés, Lectura Crítica, Matemáticas, Sociales: Valores atípicos en ambos extremos de las puntuaciones.

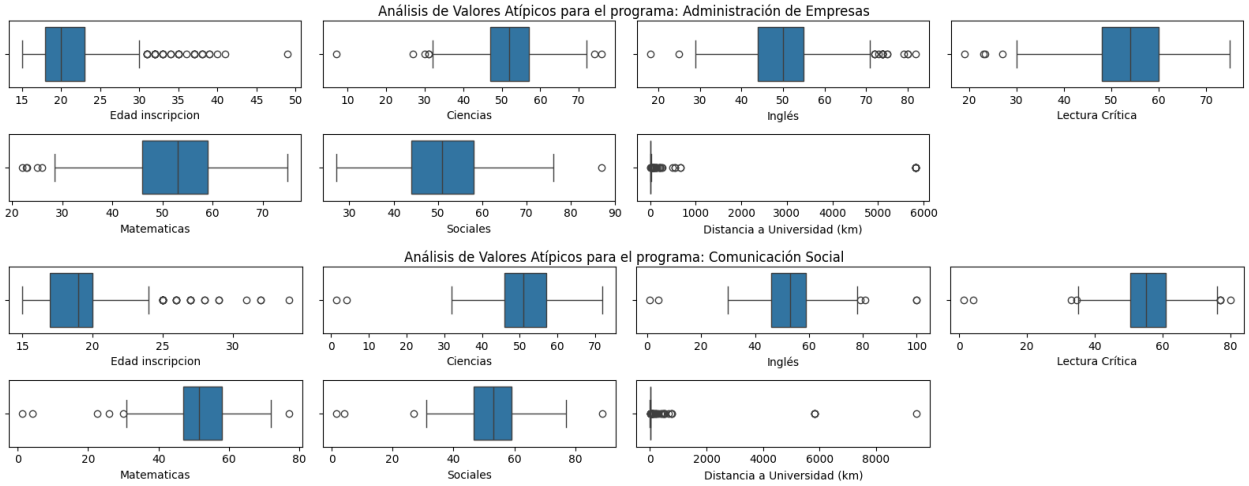

Fig. 11.Valores atípicos Administración de Empresas y Comunicación Social

Comercio Internacional y Contaduría Pública:

- Edad inscripción y Distancia a Universidad (km): En la figura 12 y 13 se observan valores atípicos en el extremo superior.
- Ciencias, Inglés, Lectura Crítica, Matemáticas, Sociales: Se aprecian valores atípicos en ambos extremos de las puntuaciones.

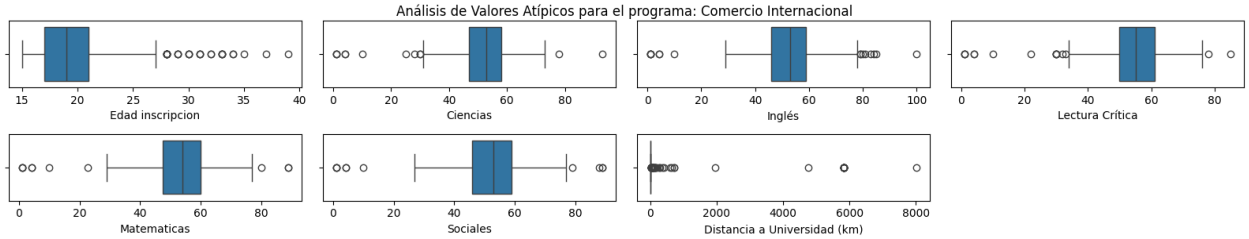

Fig. 12. Valores atípicos Comercio Internacional

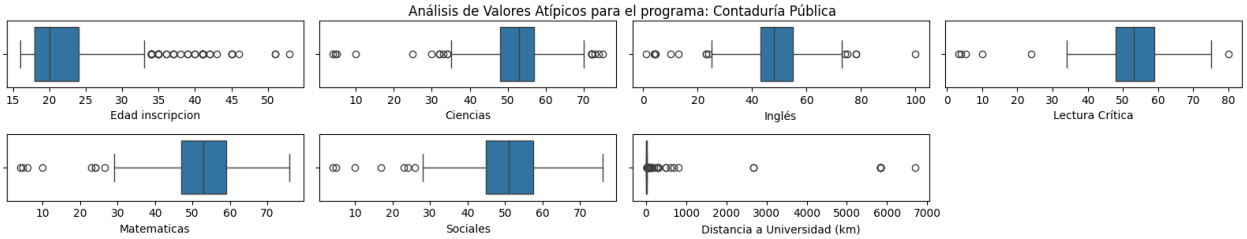

Fig. 13. Valores atípicos Contaduría Pública

En general, la presencia de valores atípicos en la 'Edad inscripción' y la 'Distancia a Universidad (km)' es consistente en la mayoría de los programas. La puntuación de las pruebas también muestra valores atípicos en varios programas, lo que sugiere una variabilidad en el rendimiento académico de los aspirantes.

Cabe resaltar que, en la distribución de los datos, algunas de las variables muestran un rango más amplio de valores, sugiriendo que pueden tener una mayor relevancia para el modelo predictivo (Edad, Fuente Referencia, Forma de Pago, Puntuaciones en la prueba).

Correlaciones entre las variables numéricas

Se generaron mapas de calor a partir de matrices de correlación para evaluar la relación lineal entre las variables numéricas. A continuación, se presenta un análisis de las correlaciones entre las variables numéricas para cada programa académico.

- Psicología: En la figura 14 se observan fuertes correlaciones positivas entre las pruebas Ciencias y Sociales (0.71), Lectura Crítica y Sociales (0.70). Correlación negativa moderada entre Edad inscripción e Inglés (-0.24) y Matemáticas (-0.21).

- Derecho: En la figura 14 se observan fuertes correlaciones positivas entre las pruebas Ciencias y Sociales (0.65), Lectura Crítica y Sociales (0.67). Correlación negativa moderada entre Edad inscripción y Lectura Crítica (-0.31) e Inglés (-0.37).

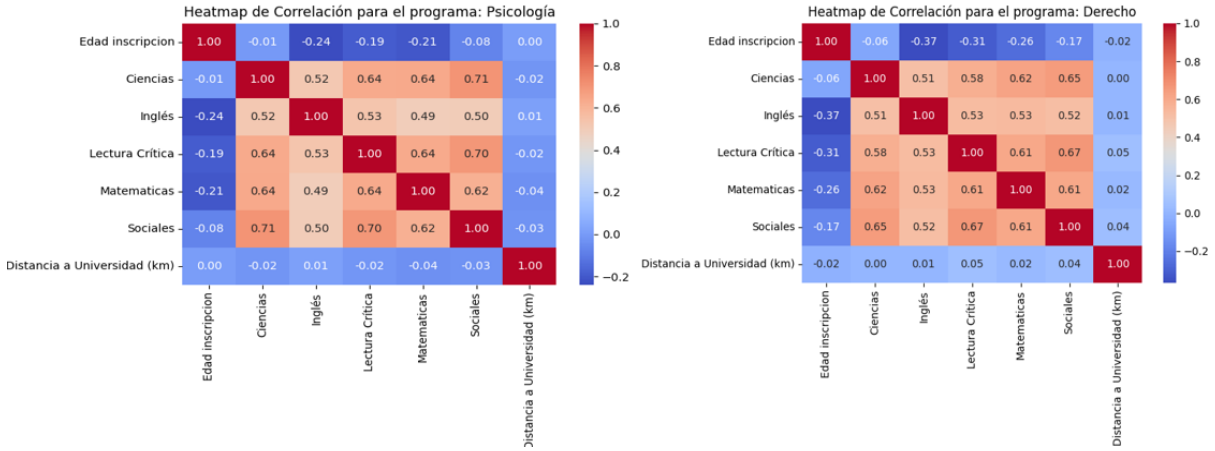

Fig. 14. Correlaciones para Psicología y Derecho

- Administración de Empresas: En la figura 15 se observan fuertes correlaciones positivas entre las pruebas Ciencias y Sociales (0.62), Lectura Crítica y Sociales (0.66). Correlación negativa moderada entre Edad inscripción y Matemáticas (-0.30).

- Comunicación Social: En la figura 15 se observan fuertes correlaciones positivas entre las pruebas Ciencias y Sociales (0.70), Lectura Crítica y Sociales (0.73). Correlación negativa moderada entre Edad inscripción e Inglés (-0.21) y Lectura Crítica (-0.20).

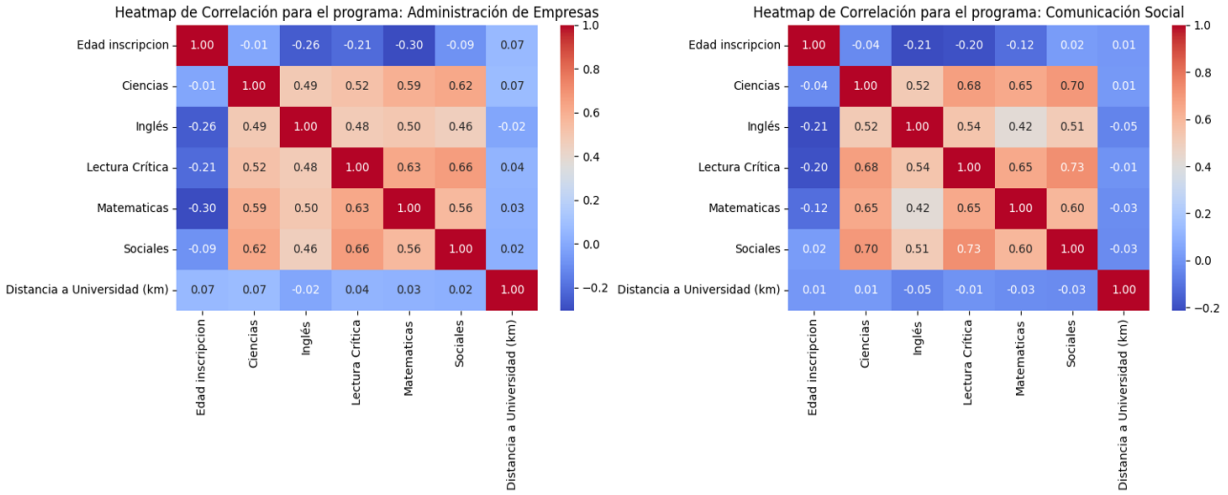

Fig. 15. Correlaciones para Administración de Empresas y Comunicación Social

- Comercio Internacional: En la figura 16 se observan fuertes correlaciones positivas entre las pruebas Ciencias y Sociales (0.72), Lectura Crítica y Sociales (0.74). Correlación negativa moderada entre Edad inscripción e Inglés (-0.22) y Lectura Crítica (-0.21).

- Contaduría Pública: En la figura 16 se observan fuertes correlaciones positivas entre las pruebas Ciencias y Sociales (0.67), Lectura Crítica y Sociales (0.66). Correlación negativa moderada entre Edad inscripción y Matemáticas (-0.36) e Inglés (-0.32).

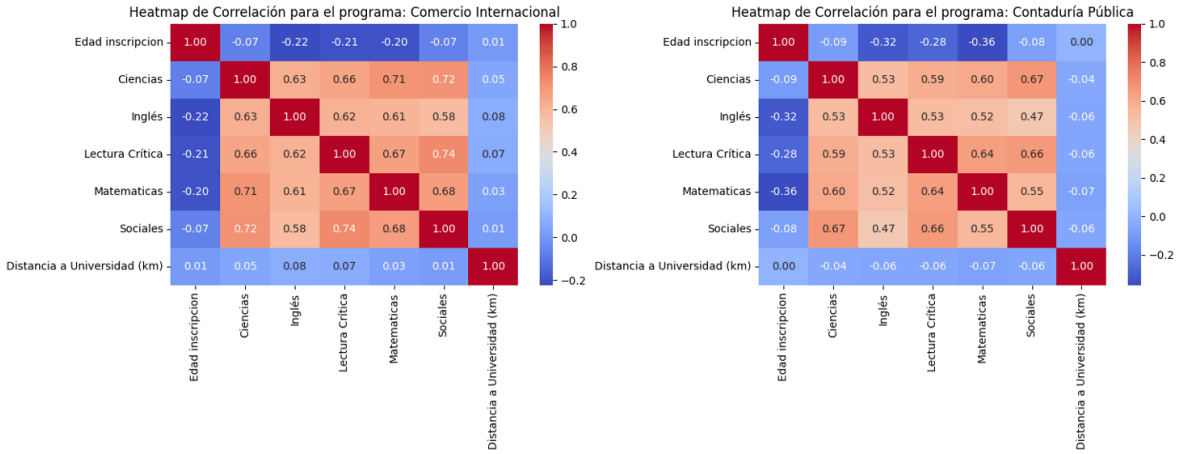

Fig. 16. Correlaciones para Comercio Internacional y Contaduría Pública

El análisis de correlación por programa académico respalda las relaciones observadas a nivel general del conjunto de datos, destacando la interrelación de las puntuaciones de las pruebas Saber 11 y una moderada relación inversa entre la edad de inscripción y el rendimiento en estas pruebas.

Relación entre Variables Categóricas y Estado de Matrícula

Para las variables categóricas, se utilizó la prueba de independencia Chi-cuadrado y el coeficiente V de Cramer para medir la fuerza de la asociación con el estado de matrícula. En la tabla 1 los resultados consolidados del test Chi-cuadrado y V de Cramer:

- Psicología y Comunicación Social: Se encontró una asociación estadísticamente significativa, aunque débil (V de Cramer = 0.23), entre la 'Posible Forma de Pago' y el estado de matrícula. Las otras variables categóricas analizadas no mostraron una asociación significativa con el estado de matrícula en este programa.

Programa Académico	Variable Categórica	Chi-squared Statistic	P-value	Cramer's V
Psicología	Posible Forma de Pago	62.782692	1.322248e-10	0.226480
Derecho	Posible Forma de Pago	60.442274	3.816673e-10	0.198758
Administración de Empresas	Nivel Estudios	27.875492	9.418600e-03	0.213945
Administración de Empresas	Posible Forma de Pago	46.965717	1.556633e-07	0.275229
Comunicación Social	Posible Forma de Pago	19.071422	1.448231e-02	0.230486
Comercio Internacional	Tipo Colegio	6.816064	9.034148e-03	0.103850
Comercio Internacional	Posible Forma de Pago	27.035065	6.971969e-04	0.190113
Comercio Internacional	Estrato	11.303992	2.335185e-02	0.122932
Contaduría Pública	Nivel Estudios	30.739109	9.518385e-03	0.181609
Contaduría Pública	Posible Forma de Pago	30.082241	2.044181e-04	0.174404

Tabla 1: Resultados consolidados del test Chi-cuadrado y V de Cramer

- Derecho y Administración de Empresas: Ninguna de las variables categóricas mostró una asociación estadísticamente significativa con el estado de matrícula en este programa.
- Comercio Internacional: Se encontraron asociaciones estadísticamente significativas, aunque débiles, entre el 'Tipo Colegio' (V de Cramer = 0.10) y el 'Estrato' (V de Cramer = 0.12) y el estado de matrícula.
- Contaduría Pública: Se encontraron asociaciones estadísticamente significativas, aunque débiles, entre el 'Nivel Estudios' (V de Cramer = 0.18) y la 'Posible Forma de Pago' (V de Cramer = 0.17) y el estado de matrícula.

La 'Posible Forma de Pago' es la variable categórica que más consistentemente muestra una asociación estadísticamente significativa con el estado de matrícula, aunque con una fuerza de asociación débil, en programas como Psicología, Comunicación Social y Contaduría Pública.

En todos los casos donde se identificaron asociaciones significativas, la fuerza de la asociación, medida por el coeficiente V de Cramer, fue consistentemente débil. Esto sugiere que, si bien estas variables pueden tener alguna influencia en el estado de matrícula, no son predictores determinantes por sí solos en el contexto de este análisis bivariado.

Evolución Temporal del Estado de Matrícula

Se analizó la proporción de estudiantes admitidos en estado 'No matriculados' (admisión) y 'Matriculados' a lo largo de diferentes períodos para cada programa académico. A continuación, se presentan algunas observaciones clave sobre estas tendencias temporales:

- **Administración de Empresas**: En la figura 17 se observa una variabilidad en las proporciones a lo largo de los períodos. Hay picos en la proporción de matriculados en ciertos períodos (ej. 202001, 202201, 202301, 202401), que corresponden a los períodos de inicio de año, y caídas en otros (ej. 202002, 202202, 202302, 202402), que corresponden a los segundos períodos del año con menor actividad de matrícula.

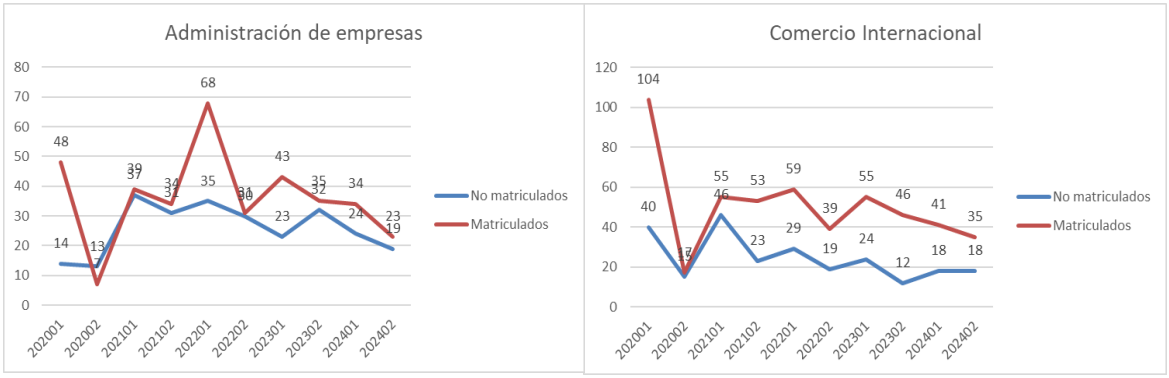

Fig. 17.Evolución temporal Comercio Internacional y Administración de Empresas

- **Comercio Internacional**: Similar a Administración de Empresas, muestra fluctuaciones con picos de matriculados en ciertos períodos (ej. 202001, 202102, 202302, 202401, 202402). La proporción de matriculados parece ser generalmente más alta que la de no matriculados en la mayoría de los períodos.
- **Comunicación Social**: Presenta fluctuaciones notables. En la figura 18 se observa un pico de matriculados en 202001, una caída en 202002, un repunte en 202201 y otro pico alto en 202302, seguido de una caída en 202402.
- **Contaduría Pública**: Muestra una tendencia con proporciones de matriculados generalmente superiores a las de no matriculados, aunque con variaciones periódicas. Se aprecian picos de matrícula en 202001, 202101, 202202, 202401 y 202402.

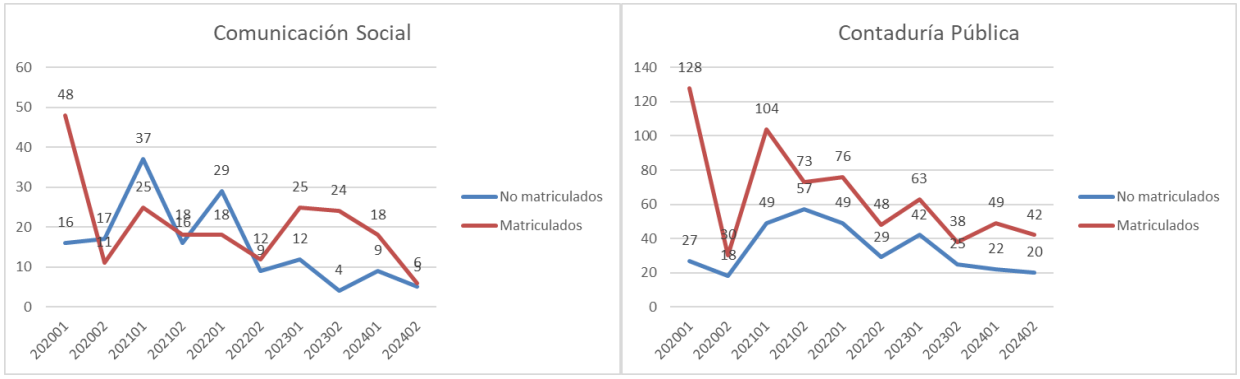

Fig. 18. Evolución temporal Contaduría Pública y Comunicación Social

- **Derecho**: Muestra fluctuaciones similares a otros programas, con picos de matriculados en 202001, 202101, 202201, 202301 y 202402. En la figura 19 se observa que la proporción entre ambos estados parece ser variable a lo largo del tiempo.
- **Psicología**: Presenta fluctuaciones con picos de matriculados en 202001, 202101, 202201, 202301, 202401 y 202402. En la figura 19 se observa la proporción entre no matriculados y matriculados parece alternar entre períodos.

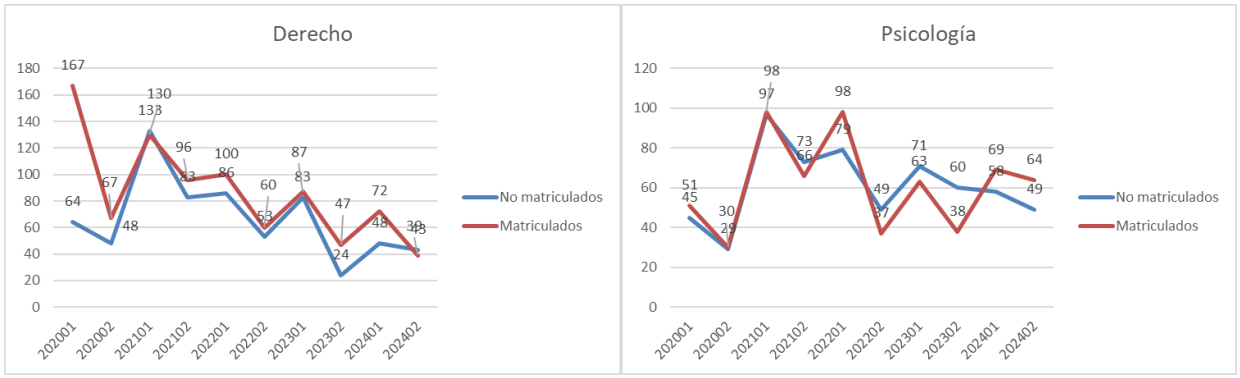

Fig. 19. Evolución temporal Psicología y Derecho

En la mayoría de los programas, se observan fluctuaciones en las proporciones de no matriculados y matriculados a lo largo del tiempo. Esto es esperable debido a los ciclos de admisión y matrícula en una institución universitaria.

El periodo de análisis incluye la contingencia asociada a la pandemia por COVID-19, cuyo impacto se evidencia en las gráficas de todos los programas académicos, particularmente en el periodo 2020-2, donde se observan variaciones atípicas en el comportamiento de la matrícula. Este contexto excepcional pudo haber modificado los patrones históricos de decisión de los aspirantes, introduciendo dinámicas que no son capturadas por las variables tradicionales utilizadas en el estudio. En conjunto, estas condiciones extraordinarias y la ausencia de variables que representen de manera directa la intención de matrícula pueden limitar la capacidad de los modelos para alcanzar niveles más altos de precisión predictiva con la información disponible.

Desbalance de Clases

La variable 'Estado' tiene dos clases: 'no matriculados' (también mencionada como Admisión) y 'Matrícula Financiera'. La tabla 2 presenta el balance acumulado entre los estados de no matriculados y Matrícula financiera para cada programa académico, considerando todos los aspirantes admitidos registrados entre los periodos 2020-01 y 2024-02. Se incluyen los valores absolutos, la diferencia en números y el porcentaje relativo.

Programa Académico	No matriculados	Matrícula Financiera	Diferencia Absoluta	Diferencia % (sobre Admisión)
Administración de Empresas	258	362	104	40,3 %
Comunicación Social	154	205	51	33,1 %
Psicología	610	614	4	0,7 %
Comercio Internacional	244	504	260	106,6 %
Contaduría Pública	338	651	313	92,6 %
Derecho	665	865	200	30,1 %
Totales	2269	3201	+932	41,1 %

Tabla 2: Desbalance de clases

El análisis acumulado del balance entre clases evidencia diferencias significativas en varios programas académicos. En Administración de Empresas, los matriculados superan a los no matriculados en un 40,3 % (362 vs. 258), mientras que en Comunicación Social el incremento es del 33,1 % (205 vs. 154). El programa de Psicología muestra un comportamiento más estable, con apenas un 0,7 % de aumento (614 vs. 610). Los casos más destacados se observan en Comercio Internacional, donde la matrícula duplica a los no matriculados con un incremento del 106,6 % (504 vs. 244), y en Contaduría Pública, donde el aumento alcanza un 92,6 % (651 vs. 338). Finalmente, en Derecho se presenta un incremento moderado del 30,1 % (865 vs. 665).

Aunque no hay un desbalance extremo en la mayoría de los programas (las proporciones están alrededor de 60/40 o 50/50), el desbalance de clases fue considerado durante la etapa de modelado para garantizar la precisión de las predicciones.

Selección de Variables

Se aplicaron varios métodos de selección de características para identificar las variables más relevantes en la predicción de la matrícula. Los métodos de filtrado, como Chi-cuadrado, F-classif e Información Mutua, evaluaron la relación entre cada variable individual y la variable objetivo, revelando la importancia de las variables forma de pago, edad de inscripción y pruebas Saber 11.

Luego del análisis realizado, se seleccionaron las variables de puntajes en *matemáticas*, *ciencias*, *inglés, lectura crítica y sociales, estrato, posible forma de pago, trabaja actualmente, edad de inscripción, distancia a la universidad, periodo* y *fuente de referencia* para el entrenamiento de los modelos, al considerarlas de mayor influencia en la decisión de matrícula.

4.2 PREPARACION DE LOS DATOS

Se realizó la carga inicial de los datos y el filtrado por programa académico, asegurando un análisis independiente, debido a que se consideró la existencia de posibles patrones específicos. A continuación, se detallan los aspectos abordados en la preparación de los datos para cada programa académico:

Ingeniería de características
Para abordar la alta cardinalidad presente en variables categóricas como Fuente de Referencia y Posible Forma de Pago, se implementó un proceso de agrupación por categorías con comportamientos similares. Esta estrategia permitió reducir la dimensionalidad del espacio de características, mejorar la estabilidad de los patrones y evitar la generación excesiva de variables dummy, lo cual puede afectar negativamente el rendimiento de algunos modelos.

Asimismo, la variable Periodo fue transformada en dos componentes: Año y Semestre. Esta desagregación facilitó la identificación de patrones temporales relevantes, como variaciones en las tasas de matrícula entre distintos ciclos académicos, lo que enriqueció la capacidad predictiva de los modelos entrenados.

Finalmente, se excluyeron del análisis las variables como Programa académico, Modalidad, ID del estudiante, Género, Estado civil y Nivel de estudios, debido a que no aportaban información significativa o podían introducir ruido en el proceso de entrenamiento. Esta depuración contribuyó a mejorar la calidad del conjunto de datos y a optimizar el desempeño de los modelos de clasificación.

Codificación de la Variable Objetivo:
La codificación de la variable objetivo se realizó en formato binario, diferenciando entre estudiantes que se matricularon y aquellos que no lo hicieron (1 para 'Matrícula Financiera' y 0 para 'Admisión'). Esta transformación es indispensable para los algoritmos de clasificación empleados en el proyecto.

División de los datos:
La división de los datos en conjuntos de entrenamiento y prueba se realizó dependiendo del tamaño del dataset del programa académico. En la mayoría de los programas, se utilizó un 80 % de los datos para entrenamiento y 20 % para prueba, asegurando que los modelos pudieran aprender de la mayor cantidad de información disponible y que el conjunto de prueba fuera suficientemente representativo para evaluar el desempeño. Solo en el caso del programa de Comunicación Social que contaba con menos de 500 registros, se optó por una división 70/30 %, con el fin de contar con un número adecuado de instancias en el conjunto de prueba.
En ambos escenarios se implementó una división estratificada de los datos, lo que permitió mantener la proporción original de las clases en ambos subconjuntos y garantizar una evaluación más confiable del modelo, especialmente ante el desbalance de clases observado.

Imputación de valores faltantes:
En relación con el manejo de valores faltantes, se aplicó una estrategia combinada: en el caso de la variable Trabaja Actualmente se utilizó imputación condicional basada en la edad, mientras que para otras variables se imputaron utilizando la mediana (para numéricas) o la moda (para categóricas) de la clase correspondiente en el conjunto de entrenamiento y de prueba.

Tratamiento de valores atípicos
Se identificaron valores atípicos en variables como la edad de inscripción y algunos puntajes académicos. No obstante, debido a la naturaleza del dominio educativo, estos valores no fueron eliminados del conjunto de datos. En este contexto, ciertos registros que podrían considerarse atípicos estadísticamente —por ejemplo, aspirantes mayores de 40 o 50 años— representan perfiles reales dentro del proceso de admisión. Además, estos casos pueden aportar información valiosa al modelo, ya que sus patrones de comportamiento, incluida una mayor probabilidad de matrícula, constituyen señales útiles para la predicción.

Codificación de variables categóricas:
Las variables categóricas fueron transformadas utilizando el método One-Hot Encoding para la mayoría de los modelos de clasificación. Esta técnica permitió representar las categorías de manera binaria, eliminando la primera categoría de cada variable con el fin de evitar problemas de multicolinealidad y garantizar que los algoritmos pudieran procesar adecuadamente estas variables sin asumir relaciones de orden o magnitud entre sus categorías.

En el caso del modelo TabTransformer, se empleó el método OrdinalEncoder, dado que es compatible con la técnica de balanceo SMOTENC utilizada durante el entrenamiento. Esta codificación asigna un valor entero a cada categoría, lo que permite al modelo generar embeddings que representan de forma adecuada las relaciones entre las categorías dentro de su arquitectura basada en transformadores.

Escalado:
Para modelos sensibles a la escala como Regresión Logística, SVM, MLPClassifier y Tab Transformer, se aplicó StandardScaler a las características numéricas. Al estandarizar los datos se asegura que todas las características contribuyan de manera equitativa al proceso de aprendizaje, lo que estabiliza el entrenamiento y a menudo mejora el rendimiento final del modelo.

Balanceo de clases:
Para abordar el desbalance de clases, que, aunque en general no fue extremo si puede afectar el aprendizaje, se aplicaron diferentes estrategias de balanceo según las capacidades del modelo.
En aquellos que soportan ponderación de clases, como Regresión Logística, SVM y Random Forest, se utilizó el parámetro class_weight='balanced', lo que proporciona un balance de clases sin alterar los datos, con menos riesgo de sobreajuste y mayor eficiencia durante el entrenamiento. En el caso del modelo XGBoost, se calculó el ratio real de desbalance para el parámetro scale_pos_weight.

Para los modelos que no soportan esta opción, como los modelos MLP Classifier y TabTransformer, se aplicó SMOTENC (Synthetic Minority Over-sampling Technique for Nominal and Continuous variables), una técnica de sobre muestreo que genera nuevas instancias sintéticas de la clase minoritaria respetando la naturaleza categórica de algunas variables. Esta técnica se aplicó en el conjunto de entrenamiento o dentro de la validación cruzada.

5 ENTRENAMIENTO Y EVALUACIÓN DE LOS MODELOS DE CLASIFICACIÓN

Este capítulo describe el proceso de entrenamiento y evaluación de los modelos de clasificación aplicados para la predicción de matrícula de los aspirantes. Se presentan los modelos implementados y los criterios empleados para comparar su desempeño. Asimismo, se detallan los procedimientos de validación, las métricas seleccionadas y la configuración de los experimentos, con el propósito de garantizar resultados reproducibles y una evaluación objetiva de cada modelo.

Para cada programa académico se desarrolló un proceso completo de entrenamiento, ajuste y evaluación de modelos de clasificación binaria utilizando los conjuntos de datos procesados según lo expuesto en el anterior apartado. Se consideraron diferentes enfoques de modelado, incluyendo una línea base con DummyClassifier, modelos de ensamble como Random Forest y XGBoost, modelos lineales como la Regresión Logística, redes neuronales como MLPClassifier y TabTransformer (Hugging face), y modelos basados en Kernel como SVM.

Entrenamiento de los modelos

Para cada programa académico, se implementó un proceso sistemático de entrenamiento que inició con una evaluación preliminar utilizando los parámetros por defecto de cada algoritmo. Posteriormente, se llevó a cabo un ajuste progresivo de hiperparámetros con el fin de optimizar el rendimiento de los modelos de acuerdo con las características particulares de cada conjunto de datos.

En todos los programas, la fase inicial consistió en una búsqueda aleatoria (RandomizedSearchCV) para explorar de manera eficiente un espacio amplio de configuraciones posibles en cada modelo, ajustadas a las características del conjunto de datos (tamaño, nivel de desbalance y número de variables). Luego, para cada programa se realizaron hasta tres rondas sucesivas de refinamiento mediante GridSearchCV. En cada ronda, las rejillas de búsqueda se construyeron a partir de los valores encontrados en la ronda anterior. De esta manera, el proceso avanzó de exploraciones amplias hacia ajustes cada vez más específicos, adaptados al comportamiento observado en los datos de cada programa.

Este enfoque permitió una búsqueda progresiva y controlada, mejorando la precisión de los valores óptimos encontrados en algunos modelos. En el caso del modelo TabTransformer, el ajuste se efectuó de forma manual mediante la modificación iterativa de parámetros críticos como la dimensión del embedding, la profundidad de las capas de atención, la tasa de aprendizaje y el número de épocas, evaluando el impacto de cada configuración a través de validación cruzada estratificada.

En resumen, aunque la metodología general fue la misma para todos los programas, en cada uno se realizó un proceso de ajuste particular basado en el tamaño y la distribución de cada conjunto. Algunos modelos mostraron mejoras significativas frente a sus versiones por defecto, lo que

subraya el valor de la exploración sistemática del espacio de hiperparámetros.

Evaluación del desempeño

Para la evaluación del desempeño de los modelos se emplearon métricas de clasificación ampliamente aceptadas, entre las que se destacan la precisión (precision), que mide la proporción de predicciones positivas correctas; el recall (sensibilidad), que indica la capacidad del modelo para identificar correctamente las observaciones positivas; y el f1-score, que combina ambas medidas en un solo indicador balanceado. Estas métricas se calcularon tanto para cada clase como en promedio ponderado, permitiendo una evaluación equitativa en presencia de posibles desbalances de clases. Adicionalmente, se utilizó la métrica ROC AUC (Área Bajo la Curva ROC) como criterio principal de comparación entre modelos, debido a su capacidad para medir la discriminación global del modelo sin depender de un umbral específico.

De acuerdo con el procedimiento descrito, se presenta a continuación la optimización de los hiperparámetros y los resultados obtenidos por los modelos entrenados para cada uno de los programas académicos.

5.1 ANALISIS DE RESULTADOS PARA EL PROGRAMA ADMINISTRACIÓN DE EMPRESAS

El conjunto de datos del programa de Administración de Empresas cuenta con 620 registros, distribuidos en 362 casos de “Matrícula” (58.4%) y 258 de “Admisión” (41.6%), mostrando un leve desbalance a favor del estado de matrícula. Dado el número limitado de registros, el ajuste de los algoritmos priorizó la búsqueda de configuraciones que garantizaran la generalización. En este sentido, la tabla 3 consolida la configuración óptima resultante de la búsqueda de hiperparámetros, evidenciando una tendencia hacia arquitecturas restringidas para prevenir el sobreajuste (overfitting).

Los modelos basados en árboles como Random Forest y XGBoost maximizaron su rendimiento con profundidades muy bajas (max_depth de 3 y 2, respectivamente) y, en el caso de XGBoost, con una fuerte regularización (L1 y L2). Asimismo, se destaca la importancia del manejo del desbalance de clases, observado en la selección del parámetro class_weight: balanced tanto en Random Forest como en SVM, donde este último logró un alto desempeño (0.688) utilizando un kernel sigmoid. El modelo ganador, TabTransformer, y el MLPClassifier convergieron hacia configuraciones ligeras (pocas capas y dimensiones reducidas), mientras que la Regresión Logística obtuvo resultados competitivos con los parámetros por defecto. En el anexo 1 se muestra la evolución del proceso de optimización de hiperparámetros realizado, evidenciando una progresiva reducción y refinamiento del espacio de búsqueda con el objetivo de equilibrar la precisión del ajuste y el tiempo de cómputo.

Modelo	Ronda	ROC AUC	Grilla de parámetros utilizada	Mejores parámetros
RandomForest	**3**	**0,666**	n_estimators': [200,250, 290], max_depth': [3,4, 5], min_samples_split': [4, 6, 8], min_samples_leaf': [1, 2, 3], criterion': ['entropy'], max_features': ['log2'], bootstrap': [False], class_weight': ['balanced']	bootstrap: False, class_weight: balanced, criterion: entropy, max_depth: 3, max_features: log2, min_samples_leaf: 1, min_samples_split: 4, n_estimators: 200
XGBoost	**1**	**0,652**	n_estimators': [100, 200, 300, 400], max_depth': [2, 3, 4, 5], learning_rate': [0.01, 0.05, 0.1, 0.2], subsample': [0.7, 0.8, 0.9, 1.0], colsample_bytree': [0.6, 0.8, 1.0], gamma': [0, 0.1, 0.3, 0.5], min_child_weight': [1, 3, 5, 7], reg_lambda': [0.5, 1, 2], reg_alpha': [0, 0.1, 0.5]	subsample: 1.0, reg_lambda: 2, reg_alpha: 0.5, n_estimators: 200, min_child_weight: 5, max_depth: 2, learning_rate: 0.1, gamma: 0.3, colsample_bytree: 0.8
Regresión Logística	**0**	**0,684**	Parametros por defecto del modelo	penalty='l2', dual=False, tol=1e-4, C=1.0, fit_intercept=True, intercept_scaling=1, class_weight=None, random_state=None, solver='lbfgs', max_iter=100, multi_class='auto', verbose=0, warm_start=False, n_jobs=None, l1_ratio=None
MLP Classifier	**3**	**0,651**	hidden_layer_sizes': [(80,), (100,), (120,), (100, 50)], activation': ['relu', 'tanh'], alpha': [0.00005, 0.0001, 0.0005, 0.001], learning_rate_init': [0.001, 0.0005, 0.0003], solver': ['adam'], batch_size': [32, 64], early_stopping': [True]	'activation': 'relu', 'alpha': 0.001, 'batch_size': 32, 'early_stopping': True, 'hidden_layer_sizes': (100,), 'learning_rate_init': 0.001, 'solver': 'adam'
SVM	**1**	**0,688**	kernel': ['rbf', 'poly', 'sigmoid'], C': uniform(0.1, 10), gamma': uniform(1e-4, 1e-1), degree': randint(2, 5), class_weight': ['balanced']	'C': 6.07, 'class_weight': 'balanced', 'degree': 3, 'gamma':0.016, 'kernel': 'sigmoid'
TabTransformer	**2**	**0,690**	No aplica.	"dim": 16, "depth": 2, "heads": 4, "attn_dropout": 0.3, "ff_dropout": 0.3, "mlp_hidden_mults": (4, 2), "epochs": 40, "batch_size": 32, "learning_rate": 5e-4

Tabla 3: Mejores hiperparámetros para cada modelo- Administración de Empresas

En la tabla 4 se observa un desempeño moderado de los modelos, con valores de ROC AUC que oscilan entre 0.47 y 0.70, reflejando una capacidad discriminativa aceptable pero aún mejorable.

Modelo	Ronda	ROC AUC	Clase 1 Matricula			Clase 0 Admisión		
			Precision	Recall	F1-score	Precision	Recall	F1-score
DummyClassifier		0,477	0,562	0,569	0,566	0,392	0,385	0,388
RandomForest	0	0,637	0,646	0,736	0,688	0,548	0,442	0,489
	1	0,657	0,729	0,597	0,657	0,554	0,692	0,615

	2	0,653	0,707	0,569	0,631	0,530	0,673	0,593
	3	**0,666**	**0,722**	**0,542**	**0,619**	**0,529**	**0,712**	**0,607**
XGBoost	0	0,635	0,671	0,736	0,702	0,578	0,500	0,536
	1	**0,652**	**0,581**	**0,944**	**0,720**	**0,429**	**0,058**	**0,102**
	2	0,651	0,586	0,944	0,723	0,500	0,077	0,133
	3	0,651	0,586	0,944	0,723	0,500	0,077	0,133
Regresión Logística	**0**	**0,684**	**0,633**	**0,792**	**0,704**	**0,559**	**0,365**	**0,442**
	1	0,678	0,774	0,569	0,656	0,563	0,769	0,650
	2	0,679	0,759	0,569	0,651	0,557	0,750	0,639
	3	0,679	0,759	0,569	0,651	0,557	0,750	0,639
MLP Classifier	0	0,648	0,671	0,681	0,676	0,549	0,539	0,544
	1	0,626	0,616	0,625	0,621	0,471	0,462	0,466
	2	0,608	0,620	0,431	0,508	0,446	0,635	0,524
	3	**0,651**	**0,694**	**0,597**	**0,642**	**0,532**	**0,635**	**0,579**
SVM	0	0,643	0,618	0,764	0,683	0,514	0,346	0,414
	1	**0,688**	**0,706**	**0,667**	**0,686**	**0,571**	**0,615**	**0,593**
	2	0,688	0,690	0,681	0,685	0,566	0,577	0,571
	3	0,688	0,690	0,681	0,685	0,566	0,577	0,571
TabTransformer	0	0,692	0,710	0,611	0,657	0,548	0,654	0,597
	1	0,676	0,733	0,611	0,667	0,563	0,692	0,621
	2	**0,690**	**0,705**	**0,597**	**0,647**	**0,540**	**0,654**	**0,591**

Tabla 4: Rendimiento de los modelos- Administración de Empresas

Entre los modelos evaluados, el TabTransformer (Ronda 2) alcanzó el mejor resultado con un ROC AUC de 0.69, superando por una mínima diferencia a los demás algoritmos tradicionales como SVM y Regresión Logística, que se ubicaron en torno al 0.68. Por su parte, los modelos clásicos como Random Forest, XGBoost y MLP mostraron un rendimiento competitivo, pero ligeramente inferior, situándose entre 0.64 y 0.67 en ROC AUC.
Finalmente, el DummyClassifier confirmó la validez del análisis al presentar el valor más bajo (0.477), evidenciando que los modelos entrenados sí lograron aprender patrones útiles más allá del azar.

Los resultados muestran que la arquitectura TabTransformer, junto con un ajuste adecuado de hiperparámetros, ofrece el mejor equilibrio entre precisión, sensibilidad y capacidad de generalización en el conjunto de datos del programa de Administración de empresas, por lo cual fue seleccionado para el desarrollo del prototipo de predicción.

5.2 ANALISIS DE RESULTADOS PARA EL PROGRAMA DE COMERCIO INTERNACIONAL

El conjunto de datos del programa de Comercio Internacional contiene 748 registros, con 504 casos de “Matrícula” (67.38%) y 244 casos de “Admisión” (32.62%), lo que evidencia un desbalance moderado entre las clases. Se aplicaron las mismas tareas de depuración, transformación y codificación de variables que en los demás programas, siguiendo los procedimientos descritos en la sección 4.2. El anexo 2 muestra las grillas de parámetros utilizadas en las rondas de ajuste de los modelos.

La Tabla 5 detalla la configuración final de los modelos, donde se observan valores para maximizar la capacidad de generalización. El XGBoost, que alcanzó el mejor desempeño (0.634), quedó con una tasa de aprendizaje conservadora (0.019) combinada con penalizaciones L1 y L2 significativas (reg_alpha: 0.586, reg_lambda: 2.43) para controlar la varianza y evitar el sobreajuste en un dataset de tamaño moderado. Un patrón similar de restricción se observa en la Regresión Logística, donde el parámetro de regularización inversa C fue bajo (0.09), forzando un modelo lento mediante penalización Lasso (L1).

Modelo	Ronda	ROC AUC	Grilla de parámetros utilizada	Mejores parámetros
RandomForest	**1**	**0,621**	n_estimators': [100, 200, 300, 400, 500], max_depth': [None, 5, 10, 15, 20, 25], min_samples_split': [2, 3, 4, 5, 6, 8, 10], min_samples_leaf': [1, 2, 3, 4, 5], max_features': ['sqrt', 'log2', 0.5, 0.7, None], bootstrap': [True, False], criterion': ['gini', 'entropy', 'log_loss'], class_weight': [None, 'balanced']	n_estimators: 200, min_samples_split: 10, min_samples_leaf: 3, max_features: sqrt, max_depth: 20, criterion: entropy, class_weight: balanced, bootstrap: False
XGBoost	**1**	0,634	n_estimators': randint(100, 400), max_depth': randint(3, 10), learning_rate': uniform(0.01, 0.2), subsample': uniform(0.7, 0.3), colsample_bytree': uniform(0.7, 0.3), min_child_weight': randint(1, 10), gamma': uniform(0, 0.3), reg_lambda': uniform(0.5, 2.0), reg_alpha': uniform(0, 1.0)	colsample_bytree: 0.726, gamma: 0.0587, learning_rate: 0.019, max_depth: 7, min_child_weight: 2,n_estimators: 152,reg_alpha: 0.586, reg_lambda: 2.430, subsample: 0.882
Regresión Logística	**1**	**0,592**	penalty': ['l2', 'l1', 'elasticnet', None], solver': ['lbfgs', 'saga','Liblinear'], C': loguniform(1e-3, 10), l1_ratio': uniform(0, 1), fit_intercept': [True, False], class_weight': ['balanced'],	'C': 0.09, 'class_weight': 'balanced', 'fit_intercept': True, 'l1_ratio':0.71, 'penalty': 'l1', 'solver': 'saga'
MLP Classifier	**0**	**0,541**	Parametros por defecto del modelo	hidden_layer_sizes=(100,), activation='relu', solver='adam', alpha=0.0001, batch_size='auto', learning_rate='constant', learning_rate_init=0.001, power_t=0.5, max_iter=200, shuffle=True, random_state=None, tol=1e-4, verbose=False, warm_start=False, momentum=0.9, nesterovs_momentum=True, early_stopping=False, validation_fraction=0.1, beta_1=0.9, beta_2=0.999, epsilon=1e-8, n_iter_no_change=10, max_fun=15000
SVM	**0**	**0,611**	Parametros por defecto del modelo	C=1.0, kernel='rbf', degree=3, gamma='scale', coef0=0.0,

				shrinking=True, probability=False, tol=1e-3, cache_size=200, class_weight=None, verbose=False, max_iter=-1, decision_function_shape='ovr', break_ties=False, random_state=None
TabTransformer	**1**	**0,522**	No aplica.	"dim": 32, "depth": 4, "heads": 8, "attn_dropout": 0.1, "ff_dropout": 0.1, "mlp_hidden_mults": (4, 2), "epochs": 10, "batch_size": 64, "learning_rate": 1e-3

Tabla 5: Mejores hiperparámetros para cada modelo -Comercio Internacional

El Random Forest basó su optimización en el manejo explícito del desbalance de clases (class_weight: balanced) y el uso del criterio de entropía. Finalmente, el bajo desempeño del TabTransformer (0.522) con una arquitectura de 4 capas y 8 cabezales sugiere que la complejidad estructural del modelo profundo no logró converger adecuadamente con la cantidad de datos disponibles, quedando por debajo de los algoritmos de ensambles y lineales.

La tabla 6 de rendimiento de los modelos resume los resultados obtenidos tras el proceso de ajuste y validación cruzada de cada algoritmo.

Modelo	Ronda	ROC AUC	Clase 1 Matricula			Clase 0 Admisión		
			Precision	Recall	F1-score	Precision	Recall	F1-score
DummyClassifier		0,515	0,683	0,703	0,693	0,348	0,327	0,337
RandomForest	0	0,588	0,699	0,921	0,795	0,529	0,184	0,273
	1	**0,621**	**0,705**	**0,852**	**0,771**	**0,464**	**0,265**	**0,338**
	2	0,593	0,717	0,852	0,778	0,500	0,306	0,380
	3	0,593	0,717	0,852	0,778	0,500	0,306	0,380
XGBoost	0	0,570	0,721	0,792	0,755	0,462	0,367	0,409
	1	0,634	**0,721**	**0,743**	**0,732**	**0,435**	**0,408**	**0,421**
	2	0,624	0,722	0,772	0,746	0,452	0,388	0,418
	3	0,624	0,722	0,772	0,746	0,452	0,388	0,418
Regresión Logística	0	0,573	0,696	0,951	0,803	0,583	0,143	0,230
	1	**0,592**	**0,759**	**0,624**	**0,685**	**0,433**	**0,592**	**0,500**
	2	0,585	0,768	0,624	0,689	0,441	0,612	0,513
	3	0,585	0,768	0,624	0,689	0,441	0,612	0,513
MLP Classifier	**0**	**0,541**	**0,694**	**0,762**	**0,726**	**0,385**	**0,306**	**0,341**
	1	0,510	0,662	0,505	0,573	0,315	0,469	0,377
	2	0,509	0,635	0,654	0,644	0,239	0,225	0,232
	3	0,537	0,709	0,723	0,716	0,404	0,388	0,396
SVM	**0**	**0,611**	**0,673**	**1,000**	**0,805**	**0,000**	**0,000**	**0,000**
	1	0,579	0,688	0,960	0,802	0,556	0,102	0,172
	2	0,579	0,692	0,713	0,702	0,370	0,347	0,358
	3	0,579	0,692	0,713	0,702	0,370	0,347	0,358
TabTransformer	0	0,509	0,703	0,634	0,667	0,373	0,449	0,407
	1	**0,522**	**0,689**	**0,614**	**0,649**	**0,350**	**0,429**	**0,385**
	2	0,510	0,688	0,654	0,670	0,352	0,388	0,369

Tabla 6: Rendimiento de los modelos -Comercio internacional

En general, se observa que los modelos como RandomForest y XGBoost, mostraron un comportamiento estable y un equilibrio adecuado entre precisión y sensibilidad, reflejando su capacidad para manejar relaciones no lineales del conjunto de datos, aunque XGBoost obtuvo el rendimiento más alto. La Regresión Logística presentó un desempeño moderado, con menor capacidad predictiva frente a modelos más complejos. El MLPClassifier y el SVM lograron mejoras parciales durante la optimización, aunque con variaciones en recall y AUC según los hiperparámetros empleados. Finalmente, el TabTransformer, obtuvo resultados competitivos en validación cruzada, aunque su rendimiento en el conjunto de prueba fue muy limitado, posiblemente debido al tamaño del conjunto de datos y a la complejidad del modelo.

Así las cosas, el modelo que mejor se adapta a los datos del programa de Comercio Internacional es el XGBoost (Ronda 1).

5.3 ANALISIS DE RESULTADOS PARA EL PROGRAMA DE COMUNICACIÓN SOCIAL

El conjunto de datos del programa de Comunicación Social comprende 359 registros, con 205 casos de “Matrícula” (57.1%) y 154 de “Admisión” (42.9%), reflejando un leve desbalance. En este contexto, y como se evidencia en la Tabla 7, las estrategias de búsqueda de hiperparámetros detalladas en el Anexo 3, aunque exhaustivas, revelaron que las configuraciones por defecto de librerías como Scikit-learn ofrecen un punto de partida altamente competitivo para este conjunto de datos específico.

Modelo	Ronda	ROC AUC	Grilla de parámetros utilizada	Mejores parámetros
RandomForest	**0**	**0,695**	Parámetros por defecto del modelo	n_estimators=100, criterion='gini', max_depth=None, min_samples_split=2, min_samples_leaf=1, min_weight_fraction_leaf=0.0, max_features='sqrt', max_leaf_nodes=None, min_impurity_decrease=0.0, bootstrap=True, oob_score=False, n_jobs=None, random_state=None, verbose=0, warm_start=False, class_weight=None, ccp_alpha=0.0, max_samples=None
XGBoost	**1**	**0,663**	n_estimators': [100, 200, 300, 400, 500, 600], learning_rate': [0.01, 0.05, 0.1, 0.2, 0.3], max_depth': [3, 4, 5, 6, 8, 10], min_child_weight': [1, 3, 5, 7], gamma': [0, 0.1, 0.2, 0.3, 0.5], subsample': [0.6, 0.7, 0.8, 0.9, 1.0], colsample_bytree': [0.6, 0.7, 0.8, 0.9, 1.0], reg_alpha': [0, 0.01, 0.1, 0.5, 1],	n_estimators=100, max_depth=6, learning_rate=0.3, subsample=1, colsample_bytree=1, gamma=0, reg_alpha=0, reg_lambda=1, min_child_weight=1, objective='binary:logistic', booster='gbtree', n_jobs=None, random_state=0, verbosity=1

			reg_lambda': [0.5, 1, 1.5, 2, 3], booster': ['gbtree', 'dart'], tree_method': ['hist']	
Regresión Logística	**0**	**0,672**	Parámetros por defecto del modelo	penalty='l2', dual=False, tol=1e-4, C=1.0, fit_intercept=True, intercept_scaling=1, class_weight=None, random_state=None, solver='lbfgs', max_iter=100, multi_class='auto', verbose=0, warm_start=False, n_jobs=None, l1_ratio=None
MLP Classifier	**0**	**0,679**	Parámetros por defecto del modelo	hidden_layer_sizes=(100,), activation='relu', solver='adam', alpha=0.0001, batch_size='auto', learning_rate='constant', learning_rate_init=0.001, power_t=0.5, max_iter=200, shuffle=True, random_state=None, tol=1e-4, verbose=False, warm_start=False, momentum=0.9, nesterovs_momentum=True, early_stopping=False, validation_fraction=0.1, beta_1=0.9, beta_2=0.999, epsilon=1e-8, n_iter_no_change=10, max_fun=15000
SVM	**0**	**0,696**	Parámetros por defecto del modelo	C=1.0, kernel='rbf', degree=3, gamma='scale', coef0=0.0, shrinking=True, probability=False, tol=1e-3, cache_size=200, class_weight=None, verbose=False, max_iter=-1, decision_function_shape='ovr', break_ties=False, random_state=None
TabTransformer	**1**	**0,576**	No aplica.	"dim": 32, "depth": 4, "heads": 8, "attn_dropout": 0.1, "ff_dropout": 0.1, "mlp_hidden_mults": (4, 2), "epochs": 10, “batch_size": 64, "learning_rate": 1e-3

Tabla 7: Mejores hiperparámetros para cada modelo - Comunicación social

A excepción del TabTransformer, cuya arquitectura profunda requiere ajuste manual, y el XGBoost, cuya arquitectura gradiente requiere calibración adicional, el resto de los modelos alcanzaron un desempeño óptimo sin necesidad de ajustes adicionales, lo cual es indicativo de la simplicidad estructural de los patrones presentes en los datos.

En la Tabla 8 sobre el rendimiento comparativo de los modelos, se observa un fenómeno particular, asociado posiblemente al tamaño reducido del conjunto de datos (359 registros): la optimización no logró superar a las configuraciones base en la mayoría de los algoritmos.

Los modelos complejos como XGBoost y las redes neuronales (MLP, TabTransformer) sufrieron una pérdida de rendimiento al intentar ajustar sus parámetros o mostraron inestabilidad (TabTransformer con AUC de 0.576), lo que sugiere que la regularización adicional o la profundidad de búsqueda limitaron su capacidad de aprendizaje en este escenario específico.

Modelo	Ronda	ROC AUC	Clase 1 Matricula			Clase 0 Admisión		
			Precision	Recall	F1-score	Precision	Recall	Precision
DummyClassifier		0,456	0,539	0,565	0,551	0,372	0,348	0,360
RandomForest	**0**	**0,695**	**0,716**	**0,774**	**0,744**	**0,659**	**0,587**	**0,621**
	1	0,666	0,671	0,790	0,726	0,629	0,478	0,543
	2	0,662	0,680	0,855	0,757	0,700	0,457	0,553
	3	0,650	0,662	0,790	0,721	0,618	0,457	0,525
XGBoost	**0**	**0,663**	**0,685**	**0,807**	**0,741**	**0,657**	**0,500**	**0,568**
	1	0,643	0,667	0,677	0,672	0,556	0,544	0,550
	2	0,643	0,667	0,677	0,672	0,556	0,544	0,550
	3	0,650	0,667	0,677	0,672	0,556	0,544	0,550
Regresión Logística	**0**	**0,672**	**0,608**	**0,774**	**0,681**	**0,517**	**0,326**	**0,400**
	1	0,667	0,647	0,710	0,677	0,550	0,478	0,512
	2	0,670	0,667	0,710	0,688	0,571	0,522	0,546
	3	0,667	0,647	0,710	0,677	0,550	0,478	0,512
MLP Classifier	**0**	**0,679**	**0,721**	**0,710**	**0,715**	**0,617**	**0,630**	**0,624**
	1	0,653	0,694	0,694	0,694	0,587	0,587	0,587
	2	0,601	0,635	0,645	0,640	0,511	0,500	0,506
	3	0,620	0,656	0,645	0,650	0,532	0,544	0,538
SVM	**0**	**0,696**	**0,627**	**0,839**	**0,717**	**0,600**	**0,326**	**0,423**
	1	0,683	0,682	0,726	0,703	0,595	0,544	0,568
	2	0,683	0,702	0,758	0,729	0,634	0,565	0,598
	3	0,683	0,687	0,742	0,713	0,610	0,544	0,575
TabTransformer	0	0,564	0,629	0,629	0,629	0,500	0,500	0,500
	1	**0,576**	**0,617**	**0,468**	**0,532**	**0,459**	**0,609**	**0,523**
	2	0,570	0,617	0,468	0,532	0,459	0,609	0,523

Tabla 8: Rendimiento de los modelos -Comunicación social

Por el contrario, el SVM y el Random Forest con parámetros por defecto lideraron la tabla con AUC cercanos a 0.70. Aunque el SVM obtuvo el puntaje más alto (0.696), su desempeño está fuertemente sesgado hacia la clase mayoritaria, presentando un Recall de 0.326 para la clase 0 (Admisión). En contraste, el Random Forest (Ronda 0), con un AUC de 0.695, ofrece un comportamiento mucho más equilibrado, elevando el Recall de la clase minoritaria a 0.587.

Para el programa de comunicación social, pese a que el SVM obtuvo el ROC AUC más elevado, RandomForest con parámetros por defecto presenta un comportamiento más equilibrado y mejores indicadores en las métricas de evaluación.

5.4 ANALISIS DE RESULTADOS PARA EL PROGRAMA DE CONTADURÍA PÚBLICA

En el caso del programa de Contaduría Pública, el conjunto de datos está conformado por 989 registros, distribuidos en 651 casos de “Matrícula” (65,8%) y 338 de “Admisión” (34,2%), lo que revela un moderado desbalance de clases a favor del estado de matrícula. La Tabla 9 detalla las configuraciones óptimas que resultaron del proceso de búsqueda de hiperparámetros.

Modelo	Ronda	ROC AUC	Grilla de parámetros utilizada	Mejores parámetros
RandomForest	**3**	**0,535**	'n_estimators': [150, 200, 250], 'max_depth': [3, 4, 5], 'min_samples_split': [2, 3], 'min_samples_leaf': [2, 3, 4], 'max_features': ['sqrt'], 'bootstrap': [True], 'class_weight': ['balanced']	bootstrap: True, class_weight: balanced, max_depth: 4, max_features: sqrt, min_samples_leaf: 4, min_samples_split: 2, n_estimators: 150
XGBoost	**3**	**0,524**	'booster': ['dart'], 'learning_rate': [0.1, 0.12], 'n_estimators': [100, 150], 'max_depth': [3, 4], 'min_child_weight': [1, 2], 'subsample': [0.9, 1.0], 'colsample_bytree': [0.9, 1.0], 'reg_lambda': [0.1, 1], 'reg_alpha': [0, 0.5],	booster: dart, colsample_bytree: 1.0, learning_rate: 0.1, max_depth: 3, min_child_weight: 1, n_estimators: 100, reg_alpha: 0, reg_lambda: 1 subsample: 0.9
Regresión Logística	**1**	**0,605**	'solver': ['lbfgs', 'saga'], 'C': np.logspace(-3, 2, 10), 'fit_intercept': [True, False], 'class_weight': [None, 'balanced'] 'penalty': ['l1', 'l2', 'elasticnet'], 'l1_ratio': np.linspace(0, 1, 5),	'solver': 'saga', 'penalty': 'l1', 'l1_ratio': 0, 'fit_intercept': False, 'class_weight': 'balanced', 'C': 2.15
MLP Classifier	**0**	**0,522**	'hidden_layer_sizes': [(32,), (64,), (32, 16), (64, 32), (128, 64)], 'activation': ['relu', 'tanh', 'logistic'], 'solver': ['adam', 'lbfgs'], 'alpha': uniform(1e-5, 1e-2), 'learning_rate_init': uniform(1e-4, 5e-3), 'batch_size': [16, 32, 64], 'learning_rate': ['constant', 'adaptive'], 'early_stopping': [True],	hidden_layer_sizes=(100,), activation='relu', solver='adam', alpha=0.0001, batch_size='auto', learning_rate='constant', learning_rate_init=0.001, power_t=0.5, max_iter=200, shuffle=True, random_state=None, tol=1e-4, verbose=False, warm_start=False, momentum=0.9, nesterovs_momentum=True, early_stopping=False, validation_fraction=0.1, beta_1=0.9, beta_2=0.999, epsilon=1e-8, n_iter_no_change=10, max_fun=15000
SVM	**1**	**0,617**	'kernel': ['rbf', 'poly', 'sigmoid', linear'], 'C': np.logspace(-2, 2, 10), 'gamma': ['scale', 'auto', 0.001, 0.01, 0.1, 1], 'degree': [2, 3], 'coef0': [0, 0.5, 1], 'shrinking': [True, False], 'tol': [1e-3, 1e-4], 'max_iter': [-1, 1000, 2000], 'class_weight': ['balanced', None],	'tol': 0.0001, 'shrinking': True, 'probability': True, 'max_iter': 2000, 'kernel': 'sigmoid', 'gamma': 0.01, 'degree': 3, 'coef0': 0.5, 'class_weight': 'balanced', 'C': 4.64
TabTransformer	**2**	**0,542**	No aplica	"dim": 64, "depth": 3, "heads": 4, "attn_dropout": 0.2, "ff_dropout": 0.2, "mlp_hidden_mults": (4, 2), "epochs": 25, "batch_size": 32, "learning_rate": 5e-4, "weight_decay": 1e-4,

Tabla 9: Mejores hiperparámetros para cada modelo - Contaduría Pública

En este conjunto de datos, se observa que los modelos que exhibieron el mejor rendimiento (SVM

y Regresión Logística) requirieron un ajuste que priorizó la regularización:

- El SVM (Ronda 1) alcanzó su pico de 0.617 al emplear el kernel sigmoide, junto con una alta penalización C (4.64) y un bajo gamma (0.01).
- La Regresión Logística (Ronda 1), con un AUC de 0.605, se optimizó con la penalización L1 y la desactivación del intercepto (fit_intercept: False), forzando al modelo a ser más lento.

Los modelos de mayor complejidad (Random Forest, XGBoost, MLP, y TabTransformer) se mantuvieron en niveles de rendimiento significativamente inferiores (AUC entre 0.52 y 0.54), a pesar de extensas rondas de ajuste. El anexo 4 muestra el proceso de ajuste de hiperparámetros realizado para cada uno de los modelos evaluados en el estudio.

Para la mayoría de los modelos, el ajuste de hiperparámetros resultó en mejoras modestas en el rendimiento medido por el ROC AUC, en comparación con los modelos con parámetros por defecto. Sin embargo, dado el tamaño relativamente pequeño del conjunto de datos, existe la posibilidad de que el espacio de búsqueda de hiperparámetros sea limitado o que el propio dataset no contenga patrones lo suficientemente fuertes para permitir mejoras sustanciales con el ajuste fino. En la tabla 10, los modelos muestran un desempeño general moderado, con valores de ROC AUC entre 0.38 y 0.62, lo cual indica una capacidad de discriminación limitada pero superior al azar en la mayoría de los casos.

Modelo	Ronda	ROC AUC	Clase 1 Matricula			Clase 0 Admisión		
			Precision	Recall	F1-score	Precision	Recall	Precision
DummyClassifier		**0,523**	**0,672**	**0,692**	**0,682**	**0,375**	**0,353**	**0,364**
RandomForest	0	0,511	0,657	0,854	0,742	0,345	0,147	0,206
	1	0,534	0,664	0,562	0,608	0,352	0,456	0,397
	2	0,533	0,642	0,592	0,616	0,321	0,368	0,342
	3	**0,535**	**0,655**	**0,600**	**0,627**	**0,342**	**0,397**	**0,367**
XGBoost	0	0,518	0,653	0,723	0,686	0,333	0,265	0,295
	1	0,502	0,631	0,592	0,611	0,303	0,338	0,319
	2	0,519	0,661	0,569	0,612	0,349	0,441	0,390
	3	**0,524**	**0,646**	**0,562**	**0,601**	**0,329**	**0,412**	**0,366**
Regresión Logística	0	0,600	0,680	0,931	0,786	0,550	0,162	0,250
	1	**0,605**	**0,742**	**0,615**	**0,674**	**0,444**	**0,588**	**0,506**
	2	0,602	0,737	0,608	0,667	0,444	0,588	0,506
	3	0,603	0,737	0,608	0,667	0,444	0,588	0,506
MLP Classifier	**0**	**0,522**	**0,654**	**0,685**	**0,669**	**0,333**	**0,294**	**0,313**
	1	0,513	0,655	0,554	0,600	0,338	0,426	0,377
	2	0,510	0,659	0,623	0,640	0,368	0,397	0,382
	3	0,510	0,659	0,623	0,640	0,368	0,397	0,382
SVM	0	0,576	0,674	0,985	0,800	0,667	0,059	0,108
	1	**0,617**	**0,763**	**0,608**	**0,676**	**0,457**	**0,662**	**0,537**
	2	0,382	0,657	1,000	0,793	0,000	0,000	0,000
	3	0,415	0,657	1,000	0,793	0,000	0,000	0,000
TabTransformer	0	0,525	0,641	0,508	0,566	0,333	0,456	0,385
	1	0,493	0,653	0,492	0,561	0,338	0,500	0,404
	2	**0,542**	**0,670**	**0,531**	**0,592**	**0,361**	**0,500**	**0,420**

Tabla 10: Rendimiento de los modelos -Contaduría Pública

El SVM (Ronda 1) es el modelo que mejor se ajusta al programa de Contaduría Pública. Sin embargo, su capacidad de discriminación es débil, lo que puede deberse al tamaño del conjunto de datos.

5.5 ANALISIS DE RESULTADOS PARA EL PROGRAMA DE DERECHO

El conjunto de datos del programa de Derecho contiene 1.530 registros, de los cuales 865 corresponden al estado “Matrícula” (56,54 %) y 665 a “Admisión” (43,46 %), lo que evidencia un desbalance de clases mínimo a favor del estado de matrícula.
La Tabla 11 resume el mejor conjunto de hiperparámetros identificado, revelando que el RandomForest (Ronda 1) alcanzó su máximo AUC (0.643) utilizando un alto número de estimadores (n_estimators: 250) y controlando el desbalance de clases (class_weight: balanced), mientras que el XGBoost (Ronda 3) logró 0.621 limitando la profundidad del árbol (max_depth: 3) y utilizando una tasa de aprendizaje muy baja (0.021).

Modelo	Ronda	ROC AUC	Grilla de parámetros utilizada	Mejores parámetros
RandomForest	**1**	**0,643**	n_estimators': randint(100, 500), max_depth': [None, 5, 10, 15, 20 min_samples_split': randint(2, 10 min_samples_leaf': randint(1, 5), max_features': ['sqrt', 'log2'], bootstrap': [True, False criterion': ['gini', 'entropy'], class_weight': [None, 'balanced']	bootstrap: True, class_weight: balanced, criterion: entropy, max_depth: None, max_features: sqrt, min_samples_leaf: 4, min_samples_split: 2 n_estimators: 250
XGBoost	**3**	**0,621**	n_estimators': [200, 220, 240], learning_rate': [0.018, 0.021, 0.024], max_depth': [3, 4, 5], min_child_weight': [4, 5, 6], gamma': [0.15, 0.2, 0.25], subsample': [0.8, 0.85], colsample_bytree': [0.6, 0.65, 0.7],	colsample_bytree: 0.6, gamma: 0.2, learning_rate: 0.021, max_depth: 3, min_child_weight: 6, n_estimators: 240, subsample: 0.85
Regresión Logística	**0**	**0,567**	Parámetros por defecto del modelo	penalty='l2', dual=False, tol=1e-4, C=1.0, fit_intercept=True, intercept_scaling=1, class_weight=None, random_state=None, solver='lbfgs', max_iter=100, multi_class='auto', verbose=0, warm_start=False, n_jobs=None, l1_ratio=None
MLP Classifier	**0**	**0,585**	Parámetros por defecto del modelo	hidden_layer_sizes=(100,), activation='relu', solver='adam', alpha=0.0001, batch_size='auto', learning_rate='constant', learning_rate_init=0.001, power_t=0.5, max_iter=200, shuffle=True, random_state=None, tol=1e-4, verbose=False, warm_start=False, momentum=0.9,

				nesterovs_momentum=True, early_stopping=False, validation_fraction=0.1, beta_1=0.9, beta_2=0.999, epsilon=1e-8, n_iter_no_change=10, max_fun=15000
SVM	**0**	**0,606**	Parámetros por defecto del modelo	C=1.0, kernel='rbf', degree=3, gamma='scale', coef0=0.0, shrinking=True, probability=False, tol=1e-3, cache_size=200, class_weight=None, verbose=False, max_iter=-1, decision_function_shape='ovr', break_ties=False, random_state=None
TabTransformer	**2**	**0,595**	No aplica	"dim": 128, "depth": 2, "heads": 4, "attn_dropout": 0.1, "ff_dropout": 0.2, "mlp_hidden_mults": (4, 2), "batch_size": 32, "learning_rate": 3e-4, "weight_decay": 0.01

Tabla 11: Mejores hiperparámetros para cada modelo -Derecho

Por otro lado, la Regresión Logística, el MLPClassifier y el SVM obtuvieron su mejor rendimiento con los parámetros por defecto (Ronda 0). Esto sugiere que la búsqueda de hiperparámetros para estos modelos introdujo inestabilidad o sobreajuste en rondas posteriores, validando la robustez de sus configuraciones base. El TabTransformer (Ronda 2), pese a su arquitectura específica con una dimensionalidad de 128 y profundidad de 2, mostró un desempeño limitado (0.595). El anexo 5 presenta los espacios de búsqueda de hiperparámetros utilizados para optimizar cada modelo durante las diferentes rondas de ajuste.

La tabla 12 presenta las diferencias de desempeño entre las configuraciones por defecto (ronda 0) y las distintas rondas de ajuste de hiperparámetros realizadas para cada modelo. En particular, el modelo RandomForest (Ronda 1) logró el mayor valor de ROC AUC (0.6427), junto con un equilibrio favorable entre precisión y recall para la clase positiva (Precision = 0.6610; Recall = 0.6763; F1 = 0.6686), superando tanto a su configuración por defecto como a las rondas posteriores de ajuste. Esto sugiere que las modificaciones iniciales en los hiperparámetros lograron una mejor capacidad de generalización antes de que los ajustes adicionales produjeran sobreajuste.
El modelo XGBoost mostró un rendimiento estable con valores de ROC AUC cercanos a 0.62, destacando el XGBoost (Ronda 2) con un mejor balance entre clases. Los modelos Regresión Logística y SVM presentaron comportamientos más variables: aunque alcanzaron recalls elevados en algunas configuraciones (por ejemplo, SVM por defecto con 0.7746), esto se compensó con una menor precisión y una caída del AUC, lo que indica sobreajuste hacia la clase mayoritaria.

Los modelos MLPClassifier y TabTransformer tuvieron un rendimiento limitado, con ROC AUC entre 0.52 y 0.59. A pesar de su menor desempeño, mostraron estabilidad en las métricas y potencial de mejora con un ajuste más fino de hiperparámetros. Finalmente, el DummyClassifier, obtuvo un ROC AUC de 0.5155, confirmando que todos los modelos entrenados superaron el rendimiento esperado por azar.

Modelo	Ronda	ROC AUC	Clase 1 Matricula			Clase 0 Admisión		
			Precision	Recall	F1-score	Precision	Recall	Precision
DummyClassifier		0,516	0,579	0,572	0,576	0,452	0,459	0,455
RandomForest	0	0,632	0,633	0,717	0,672	0,555	0,459	0,502
	1	**0,643**	**0,661**	**0,676**	**0,669**	**0,566**	**0,549**	**0,557**
	2	0,630	0,652	0,584	0,616	0,523	0,594	0,556
	3	0,630	0,652	0,584	0,616	0,523	0,594	0,556
XGBoost	0	0,606	0,645	0,694	0,669	0,558	0,504	0,530
	1	0,620	0,634	0,561	0,595	0,503	0,579	0,539
	2	0,616	0,645	0,578	0,610	0,517	0,587	0,549
	3	**0,621**	**0,642**	**0,561**	**0,599**	**0,510**	**0,594**	**0,549**
Regresión Logística	**0**	**0,567**	**0,578**	**0,688**	**0,628**	**0,460**	**0,346**	**0,395**
	1	0,564	0,560	0,786	0,654	0,413	0,196	0,265
	2	0,563	0,565	0,832	0,673	0,431	0,165	0,239
	3	0,564	0,642	0,549	0,592	0,506	0,602	0,550
MLP Classifier	**0**	**0,585**	**0,626**	**0,619**	**0,622**	**0,511**	**0,519**	**0,515**
	1	0,560	0,604	0,538	0,569	0,474	0,541	0,505
	2	0,542	0,596	0,486	0,535	0,461	0,571	0,510
	3	0,520	0,579	0,572	0,576	0,452	0,459	0,455
SVM	**0**	**0,606**	**0,596**	**0,775**	**0,673**	**0,519**	**0,316**	**0,393**
	1	0,576	0,625	0,549	0,585	0,494	0,571	0,530
	2	0,576	0,638	0,549	0,590	0,545	0,594	0,545
	3	0,576	0,638	0,549	0,590	0,503	0,594	0,545
TabTransformer	0	0,580	0,606	0,561	0,583	0,480	0,526	0,502
	1	0,578	0,609	0,613	0,611	0,492	0,489	0,491
	2	**0,595**	**0,615**	**0,619**	**0,617**	**0,500**	**0,496**	**0,498**

Tabla 12: Rendimiento de los modelos -Derecho

Así, para el conjunto de datos del programa de derecho, el modelo RandomForest (Ronda 1) es el seleccionado para la elaboración del prototipo.

5.6 ANALISIS DE RESULTADOS PARA EL PROGRAMA DE PSICOLOGÍA

El conjunto de datos del programa de psicología contiene 1,224 registros (979 de entrenamiento y 245 de prueba) y 13 variables predictoras, además de la variable objetivo. Presenta una distribución prácticamente equilibrada entre ambas clases (614 matricula vs. 610 admisión), con una diferencia mínima de 0.32 puntos porcentuales (50.16 % vs. 49.84 %). La Tabla 13 detalla las configuraciones óptimas que resultaron del proceso de ajuste, poniendo en evidencia la superioridad de los modelos lineales para este conjunto de datos.

El SVM (Ronda 1), con un AUC de 0.608, también se apoyó en una solución lineal mediante el kernel: linear. La exclusión explícita de class_weight en el Random Forest también es notable, ya que el conjunto de datos es prácticamente equilibrado. En el anexo 6 se presentan las grillas de hiperparámetros empleadas durante las diferentes rondas de ajuste de los modelos.

Modelo	Ronda	ROC AUC	Grilla de parámetros utilizada	Mejores parámetros
RandomForest	**1**	**0,563**	n_estimators': np.arange(100, 600), max_depth': [None, 5, 10, 15, 20, 25, 30], min_samples_split': [2, 5, 10, 15, 20], min_samples_leaf': [1, 2, 4, 6, 8], max_features': ['sqrt', 'log2', None], bootstrap': [True, False], class_weight': [None, 'balanced', 'balanced_subsample'], criterion': ['gini', 'entropy', 'log_loss']	n_estimators: 113, min_samples_split: 5, min_samples_leaf: 6, max_features: log2, max_depth: 20, criterion: entropy, class_weight: None, bootstrap: False
XGBoost	**1**	**0,608**	n_estimators': np.arange(50, 300, 25), max_depth': np.arange(2, 10, 1), learning_rate': np.linspace(0.01, 0.3, 10), subsample': np.linspace(0.6, 1.0, 5), colsample_bytree': np.linspace(0.6, 1.0, 5), gamma': np.linspace(0, 5, 6), min_child_weight': np.arange(1, 8,1), reg_lambda': np.linspace(0.1, 2, 10), reg_alpha': np.linspace(0, 1, 6), booster': ['gbtree', 'dart'], tree_method': ['hist']	tree_method: hist subsample: 0.8, reg_lambda: 0.73, reg_alpha: 1.0, n_estimators: 100, min_child_weight: 6, max_depth: 9, learning_rate: 0.01 gamma: 3.0, colsample_bytree: 0.7, booster: dart
Regresión Logística	**2**	**0,623**	solver': ['saga'], penalty': ['elasticnet'], C': [0.05, 0.1, 0.15, 0.18, 0.2, 0.25, 0.3], l1_ratio': [0.2, 0.3, 0.38, 0.5, 0.6], class_weight':['balanced']	C': 0.05, 'class_weight': 'balanced', 'l1_ratio': 0.5, 'penalty': 'elasticnet', 'solver': 'saga'
MLP Classifier	**2**	**0,591**	hidden_layer_sizes': [(40, 40), (50, 50), (60, 60)], activation': ['relu', 'tanh'], solver': ['adam'], alpha': [1e-4, 1e-3, 5e-3], learning_rate': ['constant', 'adaptive'], learning_rate_init': [0.001, 0.003], batch_size': [32], early_stopping': [True],max_iter': [2000]	'activation': 'relu', 'alpha': 0.0001, 'batch_size': 32, 'early_stopping': True, 'hidden_layer_sizes': (50, 50), 'learning_rate': 'constant', 'learning_rate_init': 0.003, 'max_iter': 2000, 'solver': 'adam'
SVM	**1**	**0,608**	C': np.logspace(-3, 2, 20), kernel': ['linear', 'poly', 'rbf', 'sigmoid'], gamma': ['scale', 'auto'] + list(np.logspace(-3, 1, 6)), degree': [2, 3, 4], coef0': [0.0, 0.1, 0.5, 1.0], class_weight': ['balanced'],	'kernel': 'linear', 'gamma': 0.25, 'degree': 4, 'coef0': 0.5, 'class_weight': 'balanced', 'C': 16.23
TabTransformer	**2**	**0,581**	No aplica.	"dim": 32, "depth": 3, "heads": 4, "attn_dropout": 0.1, "ff_dropout": 0.1, "weight_decay": 1e-5, "epochs": 50, "batch_size": 64, "learning_rate": 3e-4, "mlp_hidden_mults": (2,),

Tabla 13: Mejores hiperparámetros para cada modelo -Psicología

Los resultados en la tabla 14 muestran un comportamiento similar entre la mayoría de los modelos, con valores de ROC AUC entre 0.55 y 0.61, lo que indica una capacidad moderada de discriminación entre las clases. Los modelos Regresión Logística (rondas 2 y 3) y el SVM (rondas 2 y 3) alcanzaron los mejores desempeños, mostrando un equilibrio adecuado entre sensibilidad y especificidad.

Modelo	Ronda	ROC AUC	Clase 1 Matricula			Clase 0 Admisión		
			Precision	Recall	F1-score	Precision	Recall	Precision
DummyClassifier		0,543	0,545	0,545	0,545	0,541	0,541	0,541
RandomForest	0	0,534	0,521	0,512	0,516	0,516	0,525	0,520
	1	**0,563**	**0,523**	**0,561**	**0,541**	**0,522**	**0,484**	**0,502**
	2	0,562	0,531	0,561	0,545	0,530	0,500	0,515
	3	0,559	0,527	0,553	0,540	0,526	0,500	0,513
XGBoost	0	0,557	0,560	0,528	0,544	0,550	0,582	0,566
	1	**0,608**	**0,521**	**0,919**	**0,665**	**0,643**	**0,148**	**0,240**
	2	0,603	0,563	0,577	0,570	0,563	0,549	0,556
	3	0,603	0,568	0,577	0,573	0,567	0,557	0,562
Regresión Logística	0	0,598	0,556	0,561	0,559	0,554	0,549	0,551
	1	0,610	0,559	0,577	0,568	0,559	0,541	0,550
	2	**0,623**	**0,533**	**0,659**	**0,589**	**0,548**	**0,480**	**0,474**
	3	0,621	0,544	0,650	0,593	0,561	0,451	0,500
MLP Classifier	0	0,549	0,508	0,496	0,502	0,504	0,516	0,510
	1	0,580	0,549	0,407	0,467	0,526	0,664	0,587
	2	**0,591**	**0,561**	**0,561**	**0,561**	**0,557**	**0,557**	**0,557**
	3	0,581	0,553	0,512	0,532	0,542	0,582	0,561
SVM	0	0,571	0,528	0,528	0,528	0,525	0,525	0,525
	1	**0,608**	**0,566**	**0,593**	**0,579**	**0,569**	**0,541**	**0,555**
	2	0,608	0,566	0,593	0,579	0,569	0,541	0,555
	3	0,608	0,566	0,593	0,579	0,569	0,541	0,555
TabTransformer	0	0,576	0,568	0,577	0,573	0,567	0,557	0,562
	1	0,556	0,518	0,480	0,498	0,511	0,549	0,530
	2	**0,581**	**0,533**	**0,528**	**0,531**	**0,528**	**0,533**	**0,531**

Tabla 14: Rendimiento de los modelos -Psicología

Para el conjunto de datos del programa de psicología, el modelo Regresión logística (ronda 2) es el seleccionado para la elaboración del prototipo.

5.7 RESULTADOS FINALES POR PROGRAMA Y MODELO

La Tabla 15 resume el desempeño de los mejores modelos obtenidos en cada programa académico. El ROC AUC promedio entre programas se sitúa entre 0.62 y 0.70, lo que indica una capacidad discriminante moderada, pero suficiente para clasificar correctamente una proporción significativa de aspirantes entre los estados de matrícula y admisión.

El TabTransformer, RandomForest y XGBoost destacan como los enfoques con mejor equilibrio entre clases, alcanzando valores de F1-score superiores a 0.70 en los programas de Administración de Empresas, Comunicación Social y Comercio Internacional, respectivamente.
El SVM mostró un rendimiento competitivo en Contaduría Pública, logrando la mayor precisión para la clase de matrícula (0.76), aunque con un leve descenso en la recuperación de los casos de admisión.
Por otro lado, los programas con conjuntos de datos más amplios —como Derecho y Psicología— mantuvieron resultados consistentes (ROC AUC ≈ 0.63), aunque con menor diferencia entre clases, lo cual refleja una mejor generalización, pero también cierta dificultad para capturar

patrones diferenciales más específicos de los aspirantes.

Programa	Modelo	ROC AUC	Clase 1			Clase 0		
			Precision	Recall	F1-score	Precision	Recall	F1-score
Administración de empresas	TabTransformer Ronda 2	**0.695**	0.745	0.527	0.617	0.534	0.750	0.624
Comercio internacional	XGBoost R1	0.633	0.721	0.742	0.731	0.434	0.408	0.421
Comunicación social	RandomForest por Defecto	**0.695**	0.716	0.774	0.744	0.658	0.587	0.620
Contaduría Pública	SVM Ronda 1	0.616	0.763	0.607	0.676	0.456	0.661	0.536
Derecho	RandomForest Ronda 1	0.642	0.661	0.676	0.668	0.565	0.548	0.557
Psicología	Regresión Logística Ronda 2	0.623	0.532	0.658	0.589	0.548	0.418	0.474

Tabla 15: Mejores modelos en cada programa académico

En términos globales, los resultados confirman que no existe un modelo superior para todos los programas, sino que la efectividad depende del volumen de datos, la homogeneidad de los aspirantes y la relevancia de las variables incluidas.

6 ANALISIS DE RESULTADOS

Este capítulo presenta una síntesis comparativa del desempeño de los modelos de clasificación aplicados a cada programa académico, destacando las tendencias comunes y las diferencias observadas según la naturaleza de los datos.

La Tabla 16 presenta un resumen comparativo del mejor rendimiento obtenido por los modelos de clasificación en los diferentes programas académicos, expresado mediante el valor de ROC AUC en el conjunto de prueba. En términos generales, los enfoques no lineales, en particular SVM, RandomForest y XGBoost, alcanzaron los mejores resultados globales, con un promedio de ROC AUC entre 0.60 y 0.62.

El modelo SVM fue el más destacado, logrando el mayor promedio global (0.62) y los mejores resultados en programas como Comunicación Social (0.69) y Comercio Internacional (0.61).

La Regresión Logística, a pesar de su simplicidad, mostró un desempeño notablemente competitivo, alcanzando el mejor resultado en Administración de Empresas (0.68) y Psicología (0.62), lo que sugiere que las relaciones entre las variables predictoras y la variable objetivo conservan una estructura mayormente lineal.

Los modelos de ensamble, como RandomForest y XGBoost, mantuvieron un rendimiento equilibrado en todos los programas, con valores de ROC AUC en torno a 0.60, confirmando su capacidad para capturar interacciones complejas y manejar variables mixtas.

Modelo	Derecho	Comunicación Social	Contaduría Pública	Comercio Internacional	Administración de Empresas	Psicología	Promedio Global
DummyClassifier	0.52	0.46	0.52	0.51	0.48	0.54	0.50
Regresión Logística	0.57	0.67*	**0.60**	0.59	0.68*	**0.62**	0.60
RandomForest	**0.64**	**0.69***	0.53	0.62	0.67	0.56	0.60
XGBoost	0.62	0.66*	0.52	**0.63**	0.64	0.60	0.60
SVM	0.58	0.69*	0.62	0.61*	0.69	0.61	**0.62**
MLPClassifier	0.56	0.68*	0.52*	0.54*	0.67	0.58	0.60
TabTransformer	0.59	0.58	0.54	0.52	**0.69**	0.58	0.58

Tabla 16: Mejor rendimiento de los modelos en cada programa

Desde una perspectiva interpretativa, los modelos más complejos no siempre superaron a los más simples. Aunque TabTransformer y MLPClassifier ofrecen ventajas teóricas en la representación de relaciones no lineales y codificación de variables categóricas, sus beneficios se reducen cuando el tamaño de muestra o la calidad de los datos es limitada. Este comportamiento fue particularmente evidente en programas con menor volumen de registros, como Comunicación Social (359), donde la escasez de datos pudo haber restringido la capacidad de generalización de las arquitecturas profundas. En cambio, los modelos de ensamble, como RandomForest y

XGBoost, mostraron un equilibrio superior entre precisión, interpretabilidad y estabilidad, resultando más adecuados para contextos institucionales con bases de datos medianas o moderadamente desbalanceadas.

En cuanto a las variables predictoras más influyentes, los modelos interpretables (RandomForest, XGBoost y TabNet) mostraron de forma consistente que los puntajes académicos (especialmente en *Lectura Crítica* y *Matemáticas*), junto con la Edad de inscripción, el Estrato socioeconómico, y la Posible forma de pago son los factores más determinantes para el estado de matrícula. Estas variables son coherentes con los criterios institucionales de admisión, reflejando la relevancia tanto del desempeño académico como del contexto económico del aspirante.

El análisis comparativo evidencia diferencias claras entre programas en cuanto a la capacidad predictiva de los modelos. A pesar de contar con el conjunto de datos más pequeño, Comunicación Social (359 registros) alcanzó los mejores valores de ROC AUC promedio (≈ 0.67), especialmente con los modelos SVM, Random Forest y MLPClassifier, lo que sugiere que las variables disponibles permiten discriminar adecuadamente entre los aspirantes admitidos y matriculados, aunque no necesariamente refleja una generalización superior, sino una posible sensibilidad a la composición de la muestra o al menor nivel de ruido en los datos. En contraste, programas con mayor volumen de datos, como Derecho (1 530) o Psicología (1 224), presentaron rendimientos más moderados, lo que podría deberse a una mayor heterogeneidad en los perfiles de los aspirantes o a una menor relación entre las variables y la decisión final de matrícula.

Programas de tamaño intermedio, como Contaduría Pública y Administración de Empresas, mostraron comportamientos más equilibrados, con valores de ROC AUC ≈ 0.60–0.68, en los que los modelos lineales y de ensamble resultaron más consistentes. Estos hallazgos sugieren que el tamaño del dataset no garantiza un mejor desempeño, y que la relevancia y coherencia de las variables explicativas desempeña un papel más determinante que el volumen de datos en la calidad de la predicción.

En síntesis, los resultados evidencian que la elección del modelo debe responder al contexto y a la naturaleza de los datos disponibles, lo cual orienta las conclusiones y recomendaciones expuestas más adelante.

7 DESARROLLO DEL PROTOTIPO

Este capítulo describe el desarrollo del prototipo funcional que integra los modelos predictivos construidos durante el proyecto en una herramienta interactiva orientada a apoyar los procesos institucionales de admisión. Se presenta la arquitectura general del sistema, las tecnologías empleadas y el flujo de interacción con el usuario, destacando cómo el prototipo permite ingresar información de nuevos aspirantes y obtener predicciones consistentes con los modelos entrenados.

Se desarrolló un prototipo interactivo con el propósito de aplicar los resultados del modelado predictivo en una herramienta práctica para apoyar la gestión de admisiones. El sistema fue implementado en Python utilizando Streamlit, una herramienta que permite construir aplicaciones web basadas en Python de manera sencilla y dinámica, con una interfaz web sencilla e intuitiva en la que el usuario puede seleccionar el programa académico, ingresar los datos del aspirante y obtener una predicción del estado de matrícula (Fig. 20).

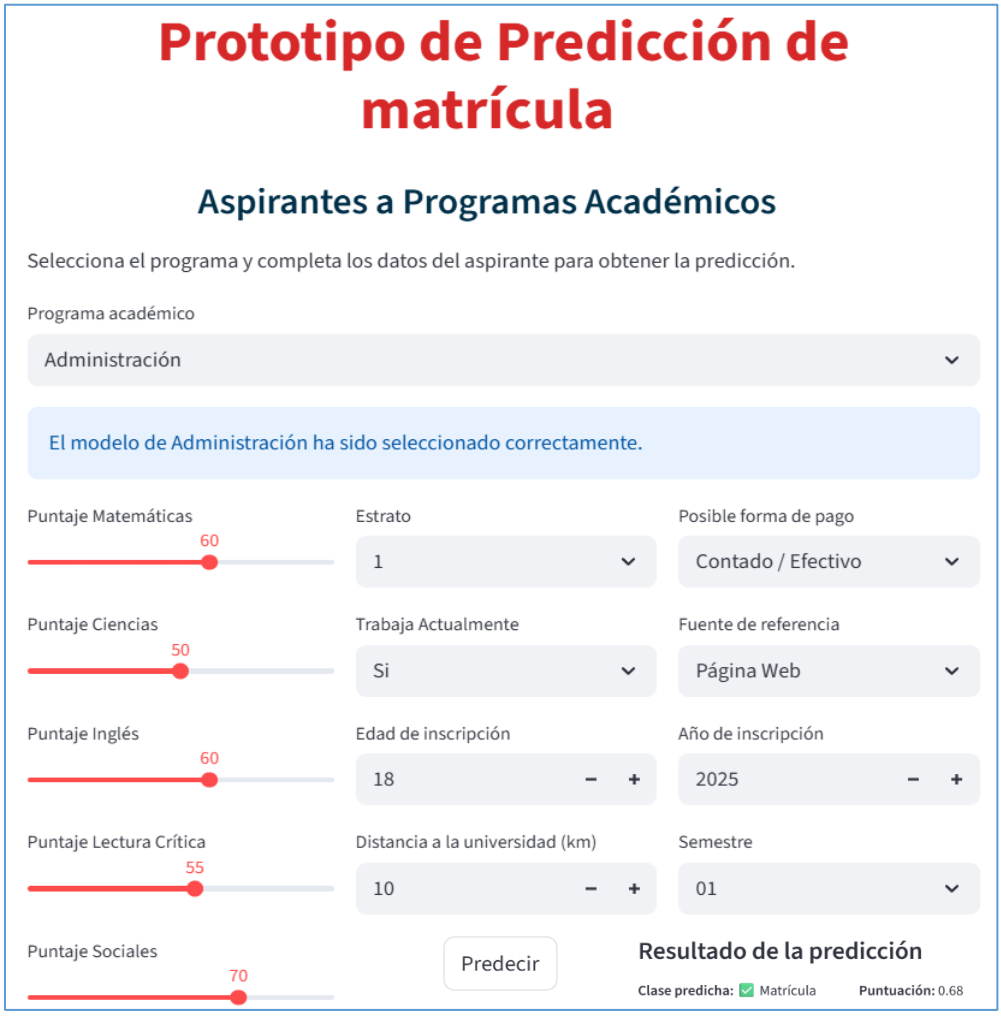

Fig. 20. Interfaz web del prototipo en Streamlit

El prototipo integra los modelos específicos para cada uno de los programas académicos, conservando únicamente la versión que alcanzó los mejores hiperparámetros según métricas como ROC AUC, precisión y F1-score. Los modelos se almacenaron en formato joblib y se cargan dinámicamente de acuerdo con el programa seleccionado por el usuario.

El prototipo fue integrado y desplegado utilizando GitHub como repositorio central y Streamlit como plataforma de ejecución de la aplicación. El código fuente, junto con los modelos entrenados y los archivos de configuración, se almacenó en un repositorio de GitHub, lo que facilitó la reproducibilidad del proyecto y simplificó el trabajo iterativo sobre la aplicación. A partir de este repositorio, Streamlit ejecuta automáticamente la aplicación, leyendo las dependencias definidas y cargando los modelos necesarios para la inferencia, lo que garantiza que cada actualización realizada en el código se refleje de forma inmediata en el prototipo.

Está desarrollado principalmente en Python, lenguaje en el que se implementaron tanto los modelos de aprendizaje automático como la lógica de la aplicación web mediante Streamlit. De forma complementaria, se utiliza HTML y CSS de manera implícita para la presentación visual de la interfaz, así como archivos de configuración en texto plano (como requirements.txt) para la gestión del entorno y dependencias, conformando una solución web interactiva, reproducible y orientada al despliegue de modelos predictivos.

En cuanto a la captura de información, la interfaz se organizó en dos columnas para mejorar la experiencia del usuario. En la columna izquierda se incluyen controles tipo *slider* para el ingreso de puntajes académicos, mientras que en la columna derecha se disponen campos de selección y entrada numérica asociados a variables sociodemográficas y de contexto.

Además, el prototipo aplica las mismas transformaciones y agrupaciones utilizadas durante el preprocesamiento de los datos, incluyendo codificación de variables categóricas, normalización de variables numéricas y consolidación de categorías como “Posible forma de pago” o “Fuente de referencia”. De esta forma, los valores ingresados por el usuario se procesan de manera consistente y compatible con el modelo correspondiente. Adicionalmente, se implementaron mecanismos de validación y manejo de excepciones para garantizar la estabilidad de la aplicación ante errores de carga de modelos o inconsistencias en los datos

Una vez se ingresan los datos del aspirante, el sistema procesa la información y genera como resultado la clase predicha (“Matrícula” o “Admisión”) junto con la puntuación de confianza (Score) asociada a la predicción. Esta puntuación refleja el nivel de confianza del modelo al clasificar al aspirante dentro de una de las dos categorías, donde valores altos indican una mayor afinidad con los perfiles de estudiantes que efectivamente se matricularon.

Este prototipo constituye una herramienta de apoyo para los procesos institucionales de análisis y toma de decisiones, al facilitar la identificación temprana de aspirantes con mayor afinidad hacia la matrícula, así como el fortalecimiento de estrategias de seguimiento y orientación.

8 CONCLUSIONES Y TRABAJOS FUTUROS

Este capítulo presenta las conclusiones derivadas del proceso de preparación de datos, modelado predictivo y evaluación realizado a lo largo del proyecto, resaltando los principales hallazgos y aportes obtenidos. Asimismo, se plantean líneas de trabajo futuro orientadas a mejorar la calidad de los datos, fortalecer el desempeño de los modelos y ampliar el alcance de la solución propuesta.

8.1 CONCLUSIONES

El desarrollo de este estudio permitió cumplir los objetivos planteados, logrando preparar los datos, entrenar múltiples modelos de clasificación, evaluarlos comparativamente y desarrollar un prototipo funcional para predecir la matrícula de los aspirantes. A través de este proceso se comprobó que la elección del modelo, la calidad de los datos y las estrategias de balanceo influyen directamente en la capacidad predictiva y de generalización de los algoritmos aplicados.

Los resultados obtenidos indican que los modelos de clasificación entrenados pueden apoyar la toma de decisiones institucionales en los procesos de admisión y matrícula, ofreciendo una herramienta predictiva complementaria para identificar aspirantes con mayor probabilidad de matricularse. No obstante, se observó que la generalización entre programas es limitada, lo cual indica que cada conjunto de datos requiere un ajuste y calibración propios. Esto sugiere que, aunque los modelos pueden compartir estructuras y metodologías, su entrenamiento debe ser específico para cada programa académico.

Desde el punto de vista metodológico, la implementación de estrategias de preprocesamiento y balanceo de clases, como SMOTENC y el uso de class weight, fue fundamental para mejorar la representación de las clases minoritarias y garantizar una evaluación justa del modelo. Asimismo, la aplicación de validación cruzada estratificada permitió obtener estimaciones más estables y reducir el riesgo de sobreajuste, asegurando la reproducibilidad de los resultados.

El trabajo también expone que la preparación cuidadosa de los datos, la selección informada de modelos y la evaluación comparativa sistemática son factores clave para construir soluciones predictivas confiables y aplicables en contextos reales. Además, se evidenció la relevancia de contar con un conocimiento profundo del dominio de los datos, pues comprender el contexto en el que se originan las variables es esencial para seleccionar atributos relevantes, interpretar los resultados correctamente y evitar conclusiones erróneas.

Un hallazgo relevante del estudio es que la capacidad predictiva de los modelos estuvo fuertemente condicionada por la naturaleza de las variables utilizadas en la modelación. Si bien las variables disponibles permitieron construir modelos funcionales y comparables, su alcance resultó moderado para capturar de manera integral la intención de matrícula de los aspirantes. La ausencia de indicadores asociados al nivel de interés, al comportamiento previo y a la

interacción del aspirante con la institución limita el desempeño de los modelos, incluso al emplear algoritmos avanzados. Este hecho explica el estancamiento del rendimiento predictivo observado y evidencia que la mejora del sistema no depende únicamente de la complejidad del modelo, sino, en mayor medida, de la calidad, pertinencia y riqueza informativa de las variables incorporadas.

El prototipo desarrollado en Streamlit representa la aplicabilidad práctica de los modelos, permitiendo que los resultados de predicción se integren en una herramienta interactiva que puede apoyar la toma de decisiones estratégicas en los procesos de admisión. Su uso potencial radica en ofrecer una visión temprana de los aspirantes con mayor probabilidad de matrícula, fortaleciendo la planeación institucional y las estrategias de seguimiento. Asimismo, el prototipo facilita la incorporación del análisis predictivo en los flujos operativos existentes, permitiendo decisiones informadas como la focalización de campañas, la asignación eficiente de recursos y el acompañamiento oportuno de los aspirantes, contribuyendo a una gestión de la matrícula basada en evidencia. En términos de impacto institucional, esta herramienta tiene el potencial de contribuir a la mejora de indicadores clave como la tasa de conversión de admitidos a matriculados, la eficiencia de las campañas de mercadeo y la priorización del seguimiento a aspirantes con mayor probabilidad de ingreso.

Finalmente, mejorar el rendimiento de los modelos requiere ampliar la base de datos y enriquecer las variables predictoras con indicadores relacionados con la intención de matrícula. Se recomienda incorporar variables que reflejen el comportamiento reciente del aspirante, como la participación en eventos institucionales, el número de interacciones con la página web o de comunicación con las áreas de mercadeo o admisiones, entre otras.

Esta implementación puede ser el punto de partida para un desarrollo futuro que incluya las variables sobre la intención del aspirante indicadas.

8.2 TRABAJOS FUTUROS

Para continuar y fortalecer el desarrollo realizado, se proponen las siguientes líneas de trabajo:

- Ampliar la base de datos incorporando nuevos periodos académicos, programas y variables adicionales (por ejemplo, factores motivacionales o de interacción de los aspirantes con la universidad), con el fin de mejorar la capacidad de generalización de los modelos.

- Fortalecer los procesos de recolección y registro de información académica y administrativa, asegurando la captura consistente y oportuna de los datos de los aspirantes desde las etapas iniciales del proceso de admisión.

- Integrar herramientas de interpretabilidad de modelos (por ejemplo, SHAP o LIME) para identificar y visualizar el peso de cada variable en la decisión de matrícula.

- Desplegar el prototipo en un entorno institucional real, con conexión a bases de datos actualizadas y un flujo de predicción automatizado, que permita validar el desempeño del sistema con datos en producción.

- Realizar estudios comparativos con otros campus de la institución para validar la robustez y transferibilidad de los resultados obtenidos.

Estas líneas de trabajo permitirán evolucionar el sistema hacia una herramienta predictiva más precisa, explicable y operativa, fortaleciendo la gestión académica basada en analítica de datos y modelos de inteligencia artificial.

9 REFERENCIAS BIBLIOGRÁFICAS

[1] A. L. Cortes Comportamiento presupuestal en universidades públicas y privadas en Colombia. [online]. Disponible en: http://hdl.handle.net/10654/11947

[2] Laboratorio de Economía de la Educación (LEE) de la Pontificia Universidad Javeriana. (2024). Informe No. 103 Educación Superior en Colombia – parte I. Disponible en https://lee.javeriana.edu.co/publicaciones-y-documentos

[3] D. Sanz-Del Vecchio, J. García-Guiliany, R. Prieto-Pulido, y H. Medina-Carrascal, Plan de marketing educativo en universidades privadas; Marketing y Competitividad en las Organizaciones. Enfoques y Perspectivas. Barranquilla-Colombia: Ediciones Universidad Simón Bolívar. 197-225. Disponible en https://bonga.unisimon.edu.co/server/api/core/bitstreams/a4fc027b-677a-4f30-96d2-adc20afbec15/content

[4] Ministerio de Educación Nacional, "Bases Consolidadas de Estadísticas," 16 de noviembre de 2024. Disponible en: https://snies.mineducacion.gov.co/portal/ESTADISTICAS/Bases-consolidadas/.

[5] J. Boucourt y M. González, "Perfil socioeconómico y demográfico del estudiante de nuevo ingreso a la Universidad del Zulia. Análisis comparativo cohortes 98-99; 99-2000; 2000-2001; 2001-2002," Revista Venezolana de Ciencias Sociales, 2006.

[6] S. Cancino, O. O. Peña Mantilla, y J. A. Velasco Mendoza, "Condiciones socioeconómicas del estudiante de pregrado de la Universidad de Pamplona (Norte de Santander - Colombia)," Investigación & Desarrollo, vol. 22, no. 1, pp. 59-78, 2014.

[7] A. Garay, "El perfil de los estudiantes de nuevo ingreso de las universidades tecnológicas en México," El Cotidiano, no. 122, pp. 75-85, nov.-dic. 2003.

[8] E. Alarcón Montiel, Elección de carrera: motivos, procesos e influencias y sus efectos en la experiencia estudiantil de jóvenes universitarios de alto rendimiento académico, reencuentro, Análisis De Problemas Universitarios vol. 31, n.º 77, pp. 55-74, ene. 2019.

[9] L. Sandoval, "Algoritmos de aprendizaje automático para análisis y predicción de datos," Revista Tecnológica, no. 11, 2018. [En línea]. Disponible en: http://redicces.org.sv/jspui/bitstream/10972/3626/1/Art6_RT2018.pdf.

[10] S. J. Russell y P. Norvig, *Inteligencia artificial: un enfoque moderno*. España: Pearson Educación, 1996.

[11] T. Chen and C. Guestrin, "XGBoost: A Scalable Tree Boosting System," Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 2016.

[12] A. Géron, Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow, 2nd ed., O'Reilly Media, 2019.

[13] Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, V., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., DeleSalle, M., Olivier, E., & Duchesnay, É. (2011). Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, *12*, 2825–2830.

[14] Huang, X., Khetan, A., Cvitkovic, M., & Karnin, Z. (2020). TabTransformer: Tabular Data Modeling Using Contextual Embeddings. arXiv. https://arxiv.org/abs/2012.06678.

[15] L. Art, *Introducción al Aprendizaje Automático: Guía Simplificada para Principiantes*. (n.d.). (n.p.): Selfpublishing.

[16] J. Velasco Rebolledo, *Machine learning: Fundamentos, algoritmos y aplicaciones para los negocios, industria y finanzas*. España: Ediciones Diaz de Santos S.A., 2024.

[17] I. P. Pedroza Salgado, E. A. Pérez Sarabia, y A. S. Polo Vásquez, "Diseño de un modelo matemático para predecir la intención de matrícula en estudiantes de primer ingreso de la Universidad del Norte," Universidad del Norte, 2018. [En línea]. Disponible: http://hdl.handle.net/10584/7972

[18] M. L. Avila *Modelo de predicción de deserción estudiantil, apoyado en Tecnologías De Data Mining, en un curso de primera matrícula de la Escuela ECBTI De La UNAD.* [En línea]. Disponible en: https://repository.unad.edu.co/handle/10596/42544

[19] Polo Ahumada, L. J., & Medina Reyes, D. J. (2022). *Modelo Analítico Para La Predicción De La Deserción Estudiantil A Nivel De Pregrado En La Universidad Autónoma Del Caribe* [Tesis de pregrado, Universidad Autónoma del Caribe]. Repositorio Institucional de la Universidad Autónoma del Caribe. http://repositorio.uac.edu.co/handle/11619/4137?show=full

10 ANEXOS

Los siguientes anexos presentan las grillas de hiperparámetros exploradas durante el proceso de ajuste de los modelos de clasificación para cada programa académico. Su propósito es documentar de manera detallada las configuraciones evaluadas, complementando la información presentada en el capítulo de entrenamiento.

Anexo 1: Grillas de hiperparámetros exploradas por modelo -Administración de empresas

Modelo	Ronda 1 Randomized SearchCV	Ronda 2 GridSearchCV	Ronda3 GridSearchCV
Random Forest Classifier	n_estimators': randint(100, 300), max_depth': [None, 5, 10, 15, 20], min_samples_split': randint(2, 10), min_samples_leaf': randint(1, 5), max_features': ['sqrt', 'log2'], bootstrap': [True, False], criterion': ['gini', 'entropy'], class_weight': [None, 'balanced']	n_estimators': [250, 290, 320], max_depth': [4, 5, 6], min_samples_split': [4, 6, 8], min_samples_leaf': [1, 2, 3], criterion': ['entropy'], max_features': ['log2'], bootstrap': [False], class_weight': ['balanced']	n_estimators': [200,250, 290], max_depth': [3,4, 5], min_samples_split': [4, 6, 8], min_samples_leaf': [1, 2, 3], criterion': ['entropy'], max_features': ['log2'], bootstrap': [False], class_weight': ['balanced']
XGBoost	n_estimators': [100, 200, 300, 400], max_depth': [2, 3, 4, 5], learning_rate': [0.01, 0.05, 0.1, 0.2], subsample': [0.7, 0.8, 0.9, 1.0], colsample_bytree': [0.6, 0.8, 1.0], gamma': [0, 0.1, 0.3, 0.5], min_child_weight': [1, 3, 5, 7], reg_lambda': [0.5, 1, 2], reg_alpha': [0, 0.1, 0.5]	'n_estimators': [100, 150], 'max_depth': [2, 3], 'learning_rate': [0.005, 0.01], 'subsample': [0.7, 0.8], 'colsample_bytree': [0.8, 0.9], 'gamma': [0.3, 0.5], 'min_child_weight': [3, 5], 'reg_lambda': [1, 2], 'reg_alpha': [0.05, 0.1]	'n_estimators': [100, 200], 'max_depth': [2, 3], 'learning_rate': [0.005, 0.01], 'subsample': [0.7, 0.8], 'colsample_bytree': [0.8, 0.9], 'gamma': [0.2, 0.3], 'min_child_weight': [3, 5], 'reg_lambda': [1, 2], 'reg_alpha': [0.05, 0.1]
Logistic Regression	penalty': ['l1', 'l2', 'elasticnet'], C': np.logspace(-2, 1, 10), solver': ['saga'], l1_ratio': np.linspace(0, 1, 5), class_weight': [None, 'balanced']	penalty': ['l1', 'elasticnet'], C': np.linspace(0.3, 0.7, 5), solver': ['saga'], l1_ratio': [0.0, 0.1, 0.2, 0.3], class_weight': ['balanced']	penalty': ['l1', 'elasticnet'], solver': ['saga'], C': [0.3, 0.5, 0.7, 1.0], l1_ratio': [0.0, 0.2, 0.4], class_weight': ['balanced'], max_iter': [1000, 2000]
MLPClassifier	hidden_layer_sizes': [(50,), (100,), (100, 50), (128, 64, 32)], activation': ['relu', 'tanh'], solver': ['adam', 'lbfgs'], alpha': np.logspace(-4, -1, 4), learning_rate_init': [1e-3, 5e-4, 1e-4], batch_size': [32, 64], early_stopping': [True]	hidden_layer_sizes': [(50,), (100,), (100, 50)], activation': ['relu', 'tanh'], alpha': [0.0001, 0.001, 0.01], learning_rate_init': [0.001, 0.0005], solver': ['adam'], batch_size': [64], early_stopping': [True]	hidden_layer_sizes': [(80,), (100,), (120,), (100, 50)], activation': ['relu', 'tanh'], alpha': [0.00005, 0.0001, 0.0005, 0.001], learning_rate_init': [0.001, 0.0005, 0.0003], solver': ['adam'], batch_size': [32, 64], early_stopping': [True]
SVM	kernel': ['rbf', 'poly', 'sigmoid'], C': uniform(0.1, 10), gamma': uniform(1e-4, 1e-1), degree': randint(2, 5), class_weight': ['balanced']	kernel': ['sigmoid', 'rbf'], C': [4, 5, 6, 7, 8], gamma': [0.01, 0.012, 0.015, 0.018, 0.02], degree': [2, 3, 4], class_weight': ['balanced']	kernel': ['rbf', 'sigmoid', 'poly'], C': [6, 7, 8, 9, 10], gamma': [0.005, 0.01, 0.02, 0.05], degree': [2, 3], class_weight': ['balanced']
Tab Transformer Hugging	"dim": 32, "depth": 4, "heads": 8, "attn_dropout": 0.1,	"dim": 64, "depth": 6, "heads": 8, "attn_dropout": 0.2,	"dim": 16, "depth": 2, "heads": 4, "attn_dropout": 0.3,

Face Trainer	"ff_dropout": 0.1, "mlp_hidden_mults": (4, 2), "epochs": 10, "batch_size": 64, "learning_rate": 1e-3	"ff_dropout": 0.2, "mlp_hidden_mults": (4, 2), "epochs": 30, "batch_size": 32, "learning_rate": 5e-4	"ff_dropout": 0.3, "mlp_hidden_mults": (4, 2), "epochs": 40, "batch_size": 32, "learning_rate": 5e-4

Anexo 2: Grillas de hiperparámetros exploradas por modelo - Comercio Internacional

	Ronda 1 Randomized SearchCV	Ronda 2 GridSearchCV	Ronda3 GridSearchCV
Random Forest Classifier	n_estimators': [100, 200, 300, 400, 500], max_depth': [None, 5, 10, 15, 20, 25], min_samples_split': [2, 3, 4, 5, 6, 8, 10], min_samples_leaf': [1, 2, 3, 4, 5], max_features': ['sqrt', 'log2', 0.5, 0.7, None], bootstrap': [True, False], criterion': ['gini', 'entropy', 'log_loss'], class_weight': [None, 'balanced']	n_estimators': [150, 200, 250], max_depth': [15, 20], min_samples_split': [8, 10, 12], min_samples_leaf': [2, 3, 4], max_features': ['sqrt'], criterion': ['entropy'], bootstrap': [False, True], class_weight': ['balanced']	n_estimators': [150, 180], max_depth': [12, 15], min_samples_split': [6, 8], min_samples_leaf': [2, 3], max_features': ['sqrt'], criterion': ['entropy'], bootstrap': [False], class_weight': ['balanced']
XGBoost	n_estimators': randint(100, 400), max_depth': randint(3, 10), learning_rate': uniform(0.01, 0.2), subsample': uniform(0.7, 0.3), colsample_bytree': uniform(0.7, 0.3), min_child_weight': randint(1, 10), gamma': uniform(0, 0.3), reg_lambda': uniform(0.5, 2.0), reg_alpha': uniform(0, 1.0)	n_estimators': [150, 170], max_depth': [5, 6], learning_rate': [0.15, 0.20, 0.25], subsample': [0.9], colsample_bytree': [0.8, 0.9], gamma': [0.1, 0.2], min_child_weight': [1], reg_lambda': [1.0, 1.2], reg_alpha': [0.2, 0.3]	learning_rate': [0.2, 0.25], max_depth': [5, 6], gamma': [0.1, 0.2], subsample': [0.8, 0.9], colsample_bytree': [0.8], reg_alpha': [0.1, 0.2], reg_lambda': [1.0], n_estimators': [160, 170, 180],
Logistic Regression	penalty': ['l2', 'l1', 'elasticnet', None], solver': ['lbfgs', 'saga','Liblinear'], C': loguniform(1e-3, 10), l1_ratio': uniform(0, 1), fit_intercept': [True, False], class_weight': ['balanced'],	penalty': ['l1', 'elasticnet'], solver': ['saga'], C': [0.1, 0.25, 0.5, 1.0], l1_ratio': [0.0, 0.1, 0.25, 0.5], class_weight': ['balanced'], fit_intercept': [True]	solver': ['saga'], penalty': ['elasticnet'], l1_ratio': [0.3, 0.5, 0.7], C': [0.05, 0.1, 0.5, 1.0], fit_intercept': [True], class_weight': ['balanced']
MLPClassifier	hidden_layer_sizes': [(50,), (100,), (50, 25), (100, 50)], activation': ['relu', 'tanh'], solver': ['adam', 'lbfgs'], alpha': np.logspace(-5, -2, 6), learning_rate_init': np.logspace(-4, -2, 6), batch_size': [16, 32, 64], learning_rate': ['constant', 'adaptive']	hidden_layer_sizes': [(100,), (80, 40)], activation': ['relu'], solver': ['adam'], alpha': [0.001, 0.002], learning_rate_init': [0.002, 0.004], learning_rate': ['adaptive'], batch_size': [32, 64]	hidden_layer_sizes': [(40,), (60,)], alpha': [0.005, 0.01], learning_rate_init': [0.002, 0.003], solver': ['adam'], activation': ['relu', 'tanh'], learning_rate': ['adaptive'], batch_size': [32], max_iter': [1000]
SVM	kernel': ['rbf', 'poly', 'sigmoid'], C': uniform(0.1, 10), gamma': uniform(1e-4, 1e-1), degree': randint(2, 5), class_weight': ['balanced']	C': [0.1, 1, 5], gamma': ['scale', 0.01, 0.1], kernel': ['rbf', 'sigmoid'], class_weight': ['balanced'],	C': [3, 5, 8], gamma': [0.05, 0.1, 0.2], kernel': ['rbf'], class_weight': ['balanced'],
Tab Transformer Hugging Face Trainer	"dim": 32, "depth": 4, "heads": 8, "attn_dropout": 0.1, "ff_dropout": 0.1, "mlp_hidden_mults": (4, 2), "epochs": 10,	"dim": 16, "depth": 3, "heads": 4, "attn_dropout": 0.2, "ff_dropout": 0.2, "mlp_hidden_mults": (2,), "epochs": 20,	"dim": 16, "depth": 2, "heads": 4, "attn_dropout": 0.3, "ff_dropout": 0.3, "mlp_hidden_mults": (2,), "epochs": 30,

	"batch_size": 64, "learning_rate": 1e-3	"batch_size": 32, "learning_rate": 5e-4, "weight_decay": 1e-4	"batch_size": 32, "learning_rate": 3e-4, "weight_decay": 5e-4,

Anexo 3: Grillas de hiperparámetros exploradas por modelo – Comunicación Social

Modelo	Ronda 1 Randomized SearchCV	Ronda 2 GridSearchCV	Ronda3 GridSearchCV
Random Forest Classifier	n_estimators': [100, 200, 300, 400, 500], max_depth': [3, 5, 7, 10, 12, None], min_samples_split': [2, 5, 10, 15], min_samples_leaf': [1, 2, 4, 6], max_features': ['sqrt', 'log2', None], bootstrap': [True, False], class_weight':['balanced', 'balanced_subsample'], criterion': ['gini', 'entropy', 'log_loss']	n_estimators': [400, 500, 600], max_depth': [8, 10, 12], min_samples_split': [3, 5, 7], min_samples_leaf': [2, 4, 6], max_features': ['sqrt'], criterion': ['gini'], class_weight': ['balanced'], bootstrap': [True]	n_estimators': [350, 400, 450], max_depth': [7, 8, 9], min_samples_split': [2, 3, 4], min_samples_leaf': [3, 4, 5], max_features': ['sqrt'], bootstrap': [True], class_weight': ['balanced'], criterion': ['gini']
XGBoost	n_estimators': [100, 200, 300, 400, 500, 600], learning_rate': [0.01, 0.05, 0.1, 0.2, 0.3], max_depth': [3, 4, 5, 6, 8, 10], min_child_weight': [1, 3, 5, 7], gamma': [0, 0.1, 0.2, 0.3, 0.5], subsample': [0.6, 0.7, 0.8, 0.9, 1.0], colsample_bytree': [0.6, 0.7, 0.8, 0.9, 1.0], reg_alpha': [0, 0.01, 0.1, 0.5, 1], reg_lambda': [0.5, 1, 1.5, 2, 3], booster': ['gbtree', 'dart'], tree_method': ['hist']	n_estimators': [150, 200], learning_rate': [0.05, 0.07], max_depth': [3, 4], min_child_weight': [6, 7], gamma': [0.4, 0.5], subsample': [0.9, 1.0], colsample_bytree': [0.8], reg_alpha': [0.01], reg_lambda': [1.0], booster': ['dart'], tree_method': ['hist']	n_estimators': [120, 150], learning_rate': [0.04, 0.05], max_depth': [3, 4], min_child_weight': [6, 7], gamma': [0.4, 0.5], subsample': [0.9, 1.0], colsample_bytree': [0.8], reg_alpha': [0.0, 0.01], reg_lambda': [1.0], booster': ['dart'], tree_method': ['hist']
Logistic Regression	solver': ['lbfgs'], C': np.logspace(-3, 2, 10), fit_intercept': [True, False], class_weight': [None, 'balanced'] penalty': ['l1', 'l2', 'elasticnet'], solver': ['saga'], l1_ratio': np.linspace(0, 1, 5),	solver': ['saga'], penalty': ['l2'], C': [0.01, 0.02, 0.04, 0.06, 0.08, 0.1], fit_intercept': [False, True], class_weight': ['balanced'], max_iter': [1000] penalty': ['elasticnet'], l1_ratio': [0.8, 0.9, 1.0],	solver': ['saga'], penalty': ['l2'], C': [0.05, 0.08, 0.1, 0.12, 0.15], fit_intercept': [False, True], class_weight': ['balanced'], max_iter': [1000]
MLPClassifier	"hidden_layer_sizes": [(50,), (100,), (50, 25), (100, 50), (100, 100), (128, 64, 32), (64, 64)], "activation": ["relu", "tanh", "logistic"], "solver": ["adam", "lbfgs", sgd"], "learning_rate": ["constant", "adaptive"], "learning_rate_init": np.logspace(-4, -2, 5), "alpha": np.logspace(-5, -2, 5), "batch_size": [16, 32, 64, 128], "max_iter": [500], "early_stopping": [True],	"solver": ["lbfgs", "adam"], "activation": ["relu", "tanh"], "hidden_layer_sizes": [(64, 64), (100, 50), (128, 64),], "alpha": [0.005, 0.01, 0.02], "batch_size": [16, 32], "learning_rate": ["adaptive"], "learning_rate_init": [0.00005, 0.0001, 0.0002], "early_stopping": [True], "max_iter": [500],	"solver": ["lbfgs"], "activation": ["relu"], "hidden_layer_sizes": [(80, 40),(100, 50),(120, 60),], "alpha": [0.01, 0.02, 0.03], "batch_size": [16], "learning_rate": ["adaptive"], "learning_rate_init": [4e-05, 5e-05, 6e-05], "early_stopping": [True], "max_iter": [600],

SVM	C': np.logspace(-2, 2, 20), # Regularización kernel': ['linear', 'poly', 'rbf', 'sigmoid'], degree': [2, 3, 4], # Solo aplica con kernel='poly' gamma': ['scale', 'auto', 0.001, 0.01, 0.1, 1], coef0': [0.0, 0.1, 0.5, 1.0],# Solo aplica con poly/sigmoid shrinking': [True, False], class_weight': [None, 'balanced'], probability': [True],	C': [0.5, 0.7, 0.8, 1.0, 1.2], kernel': ['rbf'], # mejor kernel encontrado gamma': ['scale', 0.01, 0.1], degree': [3], # fijo (no afecta al rbf) coef0': [0.0, 0.5, 1.0], shrinking': [True], class_weight': ['balanced'], probability': [True],	C': [0.8, 1.0, 1.2, 1.5], gamma': [0.05, 0.08, 0.1, 0.12, 0.15], kernel': ['rbf'], coef0': [0.0], degree': [3], shrinking': [True], class_weight': ['balanced'], probability': [True],
Tab Transformer Hugging Face Trainer	"dim": 32, "depth": 4, "heads": 8, "attn_dropout": 0.1, "ff_dropout": 0.1, "mlp_hidden_mults": (4, 2), "epochs": 10, "batch_size": 64, "learning_rate": 1e-3	"dim": 64, "depth": 6, "heads": 8, "attn_dropout": 0.05, "ff_dropout": 0.05, "mlp_hidden_mults": (2, 1), "epochs": 30, "batch_size": 64, "learning_rate": 5e-4	"dim": 64, "depth": 5, "heads": 8, "attn_dropout": 0.1, "ff_dropout": 0.1, "mlp_hidden_mults": (4, 2), "epochs": 40, "batch_size": 64, "learning_rate": 3e-4

Anexo 4: Grillas de hiperparámetros exploradas por modelo – Contaduría Pública

	Ronda 1 Randomized SearchCV	Ronda 2 GridSearchCV	Ronda3 GridSearchCV
Random Forest Classifier	n_estimators': [100, 200, 300, 500, 800, 1000], max_depth': [None, 5, 10, 15, 20, 30, 50], min_samples_split': [2, 5, 10, 15], min_samples_leaf': [1, 2, 4, 6], max_features': ['sqrt', 'log2', 0.5, 0.7, None], 'bootstrap': [True, False], 'class_weight':[None, 'balanced']	'n_estimators': [200, 300, 400], 'max_depth': [4, 5, 6], 'min_samples_split': [2, 3], 'min_samples_leaf': [3, 4, 5], 'max_features': ['sqrt'], 'bootstrap':[True], 'class_weight': ['balanced']	'n_estimators': [150, 200, 250], 'max_depth': [3, 4, 5], 'min_samples_split': [2, 3], 'min_samples_leaf': [2, 3, 4], 'max_features': ['sqrt'], 'bootstrap': [True], 'class_weight': ['balanced']
XGBoost	'booster': ['gbtree', 'dart'], 'learning_rate': np.linspace(0.01, 0.3, 10), 'max_depth': [3, 4, 5, 6, 7, 8], 'min_child_weight': [1, 2, 3, 4, 5], 'subsample': np.linspace(0.6, 1.0, 5), 'colsample_bytree': np.linspace(0.6, 1.0, 5), 'gamma': [0, 0.1, 0.2, 0.3, 0.5], 'reg_lambda': [0.1, 1, 5, 10, 20], 'reg_alpha': [0, 0.1, 0.5, 1], 'n_estimators': [100, 200, 300, 400, 500],	'learning_rate': [0.08, 0.1], 'max_depth': [4, 5], 'min_child_weight': [1, 2], 'n_estimators': [100, 120], 'subsample': [0.9, 1.0], 'colsample_bytree': [0.9, 1.0], 'booster': ['dart']	'booster': ['dart'], 'learning_rate': [0.1, 0.12], 'n_estimators': [100, 150], 'max_depth': [3, 4], 'min_child_weight': [1, 2], 'subsample': [0.9, 1.0], 'colsample_bytree': [0.9, 1.0], 'reg_lambda': [0.1, 1], 'reg_alpha': [0, 0.5],
Logistic Regression	'solver': ['lbfgs'], 'C': np.logspace(-3, 2, 10), 'fit_intercept': [True, False], 'class_weight': [None, 'balanced'] 'penalty': ['l1', 'l2', 'elasticnet'], 'solver': ['saga'], 'l1_ratio': np.linspace(0, 1, 5),	'penalty': ['l1'], 'solver': ['saga'], 'C': np.linspace(1.5, 3.0, 6), 'fit_intercept': [False, True], 'class_weight': [None, 'balanced'], 'max_iter': [1000] 'penalty': ['elasticnet'], 'l1_ratio': np.linspace(0.7, 1.0, 4),	'penalty': ['elasticnet'], 'solver': ['saga'], 'C': [0.8, 1.0, 1.2, 1.5], 'l1_ratio': [0.8, 0.9, 1.0], 'fit_intercept': [False, True], 'class_weight': [None, 'balanced'], 'max_iter': [1500]

MLPClassifier	'hidden_layer_sizes': [(32,), (64,), (32, 16), (64, 32), (128, 64)], 'activation': ['relu', 'tanh', 'logistic'], 'solver': ['adam', 'lbfgs'], 'alpha': uniform(1e-5, 1e-2), 'learning_rate_init': uniform(1e-4, 5e-3), 'batch_size': [16, 32, 64], 'learning_rate': ['constant', 'adaptive'], 'early_stopping': [True],	'hidden_layer_sizes': [(32,), (64,), (128,)], 'activation': ['relu', 'tanh'], 'solver': ['lbfgs', 'adam'], 'alpha': [0.009, 0.01, 0.02], 'learning_rate': ['constant', 'adaptive'], 'learning_rate_init': [0.0001, 0.0002], 'batch_size': [32, 64], 'early_stopping': [True]	'hidden_layer_sizes': [(32,), (64,), (128,), (64, 32)], 'activation': ['relu'], 'solver': ['lbfgs'], 'alpha': [0.005, 0.01, 0.02, 0.03], 'learning_rate_init': [5e-5, 1e-4, 2e-4], 'batch_size': [32], 'early_stopping': [True], 'max_iter': [1000, 1500]
SVM	'kernel': ['rbf', 'poly', 'sigmoid', linear'], 'C': np.logspace(-2, 2, 10), 'gamma': ['scale', 'auto', 0.001, 0.01, 0.1, 1], 'degree': [2, 3], 'coef0': [0, 0.5, 1], 'shrinking': [True, False], 'tol': [1e-3, 1e-4], 'max_iter': [-1, 1000, 2000], 'class_weight': ['balanced', None],	'kernel': ['poly', 'rbf'], 'C': [1, 5, 10, 15], 'gamma': [0.0005, 0.001, 0.005], 'degree': [2, 3], 'coef0': [0, 1], 'shrinking': [True], 'tol': [1e-3], 'max_iter': [2000],	'kernel': ['poly', 'rbf'], 'C': [0.1, 0.5, 1, 2], 'gamma': [0.0005, 0.001, 0.005], 'degree': [2], 'coef0': [0, 0.5, 1], 'shrinking': [True], 'tol': [1e-3], 'max_iter': [3000],
Tab Transformer Hugging Face Trainer	"dim": 32, "depth": 4, "heads": 8, "attn_dropout": 0.1, "ff_dropout": 0.1, "mlp_hidden_mults": (4, 2), "epochs": 10, "batch_size": 64, "learning_rate": 1e-3	"dim": 64, "depth": 3, "heads": 4, "attn_dropout": 0.2, "ff_dropout": 0.2, "mlp_hidden_mults": (4, 2), "epochs": 25, "batch_size": 32, "learning_rate": 5e-4, "weight_decay": 1e-4,	"dim": 48, "depth": 6, "heads": 8, "attn_dropout": 0.2, "ff_dropout": 0.2, "mlp_hidden_mults": (4, 2), "batch_size": 32, "epochs": 50, "learning_rate": 5e-4, "weight_decay": 1e-4,

Anexo 5: Grillas de hiperparámetros exploradas por modelo – Derecho

Modelo	Ronda 1 Randomized SearchCV	Ronda 2 GridSearchCV	Ronda3 GridSearchCV
Random Forest Classifier	n_estimators': randint(100, 500), max_depth': [None, 5, 10, 15, 20 min_samples_split': randint(2, 10 min_samples_leaf': randint(1, 5), max_features': ['sqrt', 'log2'], bootstrap': [True, False criterion': ['gini', 'entropy'], class_weight': [None, 'balanced']	n_estimators': [200,250, 290], max_depth': [None,4, 5, 6], min_samples_split': [2,4, 6], min_samples_leaf': [3,4,5], criterion': ['entropy'], max_features': ['sqrt'], bootstrap': [True,False], class_weight': ['balanced']	n_estimators': [200,250, 290], max_depth': [4, 5,6], min_samples_split': [2,4, 6], min_samples_leaf': [4,5,6], criterion': ['entropy'], max_features': ['sqrt'], bootstrap': [True], class_weight': ['balanced']
XGBoost	n_estimators': randint(150, 400), max_depth': randint(3, 8), learning_rate': uniform(0.01, 0.2), subsample': uniform(0.7, 0.3), colsample_bytree': uniform(0.7, 0.3), gamma': uniform(0, 1.5), min_child_weight': randint(1, 6)	n_estimators': [220, 240,260], max_depth': [2, 3, 4], learning_rate': [0.015, 0.018, 0.021], min_child_weight': [3, 4, 5], gamma': [0.10, 0.15, 0.20], subsample': [0.85, 0.90, 0.95], colsample_bytree': [0.65, 0.70, 0.75],	n_estimators': [200, 220, 240], learning_rate': [0.018, 0.021, 0.024], max_depth': [3, 4, 5], min_child_weight': [4, 5, 6], gamma': [0.15, 0.2, 0.25], subsample': [0.8, 0.85], colsample_bytree': [0.6, 0.65, 0.7],
Logistic Regression	penalty': ['l1', 'l2', 'elasticnet', 'none'], C': loguniform(1e-3, 1e2), solver': ['saga', 'liblinear'],	penalty': ['l1', 'l2','elasticnet'], C': [0.03, 0.05, 0.07, 0.1, 0.2, 0.5],	penalty': ['l1', 'l2', 'elasticnet'], C': [0.05, 0.1, 0.2, 0.5], solver': ['saga'],

	l1_ratio': uniform(0, 1), class_weight': [None, 'balanced'], fit_intercept': [True, False], max_iter': [2000, 3000, 5000], tol': [1e-4, 1e-3, 1e-2], warm_start': [True, False]	solver': ['liblinear', 'saga'], l1_ratio': [0, 0.25, 0.5], class_weight': [None, 'balanced'], max_iter': [3000], tol': [1e-4], fit_intercept': [True], warm_start': [True]	l1_ratio': [0, 0.25, 0.5], class_weight': ['balanced'], max_iter': [3000], warm_start': [True]
MLPClassifier	hidden_layer_sizes': [(50,), (100,), (50, 25), (100, 50)], activation': ['relu', 'tanh'], solver': ['adam', 'lbfgs'], alpha': np.logspace(-5, -1, 5), learning_rate_init': [0.001, 0.01, 0.05], max_iter': [5000], tol': [1e-4, 1e-3], warm_start': [False, True],	hidden_layer_sizes': [(100, 50),(80, 40),(50, 25)], activation': ['tanh', 'relu'], solver': ['adam', 'lbfgs'], alpha': [0.01, 0.05, 0.1], learning_rate_init': [0.001, 0.005, 0.01], max_iter': [5000], tol': [1e-4], warm_start': [True],	activation': ['tanh'], alpha': [0.001, 0.01, 0.05], learning_rate_init': [0.001, 0.005, 0.01], hidden_layer_sizes': [(80, 40), (60, 30),(100,), (50,),], solver': ['adam'], max_iter': [5000], tol': [1e-4], warm_start': [True]
SVM	kernel': ['rbf', 'poly', 'sigmoid'], C': uniform(0.1, 5), gamma': uniform(1e-4, 5e-2), degree': randint(2, 4), class_weight': ['balanced']	kernel': ['sigmoid', 'rbf'], C': [2, 3, 5, 7, 10], gamma': [0.001, 0.002, 0.003, 0.005, 0.01], degree': [2], class_weight': ['balanced']	kernel': ['sigmoid'], C': [5, 8, 10, 12, 15], gamma': [0.002, 0.0025, 0.003, 0.0035, 0.004], degree': [2], class_weight': ['balanced']
Tab Transformer Hugging Face Trainer	"dim": 32, "depth": 4, "heads": 8, "attn_dropout": 0.1, "ff_dropout": 0.1, "mlp_hidden_mults": (4, 2), "epochs": 10, "batch_size": 64, "learning_rate": 1e-3	"dim": 128, "depth": 2, "heads": 4, "attn_dropout": 0.1, "ff_dropout": 0.2, "mlp_hidden_mults": (4, 2), "batch_size": 32, "learning_rate": 3e-4, "weight_decay": 0.01	"dim": 128, "depth": 3, "heads": 8, "attn_dropout": 0.1, "ff_dropout": 0.3, "mlp_hidden_mults": (4, 2), "epochs": 25, "batch_size": 32, "learning_rate": 3e-4, "weight_decay": 0.01

Anexo 6: Grillas de hiperparámetros exploradas por modelo – Psicología

Modelo	Ronda 1 Randomized SearchCV	Ronda 2 GridSearchCV	Ronda3 GridSearchCV
Random Forest Classifier	n_estimators': np.arange(100, 600), max_depth': [None, 5, 10, 15, 20, 25, 30], min_samples_split': [2, 5, 10, 15, 20], min_samples_leaf': [1, 2, 4, 6, 8], max_features': ['sqrt', 'log2', None], bootstrap': [True, False], class_weight': [None, 'balanced', 'balanced_subsample'], criterion': ['gini', 'entropy', 'log_loss']	n_estimators': [90, 110, 130], max_depth': [15, 20, 25], min_samples_split': [3, 5, 7], min_samples_leaf': [4, 6, 8], max_features': ['log2', 'sqrt'], criterion': ['entropy', 'gini'], bootstrap': [False], class_weight': [None, 'balanced']	n_estimators': [120, 130, 140], max_depth': [18, 20, 22], min_samples_split': [2, 3, 4], min_samples_leaf': [3, 4, 5], max_features': ['log2'], criterion': ['entropy'], bootstrap': [False], class_weight': ['balanced']
XGBoost	n_estimators': np.arange(50, 300, 25), max_depth': np.arange(2, 10, 1), learning_rate': np.linspace(0.01, 0.3, 10),	n_estimators': [80, 100], learning_rate': [0.005, 0.01], max_depth': [8, 9], min_child_weight': [5, 6] gamma': [2, 3],	n_estimators': [90, 100, 110], learning_rate': [0.008, 0.01, 0.012], max_depth': [8, 9], min_child_weight': [4, 5, 6],

	subsample': np.linspace(0.6, 1.0, 5), colsample_bytree': np.linspace(0.6, 1.0, 5), gamma': np.linspace(0, 5, 6), min_child_weight': np.arange(1, 8,1), reg_lambda': np.linspace(0.1, 2, 10), reg_alpha': np.linspace(0, 1, 6), booster': ['gbtree', 'dart'], tree_method': ['hist']	subsample': [0.7, 0.8], colsample_bytree': [0.6, 0.7] reg_alpha': [0.5, 1.0], reg_lambda': [0.5, 0.75], booster': ['dart'], tree_method': ['hist']	gamma': [2.5, 3], colsample_bytree': [0.6], subsample': [0.8], reg_alpha': [0.5, 0.75], reg_lambda': [0.5, 0.75], booster': ['dart'], tree_method': ['hist']
Logistic Regression	solver': ['lbfgs'], C': loguniform(1e-3, 10), solver': ['liblinear'], penalty': ['l1', 'l2'], class_weight':['balanced'] solver': ['saga'], penalty': ['l1', 'l2', 'elasticnet'], l1_ratio': uniform(0, 1),	solver': ['saga'], penalty': ['elasticnet'], C': [0.05, 0.1, 0.15, 0.18, 0.2, 0.25, 0.3], l1_ratio': [0.2, 0.3, 0.38, 0.5, 0.6], class_weight':['balanced']	C': [0.03, 0.05, 0.07], l1_ratio': [0.4, 0.5, 0.6], penalty': ['elasticnet'], solver': ['saga'], class_weight':['balanced']
MLP Classifier	hidden_layer_sizes': [(50,), (100,), (50, 50), (100, 50), (100, 100), (64, 32), (128, 64)], activation': ['relu', 'tanh'], solver': ['adam', 'lbfgs'], alpha': uniform(1e-5, 1e-3), learning_rate': ['constant','adaptive'], learning_rate_init': uniform(1e-4, 5e-3), batch_size': [32, 64, 128], early_stopping': [True],	hidden_layer_sizes': [(40, 40), (50, 50), (60, 60)], activation': ['relu', 'tanh'], solver': ['adam'], alpha': [1e-4, 1e-3, 5e-3], learning_rate': ['constant', 'adaptive'], learning_rate_init': [0.001, 0.003], batch_size': [32], early_stopping': [True], max_iter': [2000]	hidden_layer_sizes': [(40, 40), (50, 50), (60, 60)], activation': ['relu'], solver': ['adam'], alpha': [5e-5, 1e-4, 5e-4], learning_rate': ['constant'], learning_rate_init': [0.002, 0.003, 0.004], batch_size': [32], early_stopping': [True], max_iter': [2000]
SVM	C': np.logspace(-3, 2, 20), kernel': ['linear', 'poly', 'rbf', 'sigmoid'], gamma': ['scale', 'auto'] + list(np.logspace(-3, 1, 6)), degree': [2, 3, 4], coef0': [0.0, 0.1, 0.5, 1.0], class_weight': ['balanced'],	kernel': ['linear'], C': [10, 15, 20], gamma': [0.1, 0.25, 0.5], degree': [3, 4], coef0': [0.0, 0.5, 1.0], class_weight': ['balanced']	kernel': ['linear', 'rbf'], C': [5, 10, 15, 20, 25, 30], gamma': [0.01, 0.05, 0.1, 0.2, 0.5], degree': [2, 3], coef0': [0.0, 0.5, 1.0], class_weight': ['balanced']
Tab Transformer Hugging Face Trainer	"dim": 32, "depth": 4, "heads": 8, "attn_dropout": 0.1, "ff_dropout": 0.1, "mlp_hidden_mults": (4, 2), "epochs": 10, "batch_size": 64, "learning_rate": 1e-3	"dim": 48, "depth": 5, "heads": 8, "attn_dropout": 0.2, "ff_dropout": 0.2, "mlp_hidden_mults": (4, 2), "weight_decay": 1e-4, "epochs": 40, "batch_size": 64, "learning_rate": 5e-4, "	"dim": 32,) "depth": 3, "heads": 4, "attn_dropout": 0.1, "ff_dropout": 0.1, "weight_decay": 1e-5, "epochs": 50, "batch_size": 64, "learning_rate": 3e-4, "mlp_hidden_mults": (2,),