ESCUELA SUPERIOR POLITÉCNICA DEL LITORAL

Facultad de Ingeniería en Electricidad y Computación

Predicción de la Emisión de Tarjetas de Crédito por Instituciones Financieras del Ecuador con Técnicas de Aprendizaje Automático.

PROYECTO DE TITULACIÓN

Previo la obtención del Título de:

Magister en Ciencias de Datos

Presentado por:

Edwin Xavier Maita Tepan

GUAYAQUIL - ECUADOR

Año: 2024

DEDICATORIA

A mi prometida Nube, mi inspiración constante, por su amor incondicional, su paciencia infinita. Y a Dante, por su fiel compañía. Son parte esencial de este logro.

A mis Padres, Angel y Maria, por su amor inagotable, su valores, enseñanzas y aliento a seguir formándome profesionalmente, mi amor y gratitud a ustedes.

A mis hermanos, mi cuñado y mis sobrinos, por su cariño y apoyo incondicional durante esta etapa. Gracias por estar siempre a mi lado.

AGRADECIMIENTOS

Mi más sincero agradecimiento a ESPOL y mis profesores de la maestría, quienes generosamente compartieron sus conocimientos y experiencias en Ciencia de Datos.

A mi tutora Karen y a mi revisora María Isabel, por su invaluable aporte, su experticia y los valiosos consejos brindados durante la elaboración de este proyecto.

Declaración Expresa

Yo Edwin Xavier Maita Tepán acuerdo y reconozco que: La titularidad de los derechos patrimoniales de autor (derechos de autor) del proyecto de graduación corresponderá al autor, sin perjuicio de lo cual la ESPOL recibe en este acto una licencia gratuita de plazo indefinido para el uso no comercial y comercial de la obra con facultad de sublicenciar, incluyendo la autorización para su divulgación, así como para la creación y uso de obras derivadas. En el caso de usos comerciales se respetará el porcentaje de participación en beneficios que corresponda a favor del autor o autores. El o los estudiantes deberán procurar en cualquier caso de cesión de sus derechos patrimoniales incluir una cláusula en la cesión que proteja la vigencia de la licencia aquí concedida a la ESPOL.

La titularidad total y exclusiva sobre los derechos patrimoniales de patente de invención, modelo de utilidad, diseño industrial, secreto industrial, secreto empresarial, derechos patrimoniales de autor sobre software o información no divulgada que corresponda o pueda corresponder respecto de cualquier investigación, desarrollo tecnológico o invención realizada por mí/nosotros durante el desarrollo del proyecto de graduación, pertenecerán de forma total, exclusiva e indivisible a la ESPOL, sin perjuicio del porcentaje que me/nos corresponda de los beneficios económicos que la ESPOL reciba por la explotación de mi/nuestra innovación, de ser el caso.

En los casos donde la Oficina de Transferencia de Resultados de Investigación (OTRI) de la ESPOL comunique al autor que existe una innovación potencialmente patentable sobre los resultados del proyecto de graduación, no se realizará publicación o divulgación alguna, sin la autorización expresa y previa de la ESPOL.

Guayaquil, 30 de noviembre del 2024.

Firmado electrónicamente por:
EDWIN XAVIER
MAITA TEPAN

Edwin Xavier Maita Tepan

EVALUADORES

Firmado electrónicamente por:
KAREN PRISCILLA CALVA YAGUANA

MSc. Karen Priscilla Calva Yaguana

Tutor del Proyecto

Firmado electrónicamente por:
MARIA ISABEL MERA COLLANTES

María Isabel Mera Collantes, Ph.D

Evaluador del proyecto

RESUMEN

La emisión de tarjetas de crédito juega un papel crucial para las entidades financieras, ya que les permite generar ingresos mediante comisiones, intereses y cuotas, al mismo tiempo que brinda beneficios tanto a los clientes como a las instituciones. Para las entidades, prever con precisión la demanda de tarjetas es fundamental, ya que les permite optimizar la asignación de recursos, diseñar campañas de marketing más efectivas y gestionar de manera más eficiente los riesgos financieros.

Se desarrolló un pipeline predictivo basado en series de tiempo utilizando un enfoque de AutoML, que automatiza el proceso de entrenamiento, evaluación y selección del modelo más adecuado mediante métricas como el coeficiente de determinación (R^2) y el error cuadrático medio (MSE). Los resultados mostraron coeficientes de determinación (R^2) de 0.5294 para el agregado de todas las entidades, 0.7822 para Diners Club y 0.6127 para Produbanco, con un promedio general de 64.8%, lo que valida la capacidad del modelo para capturar el comportamiento y las tendencias de cada institución.

Con base en estos resultados, se desarrolló una herramienta web interactiva que permite a las entidades financieras predecir la demanda futura de tarjetas de crédito, optimizando así la toma de decisiones estratégicas. La herramienta incluye un tablero interactivo mediante el cual los usuarios pueden explorar las diferentes características de los datos y visualizar el desempeño de las entidades. Además, facilita la comparación de estos resultados con los de sus competidores, mejorando la eficacia de las políticas comerciales y de marketing al proporcionar una visión más clara de las tendencias del mercado.

Palabras Clave: Emisión de Tarjetas de Crédito, AutoML, Serie de Tiempo, Predicción

ABSTRACT

The issuance of credit cards plays a crucial role for financial institutions, as it allows them to generate revenue through fees, interest, and installments, while also providing benefits to both customers and the institutions. For these entities, accurately forecasting credit card demand is essential, as it enables them to optimize resource allocation, design more effective marketing campaigns, and manage financial risks more efficiently.

A predictive pipeline based on time series analysis was developed using an AutoML approach, which automates the training, evaluation, and selection of the most appropriate model using metrics such as the coefficient of determination (R^2) and mean squared error (MSE). The results showed R^2 values of 0.5294 for the aggregate of all entities, 0.7822 for Diners Club, and 0.6127 for Produbanco, with an overall average of 64.8%, which validates the model's ability to capture the behavior and trends of each institution.

Based on these results, an interactive web tool was developed, allowing financial institutions to predict future credit card demand, thus optimizing strategic decision-making. The tool includes an interactive dashboard through which users can explore various data features and visualize the performance of the entities. Additionally, it facilitates the comparison of these results with those of competitors, improving the effectiveness of commercial and marketing policies by providing a clearer view of market trends.

Keywords: Credit Card Issuance, AutoML, Time Series, Prediction

ÍNDICE GENERAL

ABREVIATURAS

ESPOL	Escuela Superior Politécnica del Litoral
ML	Aprendizaje de Máquinas
IA	Inteligencia Artificial
AUTOML	Aprendizaje de Máquinas Automatizado
R^2	Coeficiente de Determinación
MAE	Error Absoluto Medio
ETS	Modelos de Suavizado Exponencial en Espacio de Estados

ÍNDICE DE FIGURAS

ÍNDICE DE TABLAS

CAPÍTULO 1

1. INTRODUCCIÓN

La emisión de tarjetas de crédito es una actividad fundamental para las entidades financieras, ofreciendo numerosos beneficios tanto para la institución como para los clientes. Las tarjetas de crédito permiten a los clientes disponer de una línea de crédito flexible, facilitando sus transacciones diarias y ofreciendo beneficios adicionales como programas de recompensas. Para las entidades financieras, la cantidad de tarjetas de crédito emitidas representa una fuente significativa de ingresos a través de intereses, comisiones y cuotas anuales.

En un entorno competitivo y en constante cambio, las entidades financieras buscan optimizar sus procesos y estrategias de emisión de tarjetas de crédito. La capacidad de predecir la demanda de tarjetas de crédito es crucial, ya que permite a las instituciones planificar de manera eficiente la asignación de recursos, diseñar campañas de marketing efectivas y gestionar los riesgos crediticios. Además, una previsión precisa de la demanda facilita la personalización de ofertas y servicios, mejorando la experiencia del cliente y cumpliendo con las regulaciones vigentes.
En este contexto, el uso de técnicas de aprendizaje automático se presenta como una herramienta poderosa para predecir la emisión de tarjetas de crédito. Al analizar datos históricos y características relevantes, los algoritmos de aprendizaje automático pueden identificar patrones y tendencias que permitan anticipar la demanda futura. Esto no solo optimiza la planificación financiera y operativa, sino que también impulsa la innovación y la competitividad al identificar oportunidades para el desarrollo de nuevos productos y servicios.

Este proyecto se centra en el desarrollo de un modelo predictivo utilizando técnicas de aprendizaje automático para predecir la cantidad tarjetas de crédito que emitirá una entidad financiera. A través de un enfoque estructurado que incluye la adquisición, preprocesamiento y análisis de datos, así como la implementación y

evaluación de modelos predictivos, se busca proporcionar una herramienta efectiva para las entidades financieras en la toma de decisiones estratégicas.

1.1 Descripción del problema

Emitir un gran número de tarjetas de crédito ofrece numerosos beneficios a una entidad financiera, incluyendo la generación de ingresos a través de intereses, comisiones por transacciones y cuotas anuales, así como la fidelización y retención de clientes mediante programas de recompensas y relaciones a largo plazo (Xu & Qu, 2024). Además, permite la diversificación de productos y la ampliación de la base de clientes, facilitando el cross-selling de otros productos financieros (Boustani et al., 2023).

Asimismo, una mayor emisión de tarjetas incrementa la participación de mercado y permite aprovechar economías de escala, reduciendo costos operativos. Todo esto, junto con la flexibilidad financiera que proporcionan las tarjetas de crédito a los clientes y empresas, fortalece la posición competitiva de la entidad financiera y fomenta la innovación continua para adaptarse a las tendencias del mercado (Zywicki, 2010).

Proyectar la cantidad de tarjetas de crédito que emitirá una entidad financiera permite gestionar eficientemente sus recursos, optimizar la planificación financiera y asegurar la disponibilidad de servicios adecuados para soportar el crecimiento previsto (Stewart, 2011).

1.2 Justificación del problema

Predecir la emisión de tarjetas de crédito es fundamental para las entidades financieras debido a los múltiples beneficios que proporciona. Al anticipar con precisión la demanda de tarjetas, las instituciones pueden optimizar la asignación de recursos financieros y operativos, planificar estrategias de marketing más efectivas y gestionar de manera más eficiente los riesgos crediticios (Grodzicki, n.d.). Esta capacidad de anticipación también permite mejorar la experiencia del cliente al personalizar ofertas y servicios, agilizar los procesos de solicitud y emisión, y cumplir con políticas y regulaciones.

Adicional, la predicción de la emisión de tarjetas impulsa la rentabilidad de la entidad financiera al maximizar los ingresos por intereses, comisiones y cuotas. Lo cual permite la innovación y la competitividad al identificar oportunidades para el desarrollo de nuevos productos y servicios que se ajusten a las necesidades del mercado (Sargeant, 2023).

1.3 Objetivos

1.3.1 Objetivo General

Implementar un modelo predictivo basado en técnicas de aprendizaje automático mediante el análisis de series de tiempo, que permita estimar con precisión la cantidad de tarjetas de crédito que serán emitidas por instituciones financieras del Ecuador en un periodo especifico.

1.3.2 Objetivos Específicos

- Estudiar el conjunto de datos para identificar características y variables relevantes para cada institución.
- Implementar un modelo de aprendizaje automático utilizando características de las tarjetas de crédito que son emitidas para las dos principales entidades financieras del conjunto de datos, analizando posibles sesgos relacionado a la inteligencia artificial.
- Evaluar el rendimiento del algoritmo utilizando métricas de evaluación de predicción, tales como el error absoluto medio, error cuadrático medio y el coeficiente de determinación, y plantear formas de monitoreo y ajustes continuamente.
- Diseñar un tablero de control interactivo que permita visualizar las principales características de los tipos de tarjetas de crédito emitidas y realizar predicciones basadas en el modelo desarrollado.

1.4 Metodología:

La metodología empleada para la solución de los objetivos planteados se fundamenta en el pipeline de ciencia de datos con un enfoque de aprendizaje de máquinas automático. Este enfoque permite la selección, optimización y validación de modelos predictivos de manera estructurada y eficiente. A continuación, se describen las etapas del proceso, representadas en la Figura 1.1.

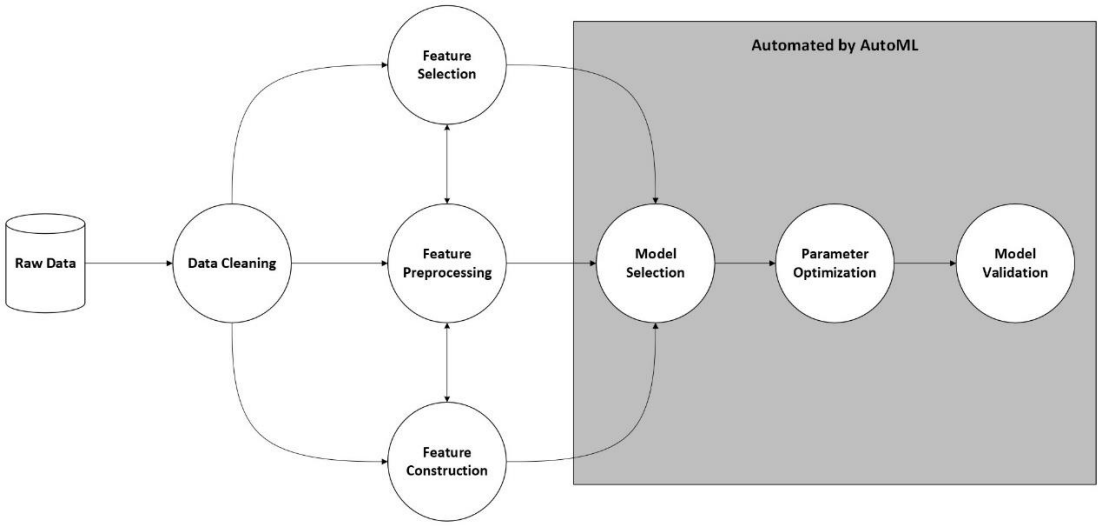


Figura 1.1. Pipeline de Ciencia de Datos (Olson et al., 2016).

- Adquisición de datos: Recolectar datos públicos relacionados a tarjetas de crédito de la superintendencia de bancos del Ecuador.

- Preprocesamiento de Datos: Preparar los datos para su análisis mediante normalización, eliminación de ruido y codificación de variables, asegurando que sean adecuados para los algoritmos de aprendizaje automático.

- Selección de Características: Identificar y retener las variables más relevantes, reduciendo la dimensionalidad del conjunto de datos y mejorando la eficiencia del modelo.

- Construcción de Características: Crear nuevas variables derivadas de las existentes, capturando relaciones complejas entre las características para mejorar la precisión del modelo.

- Selección del Modelo: Evaluar y seleccionar el algoritmo más adecuado (e.g., Arima, ETS) para la tarea específica, maximizando el rendimiento.

- Optimización de Hiperparámetros: Ajustar automáticamente los parámetros clave del modelo para maximizar su precisión y efectividad.

- Validación del Modelo: Garantizar que el modelo es robusto y generaliza bien a nuevos datos mediante técnicas como la validación cruzada y la evaluación en conjuntos de prueba.

- Generación y Selección de Tuberías: Combinar y optimizar los pasos anteriores en un pipeline eficiente, utilizando programación para equilibrar precisión y simplicidad.

1.5 Resultados Esperados:

Precisión en las predicciones: Se espera que los modelos predictivos desarrollados sean capaces de proporcionar estimaciones con un coeficiente de determinación superior al 0.6, garantizando presión y confiables sobre la demanda futura de tarjetas de crédito.

1.6 Conjunto de Datos:

La fuente de datos se extrae de la página oficial de la superintendencia de bancos de ecuador en la sección de Estadísticas de Consumos con Tarjetas (crédito, débito, prepago) (A12), contiene información de las entidades financieras del ecuador de los años 2021 hasta agosto 2024 (Superintendencia de Bancos de Ecuador, 2024). A continuación, se detalla en Tabla 1.1 la información que contiene.

Tabla 1.1 Descripción de los Campos del Conjunto de Datos (Elaboración Propia).

Característica	Tipo de Dato	Descripción
Entidad Emisora	Categórico	Entidad financiera
Tipo de identificación	Categórico	Cédula/RUC
Tipo de Tarjeta Habiente	Categórico	Titular/Adicional
Tipo de medio de pago	Categórico	Crédito/Debito/Prepago
Marca de la tarjeta	Categórico	VISA/MC/DINERS/etc
Clasificación de la tarjeta	Categórico	Empresarial/Natural
Segmento de la tarjeta	Categórico	A/A+/B++
Tipo de Consumo	Categórico	Corriente/Diferido
Canal	Categórico	Oficina/Pos/Internet
Ubicación geográfica	Categórico	Nacional/Exterior
Nro de consumos	Numérico	Valor agrupado
Monto de facturación	Numérico	Valor agrupado
Nro de tarjetas activas	Numérico	Valor agrupado
Mes	Temporal	Mes del reporte
Año	Temporal	Año del reporte

CAPÍTULO 2

2. ESTADO DEL ARTE

Las siguientes secciones muestra el resultado de una revisión exhaustica de artículos científicos relacionadas a la predicción de la cantidad de tarjetas de crédito emitidas. Por otro lado, el marco teórico tecnológico describe la conceptos, metodología y herramientas utilizadas en el proyecto.

2.1 Estado del arte

La investigación relacionada en modelos de predicción de tarjetas de crédito con aprendizaje automático como la inteligencia artificial, aprendizaje de máquina, minería de datos o aprendizaje profundo se enfoca en diferentes áreas. Se realizó una exploración sobre artículos científicos indexados a scopus, donde se buscó en títulos, resúmenes y palabras claves los siguientes términos:
“Credit Card” and (“Machine Learning” OR “Artificial Intelligence” OR “Data Mining” or “Deep Learning”) and (“Risk” OR “Issuance” OR “Issuer”).

Según los resultados de la búsqueda se ha encontrado los siguientes enfoques principales.

La investigación (Caruso et al., 2021) se centra en la evaluación del riesgo de crédito en el sector bancario, enfatizando la importancia de integrar tanto características cuantitativas como cualitativas de las solicitudes de préstamos. Tradicionalmente, la evaluación del riesgo de crédito se ha basado en indicadores cuantitativos, sin embargo, este estudio propone un enfoque más holístico mediante el uso de algoritmos de agrupamiento para datos mixtos. La correcta evaluación del riesgo de crédito no solo protege a los bancos del incumplimiento, sino que también optimiza el rendimiento de los préstamos concedidos y asegura una asignación más eficiente de los fondos. El estudio considera y aplica tres algoritmos de clustering para datos mixtos a un conjunto de datos real sobre riesgo de crédito en Alemania:

- Método de Huang (1998): Extiende el algoritmo k-means para manejar grandes conjuntos de datos con valores categóricos.

- Método de Ahmad & Dey (2007): Un algoritmo k-means para datos mixtos que combina atributos numéricos y categóricos.
- Método de Cheung & Jia (2013): Un enfoque de clustering basado en una métrica de similitud unificada sin necesidad de conocer el número de clusters a priori.

El autor (Caruso et al., 2021) también comparó entre sí los métodos y también con algunos métodos estándar. La comparación se centró en la precisión de la clasificación y los índices de validación interna. Grupo de buenos clientes: Incluye 771 solicitantes con una edad promedio de 35.33 años, residentes durante 2.82 años. No tienen una cuenta corriente o han tenido una recientemente (en promedio durante 16.43 meses). Tienen un monto de crédito promedio de 2092.19 DM (marco alemán), una tasa de cuota de 3.05 y 1.35 créditos existentes en el banco. Poseen bienes raíces y solicitaron préstamos para comprar radio y televisión. Grupo de malos clientes: Caracterizados por una duración mayor de la cuenta de crédito existente (35.98 meses), un alto monto de crédito (7240.97 DM), no poseen bienes raíces, pero sí un coche, y solicitaron préstamos para comprar un coche usado. Los resultados mostraron que todos los algoritmos de datos mixtos tienen un error de clasificación muy bajo, siendo el método de Ahmad & Dey el que presentó el mejor rendimiento.

El estudio de (Bhatore et al., 2020) realizó una revisión exhaustiva de la literatura de 1032 artículos de investigación y se seleccionaron 136 estudios para la revisión, publicados entre 1993 y marzo de 2019. Los estudios seleccionados se centraron en la evaluación del riesgo crediticio, con un enfoque en técnicas de puntaje crediticio, predicción de activos no productivos (NPA) y detección de fraudes. Las técnicas empleadas incluyen modelos basados en lógica difusa, ontologías, y sistemas expertos. Estos métodos se han utilizado para el puntaje crediticio y la detección de fraudes, destacando los modelos basados en lógica difusa por su popularidad. Se identificaron diversos enfoques de aprendizaje automático (ML), incluidos los modelos probabilísticos, redes neuronales, optimización y modelos de ensamblaje. Los modelos híbridos y de ensamblaje han mostrado ser más efectivos que los clasificadores individuales. La mayoría de los estudios utilizaron técnicas no lineales como redes neuronales y máquinas de soporte vectorial (SVM). Cada modelo presenta sus propios riesgos y desafíos, y no se puede confiar completamente en un

solo clasificador complejo para la evaluación del riesgo crediticio o la detección de fraudes. Las diferencias en las regulaciones y las características de los datos entre diferentes bancos y regiones geográficas afectan significativamente la precisión de los modelos.

La Figura 2.1 muestra los resultados que se obtuvo, los modelos de ensamblaje y los modelos híbridos, combinando redes neuronales y SVM, demostraron ser más adaptativos y efectivos. Sin embargo, la falta de conjuntos de datos públicos completos representa una preocupación significativa para los investigadores. La mejora en la interpretabilidad de los modelos de ensamblaje es un área importante que necesita más exploración.

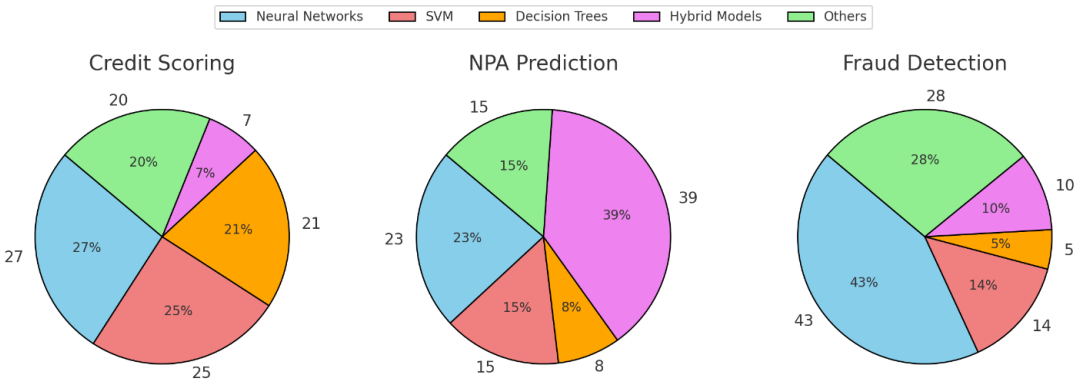


Figura 2.1 Resultados de los Algoritmos De Aprendizaje Automático en Diferentes Áreas (Bhatore et al., 2020).

La minería de datos en la industria de tarjetas de crédito se ha utilizado para el desarrollo del mercado y el mantenimiento de clientes, así como para el control de riesgos. (Li et al., 2010) propone una segmentación de clientes, se basa en datos reales de un banco comercial chino y selecciona variables relevantes para la segmentación de clientes de alta calidad utilizando K-means. Se construyen modelos predictivos para identificar las características de diferentes tipos de clientes y se comparan para seleccionar el más eficiente. Los datos de tarjetas de crédito provienen del centro de tarjetas de crédito del banco comercial chino, incluyendo información personal, familiar y registros de consumo de 72,544 registros. Después de eliminar registros ilógicos y valores faltantes, se utilizaron 65,012 registros para el análisis. Se seleccionaron variables relacionadas con los ingresos, el monto del

consumo y el historial crediticio para el clustering. Los clientes se clasificaron en cuatro tipos mediante K-means:

1. Clientes de alta calidad: Alto monto de consumo mensual, buenos ingresos y buen historial crediticio.
2. Clientes potencialmente de alta calidad: Altos ingresos, pero baja puntuación crediticia.
3. Clientes comunes: Ingresos y consumo moderados, buen historial crediticio.
4. Clientes desfavorables: Bajos ingresos y consumo, mal historial crediticio.

Se construyeron modelos de pronóstico utilizando redes neuronales, C5.0 (Bujlow et al., 2012), Árboles de Clasificación y Regresión (CART) (Steinberg, 2009) y el Detector Automático de Interacciones basado en Chi-cuadrado (CHAID) (Deng et al., 2023). La red neuronal mostró la mayor precisión, pero se descartó por posible sobreajuste y dificultad de interpretación. El modelo C5.0 se seleccionó por su equilibrio entre precisión y explicabilidad, proporcionando reglas “if-then” útiles para identificar características de diferentes tipos de clientes. El estudio demuestra que la tecnología de minería de datos es altamente efectiva en la segmentación de clientes y marketing dirigido en la industria de tarjetas de crédito. Los bancos que utilicen estas tecnologías para convertir datos brutos en información valiosa tendrán una ventaja competitiva significativa.

En el estudio de (Gao et al., 2021) se propone un algoritmo de bosque aleatorio (random forest) incremental que combina el aprendizaje incremental con random forest para abordar la evaluación de riesgo crediticio. El aprendizaje incremental permite que un modelo aprenda continuamente de nuevas muestras sin olvidar el conocimiento adquirido previamente. Esto es similar al aprendizaje humano, donde la información nueva se integra con el conocimiento existente. El algoritmo de random forest incremental se implementa de la siguiente manera: se construye un modelo básico utilizando random forest tradicional con las muestras existentes; las nuevas muestras que llegan con el tiempo se incorporan directamente al modelo incremental; se construye un árbol de decisión de clasificación que admite el aprendizaje incremental, almacenando muestras en los nodos hoja y dividiéndolos cuando la impureza alcanza un umbral predeterminado; y se descartan las muestras con menor contribución para optimizar el espacio y eficiencia del modelo.

Los datos experimentales provienen de un banco en Taiwán e incluyen información básica de los titulares de tarjetas de crédito, líneas de crédito, información de facturación y pagos durante seis meses (de abril a septiembre de 2005), y etiquetas de incumplimiento. Se asumió que las muestras con números de identificación menores o iguales a 5000 llegaban de una vez, mientras que las 25000 muestras restantes llegaban en orden de número de identificación. El experimento comparó el rendimiento del random forest incremental (IRF) con otros seis algoritmos tradicionales: random forest (RF), árbol de decisión (C4.5), regresión logística (Logistic), naive bayes (NaiveBayes), red neuronal BP (BP) y máquina de vectores de soporte (SVM). Los resultados se evaluaron en términos de precisión, exactitud, recall y AUC (Área Bajo la Curva).

El IRF superó a los otros algoritmos en todos los índices de evaluación. Mostró una mayor capacidad para clasificar correctamente a los clientes que incumplen y los que no (precisión y exactitud), tuvo un mejor rendimiento en la recuperación de verdaderos positivos (recall) y tuvo la mayor área bajo la curva AUC, indicando el mejor rendimiento general en la clasificación. Además, el tiempo de modelado del IRF fue más corto en comparación con RF, BP y SVM, mostrando una mayor eficiencia computacional.

El artículo de (Rajamohamed & Manokaran, 2018) presenta un modelo mejorado para predecir la deserción de clientes de tarjetas de crédito utilizando técnicas avanzadas de agrupamiento y aprendizaje supervisado. Combinan el algoritmo modificado de K-means con varios clasificadores supervisados, incluyendo máquinas de soporte vectorial (SVM), bosque aleatorio, árboles de decisión, K-vecinos más cercanos y Naive Bayes. La metodología del estudio se basa en el uso de un conjunto de datos del repositorio UCI que contiene información sobre 30,000 titulares de tarjetas de crédito de un banco en Taiwán. Se aplicaron técnicas de preprocesamiento de datos, como la normalización min-max, para preparar los datos para el análisis. Los autores utilizaron tres algoritmos de clustering: K-means, Fuzzy C-means y Rough K-means, siendo este último el que mostró mejores resultados al manejar la incertidumbre en la asignación de datos a clusters.

En cuanto a los algoritmos de clasificación, se evaluaron cinco principales: KNN, árboles de decisión, bosque aleatorio, SVM y Naive Bayes. Cada uno de estos algoritmos fue combinado con los resultados de clustering para formar modelos híbridos. Los experimentos demostraron que los modelos híbridos, especialmente la combinación de Rough K-means con SVM, superaron significativamente a los clasificadores individuales en términos de precisión y reducción de errores de clasificación. El modelo híbrido Rough K-means combinado con SVM alcanzó una precisión del 96.85%, superando significativamente a los clasificadores individuales. La combinación de técnicas de clustering aproximado con SVM demostró ser la más efectiva en la predicción de la deserción de clientes.

En su trabajo (Hemdan Ezz El-Dinand Manjaiah, 2022), presenta un modelo de aprendizaje profundo para la detección de anomalías en fraudes con tarjetas de crédito y se compara con varios algoritmos de aprendizaje automático existentes. El modelo propuesto se aplica a un conjunto de datos recopilado en Europa durante dos días en septiembre de 2013. Se utiliza la métrica de precisión para evaluar la capacidad del modelo propuesto para detectar fraudes con tarjetas de crédito. El modelo de aprendizaje profundo desarrollado para la detección de fraudes con tarjetas de crédito mostró una mayor precisión en comparación con otros algoritmos de aprendizaje automático como la regresión logística, el análisis discriminante lineal, K-Nearest Neighbors, el árbol de decisión, Naive Bayes gaussiano y random forest como se puede visualizar en la Figura 2.2. La precisión del modelo de aprendizaje profundo fue del 99.97%, ligeramente superior a la de los otros algoritmos. Los resultados destacan la efectividad del modelo de aprendizaje profundo para detectar fraudes en transacciones financieras.

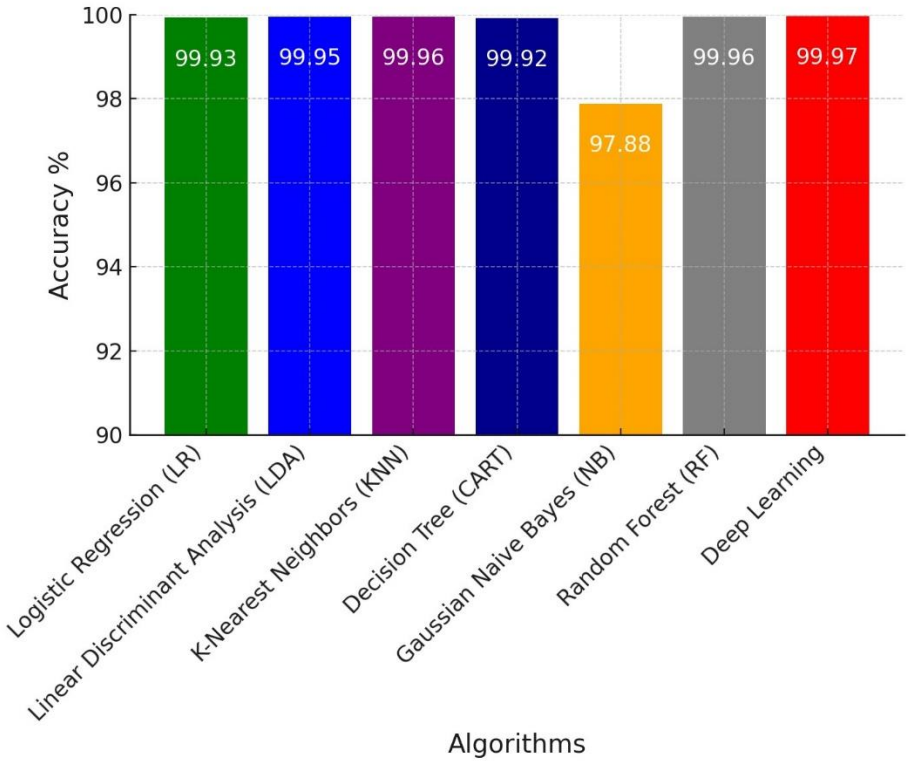


Figura 2.2 Rendimiento de los Algoritmos en Deteccion de Fraudes en Tarjetas de Credito (Hemdan Ezz El-Dinand Manjaiah, 2022).

Las entidades de tarjetas de crédito como obligación deben detectar transacciones fraudulentas para evitar que los clientes sean cobrados por compras no realizadas. La ciencia de datos y el aprendizaje automático son cruciales en esta tarea. El fraude financiero, que incluye el fraude con tarjetas de crédito, causa pérdidas de miles de millones de dólares anualmente. El autor (Singh et al., 2022), propone un sistema de detección de fraudes con tarjetas de crédito que monitoreará el patrón de gasto del usuario y determinará si una transacción es fraudulenta o no. Analiza y compara cuatro algoritmos en función de su precisión, regresión logística, arboles de decisión, bosque aleatorio y CatBoost (boosting de gradiente). Se encontró que el algoritmo Catboost es el más efectivo, con una precisión del 99.87%. El conjunto de datos para la detección de fraudes fue tomado de Kaggle.

De los artículos científicos revisados en los párrafos anteriores, se destaca una línea de investigación predominantemente centrada en la evaluación de riesgos de crédito. Estos estudios abordan tanto características cuantitativas como cualitativas, adoptando enfoques holísticos que integran diversos tipos de datos. En particular, la

evaluación de riesgos de crédito se ha beneficiado significativamente del uso de algoritmos de agrupamiento para datos mixtos y otros métodos avanzados de agrupamiento. Además, se menciona la eficacia de los modelos híbridos y de ensamblaje, que combinan redes neuronales y máquinas de soporte vectorial (SVM). Estos modelos híbridos han demostrado ser más adaptativos y precisos que los clasificadores individuales, ofreciendo mejoras sustanciales en la precisión y reducción de errores en la clasificación. La combinación de diferentes técnicas de aprendizaje automático permite abordar la complejidad inherente de los datos financieros y mejorar la capacidad predictiva de los modelos.

Para la segmentación de clientes, la minería de datos juega un papel crucial, siendo el algoritmo K-means una de las herramientas más comúnmente empleadas. Este método permite clasificar a los clientes en diferentes segmentos basados en características como el comportamiento de gasto, los ingresos y el historial crediticio. La segmentación precisa de clientes es esencial para el desarrollo de estrategias de marketing dirigidas y la mejora de la satisfacción del cliente. En cuanto a la detección de fraudes, se utilizan diversos algoritmos de aprendizaje automático, incluidos K-Nearest Neighbors (KNN), árboles de decisión, Naive Bayes gaussiano, random forest, CatBoost y algoritmos de aprendizaje profundo. Estos algoritmos son evaluados en términos de precisión, exactitud y capacidad para identificar transacciones fraudulentas. Los estudios han demostrado que los algoritmos de aprendizaje profundo, en particular, ofrecen una mayor precisión en la detección de fraudes, superando a los métodos tradicionales.

En general, existen numerosas investigaciones centradas en el uso de tarjetas de crédito, lo cual destaca la importancia de la exploración de este tema a través de la ciencia de datos. Sin embargo, hay una notable brecha en la investigación: no se encuentran estudios primarios que se enfoquen en brindar una solución específica para la predicción de la cantidad de tarjetas de crédito emitidas por entidades financieras, considerando sus diferentes características.

Este vacío en la literatura académica resalta la necesidad de desarrollar modelos predictivos con aprendizaje automático que puedan abordar este aspecto crucial. La capacidad de predecir con precisión la cantidad de tarjetas de crédito a emitir, no solo ayudaría a las instituciones financieras a optimizar sus estrategias de marketing

y gestión de riesgos, sino que también mejoraría la eficiencia en la asignación de recursos y la planificación operativa.

2.2 Marco teórico y técnico

La predicción de la cantidad de tarjetas de crédito que emite una entidad financiera se puede resolver mediante técnicas de regresión. Es decir, empleando algoritmos de aprendizaje automático con series de tiempo, la serie de tiempo se compone de observaciones históricas del número de tarjetas emitidas, registradas en intervalos temporales regulares.

La regresión en series de tiempo implica modelar la relación entre las variables temporales (tiempo) y la variable objetivo (cantidad de tarjetas emitidas). A diferencia de una regresión simple, que podría considerar las características estáticas, la regresión en series de tiempo incorpora la dependencia temporal, considerando que los valores pasados influyen directamente en los valores futuros.
En las siguientes secciones, se describe la metodología y algoritmos de aprendizaje automático, y conceptos relacionados a series de tiempo.

2.2.1 Pipeline de Ciencia de Datos con Aprendizaje Automático Automatizado

El pipeline tradicional requiere diversas etapas, que van desde la adquisición y limpieza de datos hasta la implementación de modelos predictivos. Sin embargo, el desarrollo de AutoML (Automated Machine Learning) ha permitido la automatización de varias de estas etapas, lo que facilita y acelera el proceso, haciéndolo más accesible para usuarios con diferentes niveles de experiencia. Para este caso se utilizará AutoML para la selección del modelo, optimización de hiperparámetros y la validación del modelo, para los otros pasos se utiliza los pasos tradicionales.

1. Adquisición y Limpieza de Datos. Los datos en bruto, provenientes de diversas fuentes, suelen estar incompletos, desordenados o contener ruido (valores atípicos, datos faltantes, etc.). Esta etapa es crítica, ya que la calidad de los datos influye directamente en la calidad del modelo predictivo. La limpieza de datos incluye la eliminación de duplicados, imputación de valores faltantes y corrección de errores en el conjunto de datos.
2. Preprocesamiento y Selección de Características. Una vez los datos están limpios, es necesario preprocesarlos para que sean aptos para el

entrenamiento de modelos. Esto puede incluir la transformación de variables categóricas en numéricas, el escalado de características o la normalización. Posteriormente, se realiza una selección de características para eliminar variables irrelevantes o redundantes y mejorar la eficiencia del modelo.

3. Construcción de Características. En esta etapa se pueden generar nuevas características derivadas de las existentes. Por ejemplo, en un conjunto de datos de series temporales, se podrían construir variables como "tendencia" o "estacionalidad". La ingeniería de características puede mejorar el rendimiento del modelo al proporcionar más información útil.
4. Selección de Modelos (AutoML): Una vez que los datos están listos, se procede a seleccionar el modelo de machine learning que se va a entrenar. Los modelos disponibles pueden ser de naturaleza supervisada (como la regresión logística, árboles de decisión, redes neuronales, etc.) o no supervisada (como clustering o detección de anomalías).
5. Optimización de Hiperparámetros (AutoML): Cada modelo tiene una serie de hiperparámetros que controlan su comportamiento. La optimización de hiperparámetros implica buscar los valores óptimos para estos parámetros, generalmente mediante técnicas como grid search, random search o optimización bayesiana, con el fin de mejorar el rendimiento del modelo.
6. Validación del Modelo (AutoML): El modelo final debe ser validado para garantizar su capacidad de generalización en datos no vistos. Para ello, se utilizan técnicas como la validación cruzada, donde el conjunto de datos se divide en múltiples partes y el modelo se entrena y prueba varias veces para evaluar su desempeño.

2.2.2 Modelos de regresión

Los modelos de regresión son herramientas esenciales para analizar relaciones entre variables y realizar predicciones. A continuación, se presentan las definiciones de los principales tipos de regresión:

- Regresión Lineal Simple: Es un modelo estadístico que describe la relación entre una variable dependiente (o respuesta) y una única variable independiente (o predictor) utilizando una línea recta (Rodríguez del Águila & Benítez-Parejo, 2011).

- Regresión Lineal Múltiple**:** Es una extensión de la regresión lineal simple que analiza la relación entre una variable dependiente y dos o más variables independientes (Rodríguez del Águila & Benítez-Parejo, 2011).
- Regresión en Series de Tiempo: es la técnica que analiza y modela datos dependientes del tiempo para identificar tendencias y patrones dominantes en un conjunto de datos secuenciales. Se aplica principalmente para descubrir cambios importantes o inusuales dentro de un contexto dinámico y para realizar predicciones sobre el comportamiento futuro de los datos con base en información pasada (Chen et al., 2002).

2.2.3 Series de tiempo

Las series de tiempo son técnicas de predicción que se utilizan ampliamente en diversas industrias para modelar y estimar datos futuros analizando patrones históricos significativos. Ayudan a tomar decisiones mejor informadas al identificar tendencias y señales tempranas en los datos históricos (Wu et al., 2024).

Según (Wu et al., 2024) las series de tiempo tiene los principales componentes:

- Estacionalidad: La estacionalidad en las series de tiempo representa patrones repetitivos o cíclicos que ocurren a intervalos regulares debido a factores como estaciones del año o eventos específicos. Por ejemplo, el procedimiento de descomposición de estacionalidad-tendencia (STL) se utiliza para analizar y aislar los componentes estacionales y de tendencia en una serie.
- Tendencia: La tendencia refleja el comportamiento subyacente de los datos a lo largo del tiempo, identificando aumentos, disminuciones o estabilidad general en los niveles de una serie temporal. Es uno de los componentes principales extraídos al aplicar descomposiciones como STL.
- Ruido: El ruido incluye las fluctuaciones aleatorias o irregulares que no pueden ser explicadas por los componentes de tendencia o estacionalidad. Representa la variabilidad no sistemática y los factores impredecibles que afectan la serie temporal.

2.2.4 Métricas de rendimiento de modelos de serie de tiempo.

A continuación, se presentan las métricas clave utilizadas para evaluar el desempeño de modelos en series de tiempo y sus definiciones (Quispe et al., 2024):

- Error Absoluto Medio Escalado (MASE): El MASE es una medida de precisión de las predicciones que escala el error absoluto promedio utilizando el error absoluto promedio de un modelo de referencia (como un modelo naïve, que utiliza el valor del periodo anterior). Es especialmente útil para comparar la precisión de las predicciones entre conjuntos de datos con diferentes escalas.

$$\text{MASE} = \frac{\frac{1}{n}\sum_{t=1}^{n}|y_t - \widehat{y_t}|}{\frac{1}{n-1}\sum_{t=2}^{n}|y_t - y_{t-1}|}$$

  Donde $y_t$es el valor observado $\widehat{y_t}$ es el valor predicho, y n es el número de observaciones.

- Raíz del Error Cuadrático Medio Escalado (RMSSE): El RMSSE es similar al MASE, pero utiliza errores cuadrados en lugar de errores absolutos. Penaliza más los errores grandes y es útil cuando estos son críticos.

$$\text{RMSSE} = \sqrt{\frac{\frac{1}{n}\sum_{t=1}^{n}(y_t - \widehat{y_t})^2}{\frac{1}{n-1}\sum_{t=2}^{n}(y_t - y_{t-1})^2}}$$

  Donde $y_t$es el valor observado $\widehat{y_t}$ es el valor predicho, y n es el número de observaciones.

- Error Absoluto Medio (MAE): El MAE mide el promedio de la magnitud de los errores en un conjunto de predicciones, sin considerar la dirección de los errores. Representa el promedio de las diferencias absolutas entre los valores observados y los predichos.

$$\text{MAE} = \frac{1}{n}\sum_{t=1}^{n}|y_t - \widehat{y_t}|$$

  Donde $y_t$es el valor observado $\widehat{y_t}$ es el valor predicho, y n es el número de observaciones.

- Raíz del Error Cuadrático Medio (RMSE): El RMSE calcula la raíz cuadrada del promedio de las diferencias cuadráticas entre los valores observados y los predichos. Es más sensible a errores grandes en comparación con el MAE.

$$\text{RMSE} = \sqrt{\frac{1}{n}\sum_{t=1}^{n}(y_t - \widehat{y_t})^2}$$

Donde y_t es el valor observado $\hat{y}_t$ es el valor predicho, y n es el número de observaciones.

- Error Porcentual Absoluto Medio (MAPE): El MAPE expresa el error de las predicciones como un porcentaje de los valores observados, lo que lo hace independiente de la escala. Sin embargo, puede ser sesgado cuando los valores reales son muy pequeños.

$$\text{MAPE} = \frac{1}{n}\sum_{t=1}^{n}\left|\frac{y_t - \hat{y}_t}{y_t}\right| \times 100\%$$

Donde y_t es el valor observado $\hat{y}_t$ es el valor predicho, y n es el número de observaciones.

- Error Porcentual Absoluto Medio Simétrico (SMAPE): El SMAPE es una alternativa al MAPE que trata de manera simétrica los errores de sobrepredicción y subpredicción. Es más robusto frente a valores observados pequeños.

$$\text{SMAPE} = \frac{1}{n}\sum_{i=1}^{n}\frac{|y_i - \hat{y}_i|}{\frac{|y_i| + |\hat{y}_i|}{2}} \times 100\%$$

Donde y_t es el valor observado $\hat{y}_t$ es el valor predicho, y n es el número de observaciones.

- Coeficiente de Determinación (R^2): El R^2 evalúa qué tan bien el modelo explica la varianza en los datos observados. Un valor cercano a 1 indica un mejor desempeño del modelo.

$$R^2 = 1 - \frac{\sum_{t=1}^{n}(y_t - \hat{y}_t)^2}{\sum_{t=1}^{n}(y_t - \bar{y})^2}$$

Donde y_t es el valor observado $\hat{y}_t$ es el valor predicho, y n es el número de observaciones.

- Tiempo de Cómputo (TT): El Tiempo de Cómputo mide la duración requerida por un modelo para entrenarse y generar predicciones. Es una métrica relevante en aplicaciones donde la eficiencia temporal es esencial.
- Medición: Se reporta típicamente en segundos, utilizando funciones del sistema para registrar la duración del proceso de predicción.

2.2.5 Sesgo en Inteligencia Artificial

Este tipo de sesgo se presenta como el riesgo de introducir o perpetuar desigualdades a través de algoritmos de aprendizaje automático en aplicaciones

financieras y otras áreas críticas. En particular, este sesgo hace énfasis en cómo los algoritmos de IA, que, al basarse en datos históricos o diseños algorítmicos que no consideran adecuadamente la equidad, pueden reproducir patrones de discriminación preexistentes en la sociedad (Song et al., 2023).

El sesgo en los datos ocurre cuando los datos utilizados para entrenar un modelo no son representativos del mundo real. Esto incluye el sesgo de selección, donde las muestras no reflejan adecuadamente la población objetivo, el sesgo de reporte, que enfatiza patrones recurrentes excluyendo diversidad, y el sesgo cognitivo, introducido por prejuicios humanos en la recolección de datos. Por otro lado, el sesgo algorítmico se produce durante el diseño o implementación del modelo. Este tipo de sesgo puede surgir de la selección de algoritmos inapropiados o el uso de funciones objetivo que priorizan ciertos resultados sobre otros, introduciendo desigualdades en las predicciones. Finalmente, el sesgo del ingeniero proviene de decisiones tomadas por los desarrolladores o evaluadores de los modelos, quienes pueden proyectar sus propios prejuicios, como el sesgo de confirmación, al buscar resultados que validen sus hipótesis iniciales. Estos tipos de sesgo son interdependientes y pueden influir en todas las etapas del desarrollo de sistemas de inteligencia artificial (Mavrogiorgos et al., 2024).

Según los autores (Mehrabi et al., 2021), definen los siguientes tipos de sesgos presentes en la inteligencia artificial (IA), los cuales pueden surgir en diferentes etapas del desarrollo de sistemas basados en aprendizaje automático:

Sesgo de Medición: Este sesgo ocurre cuando las variables empleadas como proxy no representan de manera precisa el concepto subyacente que se pretende medir. Por ejemplo, el uso de arrestos previos como indicador de "riesgo" en sistemas como COMPAS genera resultados que afectan desproporcionadamente a comunidades minoritarias, reflejando prejuicios en la estructura de los datos utilizados.

Sesgo de Variables Omitidas: Surge cuando se excluyen variables significativas del modelo, lo que resulta en interpretaciones y decisiones erróneas. Este tipo de sesgo es especialmente problemático en contextos donde las variables omitidas tienen una

relación causal con el resultado de interés. Un ejemplo puede ser ignorar la aparición de nuevos actores en el mercado al analizar la retención de clientes.

Sesgo de Representación: Ocurre cuando los datos de entrenamiento no son representativos de la población objetivo, generando modelos que no generalizan bien. Un ejemplo común es la falta de diversidad en conjuntos de datos como ImageNet, que puede llevar a resultados sesgados al procesar imágenes de ciertas poblaciones geográficas o culturales.

Sesgo de Agregación: Este sesgo se presenta cuando se ignoran las diferencias individuales dentro de subgrupos y se realizan generalizaciones a toda la población. Por ejemplo, las herramientas de evaluación de riesgos utilizadas en el sistema judicial, como COMPAS, han demostrado tener sesgos raciales significativos, etiquetando erróneamente a personas negras como de alto riesgo de reincidencia con mayor frecuencia que a personas blancas. Este problema se debe a que los algoritmos reflejan desigualdades históricas en los datos utilizados para su entrenamiento, perpetuando la discriminación en lugar de mitigarla (Angwin et al., 2016).

Sesgo de Muestreo: Aparece cuando las muestras utilizadas no son seleccionadas aleatoriamente, resultando en conjuntos de datos que no reflejan la población objetivo. Esto puede limitar la capacidad del modelo para generalizar y puede exacerbar desigualdades preexistentes.

Sesgo Histórico: Este sesgo refleja desigualdades existentes en la sociedad que se perpetúan a través de los datos históricos utilizados en los sistemas de IA. Por ejemplo, si los datos de salarios reflejan disparidades de género, el modelo puede aprender y reforzar estas desigualdades.

Sesgo Temporal: Relacionado con cambios en los comportamientos o características de las poblaciones a lo largo del tiempo. Este sesgo puede impactar la capacidad del modelo para mantenerse relevante en entornos dinámicos donde los datos evolucionan constantemente.

2.3 Plataformas y herramientas

Es la siguiente sección se describe las principales herramientas tecnológicas empleadas para el desarrollo del modelo de predicción.

2.3.1 Lenguaje de programación Python

El lenguaje de programación Python es ampliamente utilizado en ciencia de datos por su versatilidad y su rica comunidad de bibliotecas enfocadas en el análisis de datos, machine learning y automatización. Python será el principal lenguaje de desarrollo empleado en este estudio.

2.3.2 Jupyter Notebook

Jupyter Notebook proporciona un entorno interactivo para el desarrollo de scripts en Python. Su capacidad para integrar código, texto y visualizaciones lo hace ideal para la experimentación, análisis y documentación del flujo de trabajo de ciencia de datos.

2.3.3 Librería Pycaret

PyCaret es una biblioteca de bajo código que facilita la automatización de las tareas más comunes de machine learning, como el preprocesamiento de datos, la selección de modelos y la comparación de su rendimiento. Esta herramienta es clave para automatizar y acelerar el proceso de desarrollo de modelos predictivos (Ali, 2020). Los principales métodos de la librería se describen en la Tabla 2.1.

Tabla 2.1 Métodos de Librería de Pycaret (Elaboración Propia).

Método	Descripción	Parámetros importantes
setup()	Inicializa el entorno de trabajo para el análisis y modelado.	data, target, session_id, fold_strategy, n_jobs
compare_models()	Compara varios modelos y selecciona el mejor según una métrica.	n_select, sort, include, exclude, fold, verbose
create_model()	Entrena un modelo específico en base al algoritmo seleccionado.	estimator, fold, round, fit_kwargs
tune_model()	Optimiza los hiperparámetros del modelo entrenado.	model, n_iter, optimize, custom_grid, search_library
plot_model()	Genera gráficos de diagnóstico y evaluación del modelo.	model, plot, save, display_format
predict_model()	Realiza predicciones utilizando el modelo entrenado.	model, data, raw_score
finalize_model()	Entrena el modelo final utilizando todos los datos disponibles.	model
save_model()	Guarda el modelo entrenado en disco.	model, model_name, verbose
load_model()	Carga un modelo previamente guardado en disco.	model_name

A continuación, en la Tabla 2.2, se presentan los modelos implementados en la librería PyCaret, específicamente diseñados para abordar el análisis y predicción de series de tiempo.

Tabla 2.2 Modelos de Machine Learning en Pycaret (Elaboración Propia).

Modelo	Descripción
arima	Autoregressive Integrated Moving Average: Modelo estadístico clásico para series de tiempo.
sarima	Seasonal ARIMA: Extensión de ARIMA que incluye estacionalidad.
sarimax	Seasonal ARIMA with Exogenous Variables: ARIMA estacional con variables externas.
ets	Exponential Smoothing State Space Model: Modelo basado en alisado exponencial.
theta	Theta Model: Modelo basado en el alisado exponencial para series de tiempo.
exponential	Exponential Smoothing: Modelo de alisado exponencial simple.
croston	Croston's Method: Método especializado para demandas intermitentes.
lr_cds_dt	Linear Regression with Calendar Date Splitting: Regresión lineal con división de fechas.
lasso_cds_dt	LASSO Regression with Calendar Date Splitting: Regresión LASSO con división de fechas.
ridge_cds_dt	Ridge Regression with Calendar Date Splitting: Regresión Ridge con división de fechas.
rf_cds_dt	Random Forest with Calendar Date Splitting: Random Forest con división de fechas.
et_cds_dt	Extra Trees Regressor with Calendar Date Splitting: Modelo Extra Trees con división de fechas.
xgboost_cds_dt	Extreme Gradient Boosting with Calendar Date Splitting: Modelo XGBoost con división de fechas.
lightgbm_cds_dt	LightGBM with Calendar Date Splitting: Modelo LightGBM con división de fechas.
catboost_cds_dt	CatBoost Regressor with Calendar Date Splitting: Modelo CatBoost con división de fechas.
tcn	Temporal Convolutional Network: Modelo basado en redes convolucionales temporales.
lstm	Long Short-Term Memory Networks: Redes neuronales recurrentes para series de tiempo.
nbeats	Neural Basis Expansion Analysis for Time Series: Modelo avanzado basado en redes neuronales.
prophet	Facebook Prophet: Modelo híbrido que descompone tendencia y estacionalidad.
prophet_damp	Damped Prophet: Versión amortiguada del modelo Prophet.
naive	Naive Forecasting: Utiliza el último valor observado como predicción.
seasonal_naive	Seasonal Naive Forecasting: Modelo naïve que considera estacionalidad.
ensemble	Ensemble Model: Combina predicciones de varios modelos.

auto_arima	Automatically selects the best ARIMA model: Selección automática del mejor ARIMA.
tbats	Trigonometric, Box-Cox, ARMA Errors, Trend, and Seasonal Components.
bats	Box-Cox, ARMA Errors, Trend, and Seasonal Components.

CAPÍTULO 3

3. Diseño e implementación

En este capítulo se presenta el procesamiento y análisis exploratorio del conjunto de datos. Además, se detalla el diseño utilizado para la creación de modelos de series de tiempo y el sistema de predicción propuesto, incluyendo una comparación exhaustiva de varios modelos con sus métricas de rendimiento.

3.1 Preparación del conjunto de datos

El conjunto de datos que se utiliza son datos públicos, proporcionados por la Superintendencia de Bancos del Ecuador, se utiliza la información de Estadísticas de Consumos con Tarjetas (crédito, débito, prepago) (A12), se cuenta con información de los años 2021 ,2022, 2023 y agosto – 2024 (Superintendencia de Bancos de Ecuador, 2024). En esta estructura se reportará el detalle de los consumos realizados con cada una de las tarjetas de crédito, débito y prepago recargables que mantenga un tarjetahabiente en la entidad financiera. La información está disponible en formato Excel (.xlsx) que se descarga directamente desde la página web, se lo almacena en un directorio local que el proyecto tiene configurado para procesar los datos.

3.1.1 Proceso de limpieza

Es importante realizar una adecuada limpieza y validación del conjunto de datos, porque datos incompletos o inconsistentes pueden llevar a resultados erróneos, interpretaciones incorrectas y modelos ineficientes.

1. Validación y corrección de valores en blanco, nulos y vacíos, mediante conteos agrupados se identifica dichos valores y se procede a eliminar para el análisis.
2. Transformación del tipo de datos de cada columna al correcto, como las fecha y la columna “cantidad” que indican el número de tarjetas emitidas por entidad.
3. El análisis de valores atípicos mediante el rango intercuartil (IQR) agrupado por cada entidad financiera para la variable "cantidad", no reveló la presencia de valores fuera de los límites, lo que garantiza que el análisis no se verá influenciado por valores extremos.

4. Para asegurar la coherencia en el análisis, se ha llevado a cabo la homologación de categorías. En este proceso, se identificó que una misma entidad financiera aparecía con nombres diferentes a lo largo de los años, lo que podría generar inconsistencias en el análisis. Para corregir esto, los nombres de las entidades fueron agrupados manualmente y estandarizados bajo una denominación uniforme para todo el conjunto de datos. Esta unificación permite un análisis más preciso y evita duplicidades que podrían distorsionar los resultados.

3.1.2 Preprocesamiento y selección de características

El preprocesamiento implicó, la conversión de variables categóricas en variables numéricas para que puedan ser utilizadas eficazmente. Un enfoque común para lograr esta conversión es mediante One Hot Encoding. Por otro lado, la variable objetivo es la columna “Total” que se procede aplicar una normalización. En este proceso se aplica una técnica de normalización a la columna Total del conjunto de datos utilizando el método MinMaxScaler de la librería scikit-learn. El objetivo de esta técnica es transformar los valores originales de la columna Total a un rango específico, entre 0 y 1, manteniendo las proporciones relativas de los datos.

La Figura 3.1, muestra la matriz de correlación, la cual revela relaciones clave entre diversas variables del mercado de tarjetas de crédito en Ecuador entre 2021 y 2024. En términos de emisores, Diners Club y Banco del Pacífico muestran una correlación positiva con segmentos de mercado de mayor nivel, como AA+ y C+, lo que sugiere su enfoque en captar clientes con perfil crediticio intermedio-alto. Además, ciertos meses, como noviembre y diciembre, presentan correlaciones positivas con el aumento de la emisión de tarjetas, indicando una estacionalidad importante en la demanda de crédito, probablemente vinculada a la temporada de consumo de fin de año.

También se observa que las marcas de tarjetas como Visa y Mastercard tienen correlaciones positivas con una mayor cantidad de emisores, lo cual puede interpretarse como una estrategia de diversificación y un alto nivel de aceptación en el mercado ecuatoriano. Por otro lado, marcas menos comunes presentan correlaciones más débiles, indicando una menor penetración de mercado. Esta

información permite a las instituciones financieras ajustar sus estrategias de marketing y emisión de tarjetas, enfocándose en períodos de alta demanda y en los segmentos de clientes que muestran mayor receptividad a las campañas de emisión de crédito.

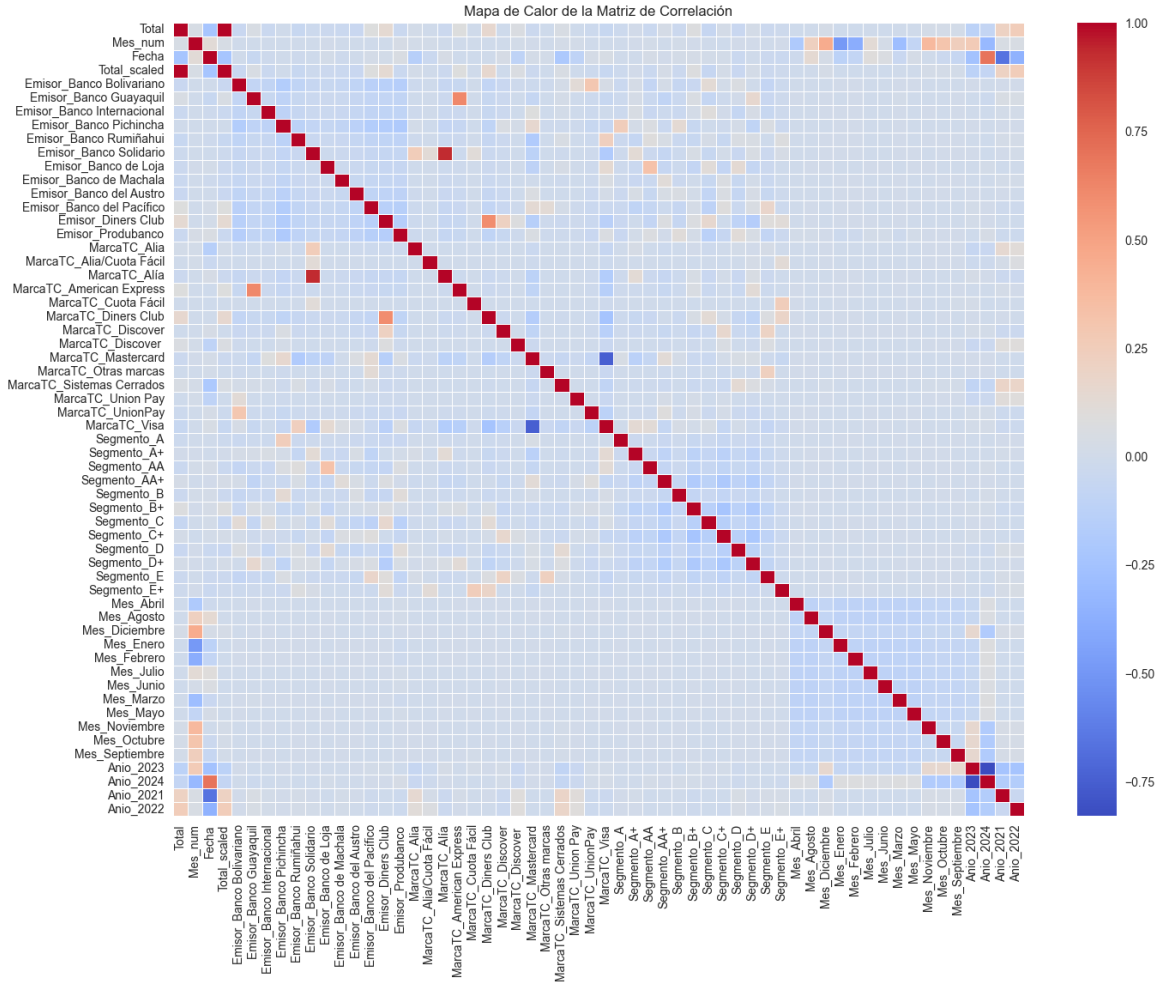


Figura 3.1 Matriz de Correlación de Variables del Conjunto de Datos (Elaboración Propia).

3.1.3 Análisis exploratorio de los datos

El la Figura 3.2 muestra la evolución mensual del total de tarjetas de crédito emitidas por las cinco principales entidades financieras en Ecuador durante los años 2021 a 2024. Cada subgráfico corresponde a un año específico, destacando el rendimiento y posición en el mercado de las instituciones Banco Guayaquil, Produbanco, Banco Pichincha, Banco del Pacífico y Diners Club. En el eje horizontal se representan los meses del año, mientras que el eje vertical muestra el total de tarjetas de crédito en

miles, permitiendo una comparación clara entre las instituciones y la observación de patrones de crecimiento o estacionalidades a lo largo del período evaluado.

Durante los años 2021 y 2022, Diners Club mantiene un claro liderazgo en la emisión de tarjetas de crédito, superando consistentemente a las demás entidades con un volumen que alcanza los 500k hacia el final de cada año. Este comportamiento sugiere una estrategia de mercado robusta y una base de clientes leal que respalda su posición. Banco Pichincha y Banco del Pacífico ocupan el segundo y tercer lugar, respectivamente, con volúmenes de emisión notablemente menores pero estables y con un crecimiento gradual. En cambio, Produbanco y Banco Guayaquil muestran volúmenes más bajos en comparación con las entidades líderes, aunque con una tendencia de crecimiento moderada hacia el final de cada año, lo que podría indicar un impulso estacional o campañas específicas de captación de clientes en esos meses.

En el año 2023, se observa que Diners Club continúa con su liderazgo en el mercado de tarjetas de crédito, aunque con un crecimiento menos pronunciado que en los años anteriores, lo cual podría señalar una estabilización de su base de clientes o una saturación en su mercado objetivo. Tanto Banco Pichincha como Banco del Pacífico mantienen sus posiciones con un incremento estable y gradual en sus totales de emisión. Produbanco y Banco Guayaquil continúan en un crecimiento constante, aunque con fluctuaciones en sus totales acumulados que pueden responder a campañas promocionales de corto plazo.

Para el año 2024 (hasta el mes de agosto), se observa un cambio significativo en la dinámica competitiva. Banco Guayaquil emerge como líder en términos de totales acumulados, incluso superando a Diners Club en algunos meses. Este cambio puede indicar estrategias de expansión más agresivas o una preferencia creciente de los consumidores por los servicios de Banco Guayaquil. Produbanco también muestra un incremento sostenido en su volumen de tarjetas emitidas, lo que sugiere un fortalecimiento en su posición de mercado y una mayor captación de clientes. En tanto, Banco Pichincha y Banco del Pacífico mantienen sus volúmenes en torno a los 300k, con una tendencia estable y variaciones mínimas.

A lo largo de los cuatro años, se identifican algunos patrones estacionales en instituciones como Diners Club, Banco Guayaquil y Produbanco, que tienden a incrementar el número de tarjetas emitidas hacia los últimos meses del año, entre octubre y diciembre. Este patrón probablemente se relacione con festividades o temporadas de alta actividad económica, como Navidad y fin de año, cuando aumenta la demanda de crédito y las instituciones financieras intensifican sus campañas comerciales.

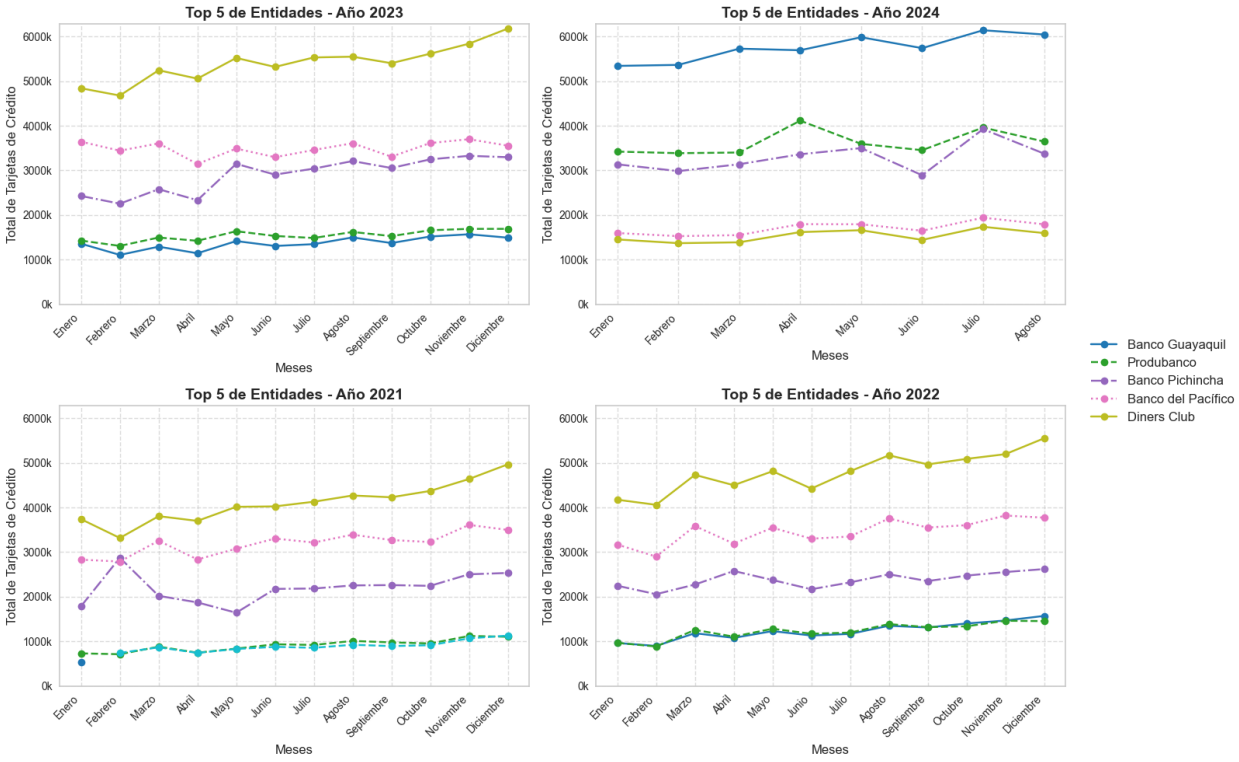


Figura 3.2. Evolución Mensual del Top 5 de Entidades Financieras por Cantidad de Tarjetas De Crédito Emitidas (Elaboración Propia).

La Figura 3.3 ilustra la cantidad total de tarjetas de crédito emitidas en Ecuador, diferenciadas según los segmentos de mercado. En el eje horizontal se presentan los distintos segmentos, mientras que el eje vertical muestra la cantidad total de tarjetas emitidas en miles. Los colores en cada barra representan los distintos años: 2021 (verde), 2022 (rojo), 2023 (azul) y 2024 (naranja), lo que permite comparar las emisiones de tarjetas a través de los años en cada segmento.

Los segmentos AA+ y C+ concentran el mayor número de tarjetas emitidas durante el período analizado, destacándose como los sectores más grandes y activos en el mercado de crédito ecuatoriano. Estos segmentos, que suelen corresponder a clientes con perfiles de crédito intermedios a altos, muestran una alta demanda de tarjetas de crédito, lo cual es consistente a lo largo de los años. El crecimiento en estos segmentos sugiere que las instituciones financieras están enfocando sus esfuerzos en captar clientes de riesgo moderado a alto, probablemente debido a la rentabilidad o la estabilidad de este grupo.

Al observar la evolución anual, se destaca que el año 2023, representado en azul, muestra un incremento notable en la emisión de tarjetas en varios segmentos, particularmente en AA+ y C+. Este pico de emisión en 2023 puede indicar un período de expansión o políticas de captación más agresivas para esos perfiles de clientes. En cambio, en 2024, representado en naranja, se nota un ligero descenso en ciertos segmentos clave, como AA+ y D+, lo que podría señalar una consolidación en la estrategia de emisión de crédito, o un ajuste en las políticas de riesgo de las entidades financieras.

Por otro lado, los segmentos menores, como A, B, D y E, presentan un volumen de tarjetas emitidas considerablemente bajo, con pocos cambios entre los años. Esto sugiere que las instituciones financieras tienen menos actividad o preferencia en estos segmentos, lo que puede estar relacionado con políticas de riesgo más restrictivas o un menor atractivo comercial debido al perfil crediticio de los clientes en estos niveles. En consecuencia, la emisión de tarjetas en estos segmentos permanece limitada y estable a lo largo del tiempo.

En conclusión, el análisis sugiere que los segmentos de crédito intermedio y alto, como AA+ y C+, son de gran importancia estratégica para las instituciones financieras, ya que representan la mayor parte de las emisiones de tarjetas de crédito. Las variaciones en los patrones de emisión reflejan decisiones estratégicas anuales, con un enfoque más expansivo en 2023 y una moderación en 2024. Estos cambios probablemente responden a factores económicos y a ajustes en la política de riesgo de las instituciones, que buscan equilibrar la captación de clientes con la gestión de exposición al riesgo en cada segmento del mercado.

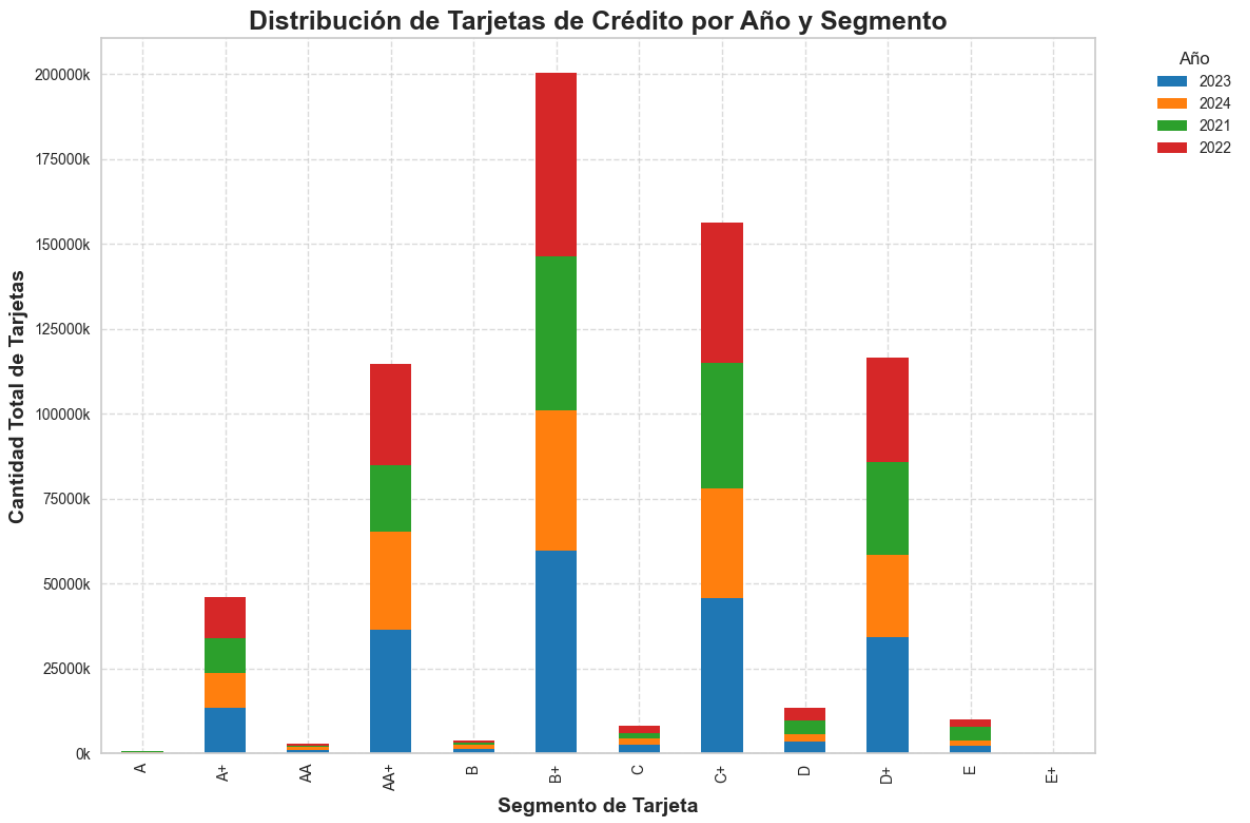


Figura 3.3. Distribución de las Tarjetas Activas Entre los Diferentes Segmentos por Año (Elaboración Propia).

La Figura 3.4 presenta el volumen total de tarjetas de crédito emitidas por distintas entidades financieras en Ecuador. En el eje vertical se representa el total de tarjetas de crédito (en miles), mientras que en el eje horizontal se listan las entidades financieras que participan en el mercado de emisión de tarjetas de crédito en el país.

También la Figura 3.4 muestra que Diners Club es el líder en el mercado de tarjetas de crédito, con un total que supera los 200,000k, posicionándose muy por encima de sus competidores. Le siguen Banco del Pacífico y Banco Pichincha, aunque con un volumen considerablemente menor. Estos bancos, aunque distantes de Diners Club, destacan como los principales competidores, cada uno con más de 100,000k en emisión de tarjetas, lo que sugiere una importante participación en el mercado de crédito.

Banco Guayaquil y Produbanco ocupan el cuarto y quinto lugar, respectivamente, con totales que siguen disminuyendo gradualmente en comparación con los líderes. A partir de este punto, las entidades financieras muestran una caída significativa en la emisión de tarjetas, con instituciones como Banco Bolivariano y Banco del Austro que participan de manera más modesta en el mercado.

En la parte baja del gráfico, bancos como Banco Internacional, Banco Solidario, Banco Rumiñahui, y otros tienen una representación mínima en comparación con los líderes del mercado, indicando una emisión de tarjetas mucho menor y, posiblemente, una participación más enfocada en nichos específicos o clientes de perfil particular.

Este análisis de la distribución del mercado de tarjetas de crédito revela una fuerte concentración en pocas entidades, con Diners Club dominando el sector, seguido de algunos competidores relevantes. Esta distribución sugiere una alta concentración del mercado en las principales entidades, lo que podría implicar una mayor competitividad en estrategias de captación de clientes entre los líderes del mercado y una segmentación clara entre los bancos en cuanto a su alcance y enfoque en la emisión de crédito.

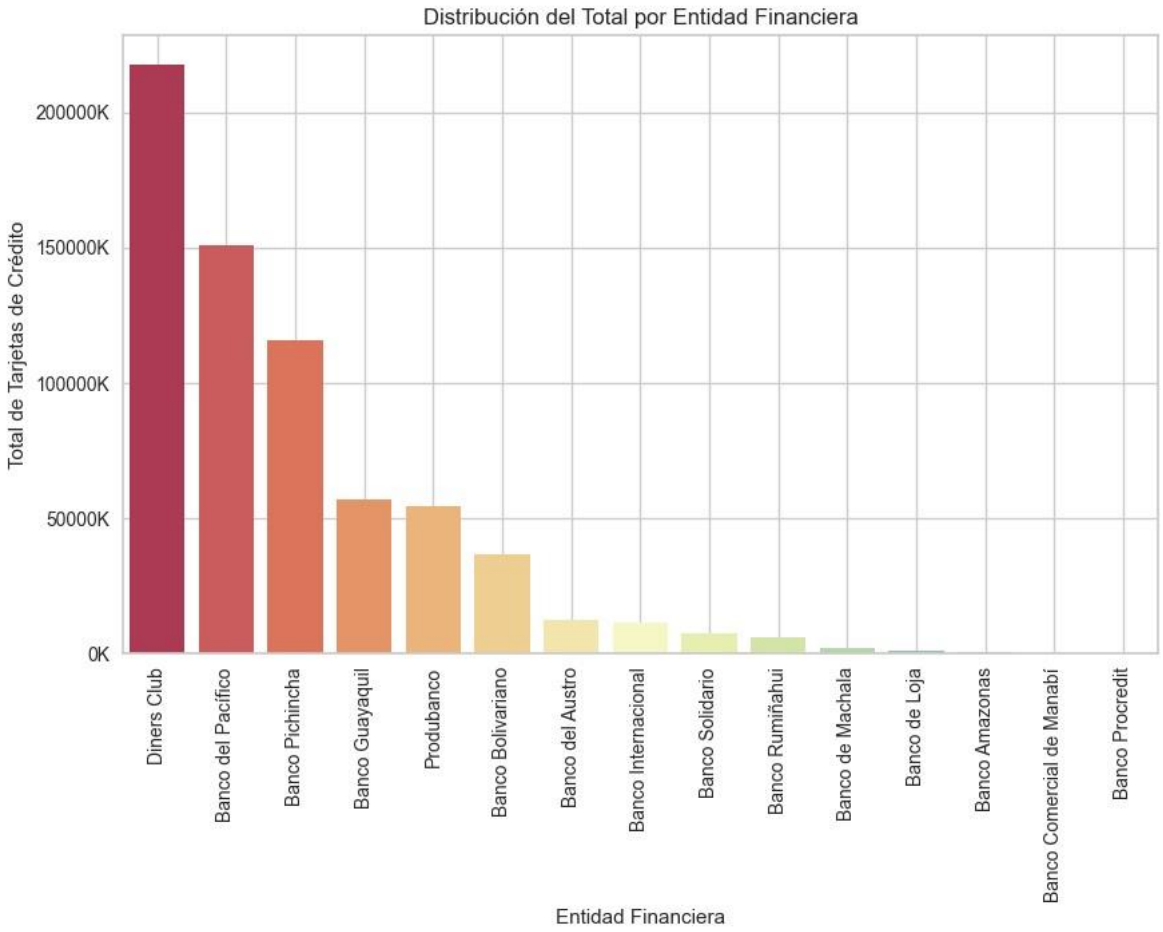


Figura 3.4. Distribución del Total de Tarjetas De Crédito Activas por Entidad Financiera (Elaboración Propia).

La Figura 3.5 de Sankey representa la distribución del mercado de tarjetas de crédito en el sistema financiero de Ecuador a través de las cinco principales entidades financieras que lideran la emisión de tarjetas durante el período 2021- agosto 2024. Este análisis permite observar la participación de mercado de cada institución: Diners Club, Banco del Pacífico, Banco Pichincha, Banco Guayaquil y Produbanco, en términos de volumen de tarjetas de crédito emitidas anualmente.

Cada flujo en el diagrama refleja el número total de tarjetas emitidas por cada entidad en cada año, proporcionando una visión clara de cómo estas instituciones han competido y evolucionado en el mercado de tarjetas de crédito. El análisis visual facilita la comprensión de la concentración del mercado y muestra las variaciones en la participación de cada banco a lo largo de los años.

Se puede identificar patrones de consumo y preferencias de los clientes ecuatorianos, así como para analizar la competitividad y el posicionamiento de cada entidad en el sector financiero. Los datos obtenidos de esta distribución del mercado pueden ser utilizados para entender la dinámica del mercado de tarjetas de crédito en Ecuador.

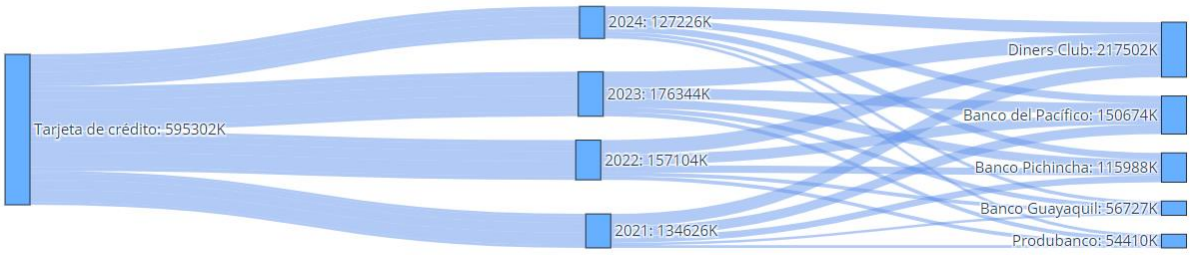


Figura 3.5. Distribucion del Mercado de las Tarjetas De Credito del Sistema Financiera de Ecuador (Elaboración Propia).

La Figura 3.6 muestra la evolución de los totales acumulados gestionados por diversas entidades financieras entre enero de 2021 y agosto de 2024. En el eje horizontal se observa el tiempo, mientras que el eje vertical refleja el total acumulado de cada institución financiera. Cada entidad está representada por una línea de color diferente, lo que facilita la visualización y comparación de las tendencias de crecimiento y el comportamiento de cada una a lo largo del periodo analizado.

Por otro lado, en el proceso de limpieza de datos se validó la integridad del conjunto de datos, pero no se detectó que algunas entidades financieras estaban presentes en algunos años, pero ausentes en otros, lo que indicaba datos faltantes. Como la información del Banco Internacional, que presenta información solo en el año 2023. Para evitar sesgos en el análisis y garantizar resultados consistentes, estas entidades fueron descartadas, ya que su inclusión podría afectar la interpretación de las tendencias generales del mercado financiero. Este ajuste asegura que las entidades representadas cuenten con información completa a lo largo de todo el periodo evaluado, permitiendo un análisis más confiable y preciso.

En términos generales, entidades como Banco Pichincha, Banco Rumiñahui y Diners Club exhiben una clara tendencia al alza, especialmente a partir de mediados de 2022. Este crecimiento podría sugerir un aumento en la base de clientes, en el

número de tarjetas de crédito emitidas o en el volumen de transacciones de estas instituciones. La tendencia sostenida de crecimiento sugiere la implementación de estrategias expansivas en estos bancos, que han logrado captar una mayor cuota de mercado a lo largo del tiempo.

Por otro lado, los bancos como Banco Internacional, Banco de Loja y Banco de Machala muestran un comportamiento más estable, con menores variaciones en sus totales acumulados durante el período analizado. Este comportamiento puede reflejar una estrategia más conservadora o enfocada en segmentos específicos del mercado, con un crecimiento moderado y consistente, pero sin grandes aumentos en la emisión de tarjetas de crédito.

Desde una perspectiva estacional, algunas instituciones, como Banco de Guayaquil y Banco Pichincha, presentan ligeras variaciones a lo largo del año. Estas fluctuaciones podrían estar asociadas a períodos de mayor actividad económica, como festividades o promociones comerciales, aunque no presentan cambios tan marcados como las tendencias de crecimiento general en las principales entidades. Sin embargo, los picos en estas fechas indican que algunas instituciones responden activamente a las oportunidades estacionales de mercado.

Finalmente, cabe destacar el caso del Banco Bolivariano, que presenta un notable incremento en sus totales acumulados a partir de noviembre de 2022, superando a otras instituciones y posicionándose como la entidad con el mayor total acumulado al final del período. Este crecimiento pronunciado sugiere una estrategia de expansión más agresiva o un impulso exitoso en la captación de clientes durante este tiempo. De manera similar, Diners Club y Banco Pichincha también muestran una tendencia de crecimiento sostenido, aunque a un ritmo más moderado.

Este análisis permite comprender las dinámicas y diferencias entre las estrategias de crecimiento de las instituciones financieras en Ecuador, proporcionando una visión clara sobre cómo cada entidad responde a la evolución del mercado de tarjetas de crédito y las oportunidades estacionales.

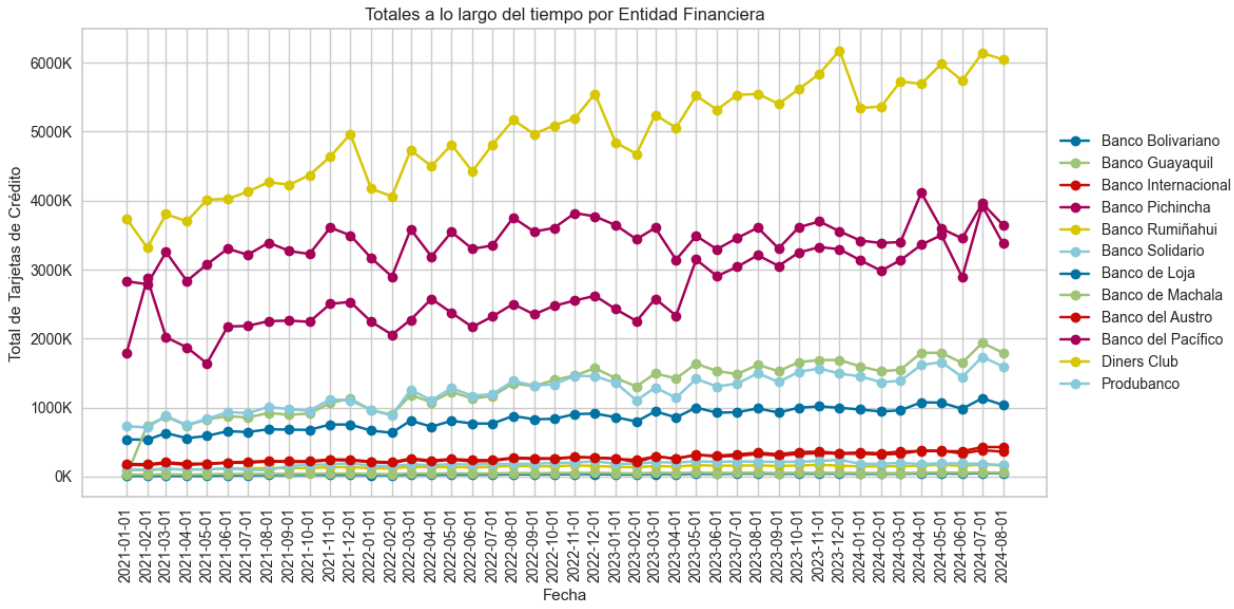


Figura 3.6. Evolución de los Totales por Entidad Financiera a lo Largo del Tiempo (Elaboración Propia).

El análisis exploratorio de los datos nos permitió dar una perspectiva clara y estratégica para la toma de decisiones de las entidades financieras en Ecuador. La evolución mensual y el volumen total de tarjetas emitidas permiten a las instituciones identificar su posición competitiva y ajustar sus estrategias para captar más clientes o fortalecer la lealtad de los actuales. La segmentación de tarjetas emitidas por perfil de cliente ayuda a las entidades a focalizar sus campañas en los segmentos con mayor demanda y rentabilidad, optimizando recursos en áreas de mayor impacto. La distribución del mercado entre diferentes entidades y la representación de flujos en el tiempo brindan una visión de concentración y dinámica competitiva, permitiendo evaluar oportunidades de expansión o consolidación. Finalmente, la identificación de patrones estacionales y tendencias generales ayuda a ajustar las campañas de marketing y políticas de emisión según las temporadas de mayor actividad, maximizando la captación y satisfacción de los clientes en momentos clave del año.

3.2 Implementación del pipeline de predicción

La selección, construcción y entrenamiento se va llevará mediante un enfoque de AutoML, que permite automatizar estas tareas y optimizar el proceso de modelado. El diseño de la solución del pipeline para la serie de tiempo se realiza a la agrupación de todas las entidades financieras para tener una visión general del sistema

financiero de tarjetas de crédito del Ecuador, con el pipeline definido en la Figura 3.7, se aplica para entidades financieras en específico, como las dos entidades financieras más grandes del país, Diners Club y Banco del Pichincha. Así, el pipeline proporciona tanto una perspectiva integral del mercado como un análisis detallado para actores clave del mercado.

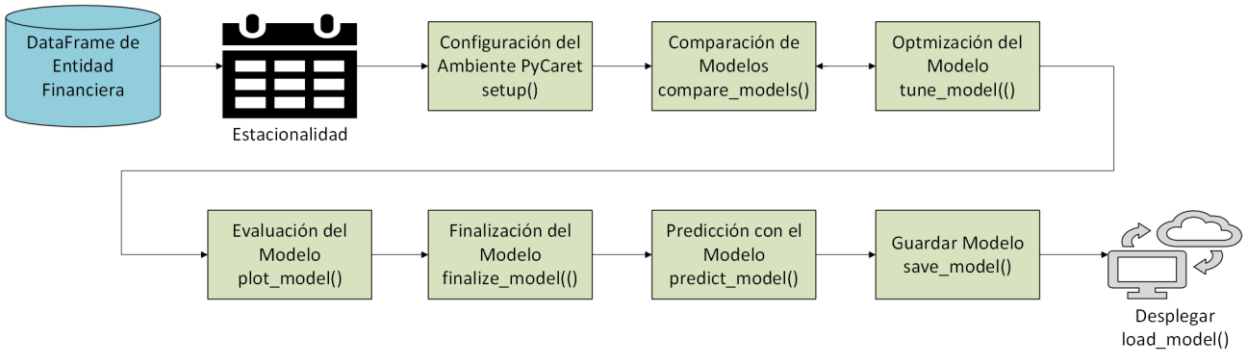


Figura 3.7. Pipeline de Predicción de Series de Tiempo con Pycaret (Elaboración Propia).

3.2.1 Datos de entidad financiera

Los datos de la entidad financiera se obtienen del conjunto de datos aplicado la limpieza y el procesamiento en la sección anterior, se procede a filtrar a la entidad de interés o en su defecto un total de todas las entidades financieras. A continuación, se muestra en la Figura 3.8 los datos del total de todas las entidades financieras.

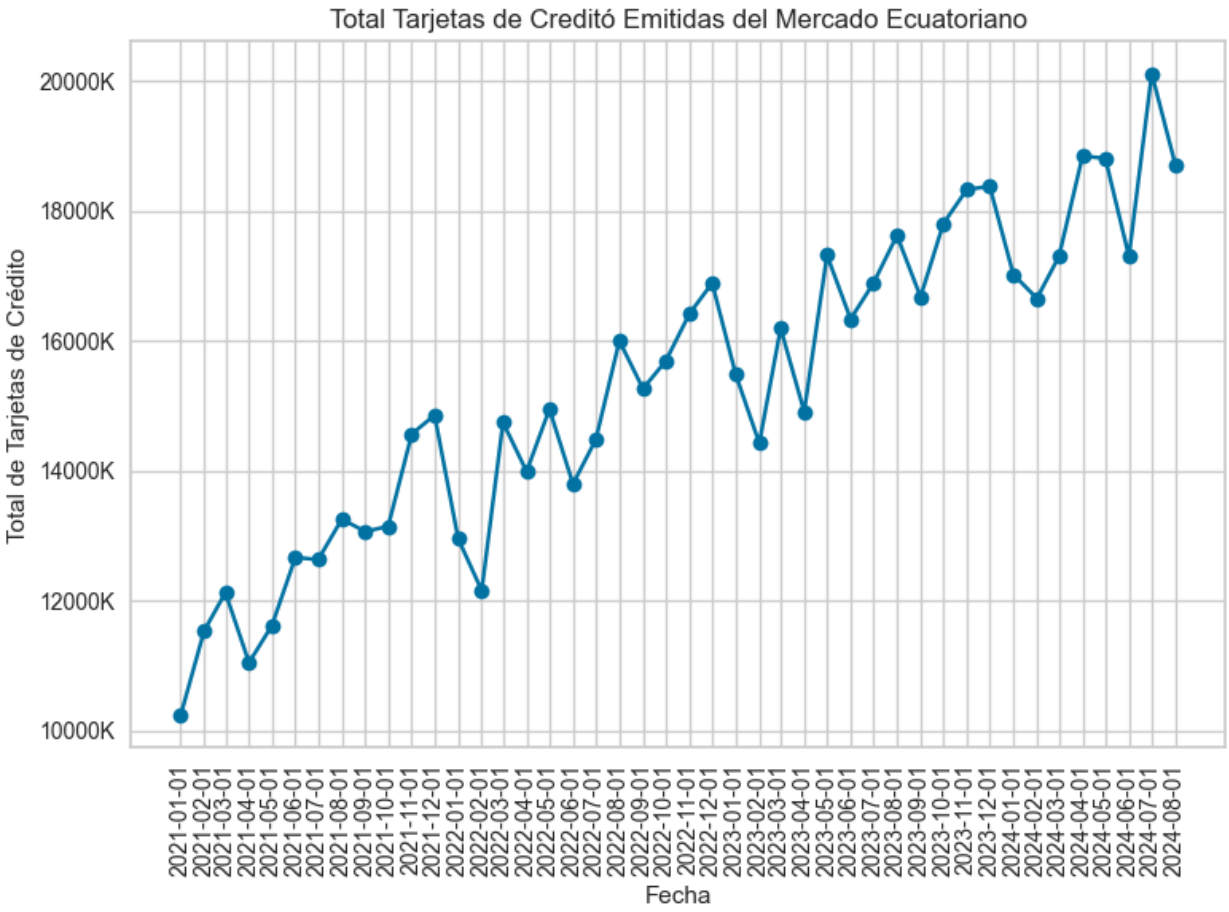


Figura 3.8. Serie de Tiempo de Cantidad de Tarjetas de Crédito Emitidas por el Sistema Financiero de Ecuador (Elaboración Propia).

3.2.2 Análisis de estacionalidad

La Figura 3.9 muestra una descomposición estacional de una serie temporal, comparando los períodos estacionales de cuatro meses y 12 meses. En cada descomposición, se presentan tres componentes: la tendencia, la estacionalidad y los residuos. Este análisis es fundamental para comprender los patrones subyacentes en la serie y evaluar la relevancia de distintos ciclos estacionales de la serie de tiempo original de la Figura 3.8.

- Tendencia: En ambos períodos, la tendencia capturada muestra una inclinación positiva, reflejando el crecimiento continuo en los valores de la serie temporal. La tendencia en la descomposición de cuatro meses parece capturar las fluctuaciones de corto plazo con mayor precisión, mientras que en la de 12 meses, la tendencia es más suave y refleja un patrón de crecimiento a largo plazo.

- Componente Estacional: En la descomposición con cuatro meses (parte superior), la estacionalidad es clara y presenta un patrón muy regular, con ciclos que se repiten consistentemente cada cuatro meses. Este comportamiento indica que la serie tiene una estacionalidad de corto plazo bien definida, posiblemente debido a variaciones trimestrales en el fenómeno estudiado. En la descomposición con 12 meses (parte inferior), se observa una estacionalidad más suave y menos pronunciada. Aunque hay ciclos que se repiten anualmente, las fluctuaciones no son tan marcadas como en el ciclo de cuatro meses, lo que sugiere que la estacionalidad de 12 meses tiene una menor influencia en la serie.
- Residuo: Los residuos en ambas descomposiciones muestran variabilidad y representan los componentes de la serie que no son explicados por la tendencia ni por la estacionalidad. En el caso de la estacionalidad de cuatro meses, los residuos muestran más fluctuaciones, indicando que hay variaciones no capturadas por el ciclo trimestral, aunque el patrón general de la serie se ajusta razonablemente bien. En la descomposición de 12 meses, los residuos son menos erráticos, sugiriendo que el ciclo anual contribuye parcialmente a explicar las variaciones, aunque su influencia es menor comparada con el ciclo de cuatro meses.

El análisis indica la presencia de una estacionalidad marcada con un período de cuatro meses, lo cual sugiere la existencia de ciclos repetitivos de corto plazo que tienen un impacto significativo en la serie temporal. Este patrón estacional podría estar relacionado con variaciones trimestrales características del fenómeno estudiado, como fluctuaciones en ventas, demanda, o cualquier otro indicador que varíe en intervalos trimestrales.

Por otro lado, aunque se observa cierta estacionalidad en el período de 12 meses, esta es más suave y menos regular. La menor prominencia del ciclo anual indica que, si bien puede existir un patrón estacional de largo plazo, este no es tan dominante ni tan claramente definido como el ciclo de cuatro meses. Por lo tanto, en el modelado y análisis de esta serie, el ciclo de cuatro meses debería considerarse el componente estacional principal, mientras que el ciclo de 12 meses podría tomarse como un factor secundario en la variabilidad de la serie.

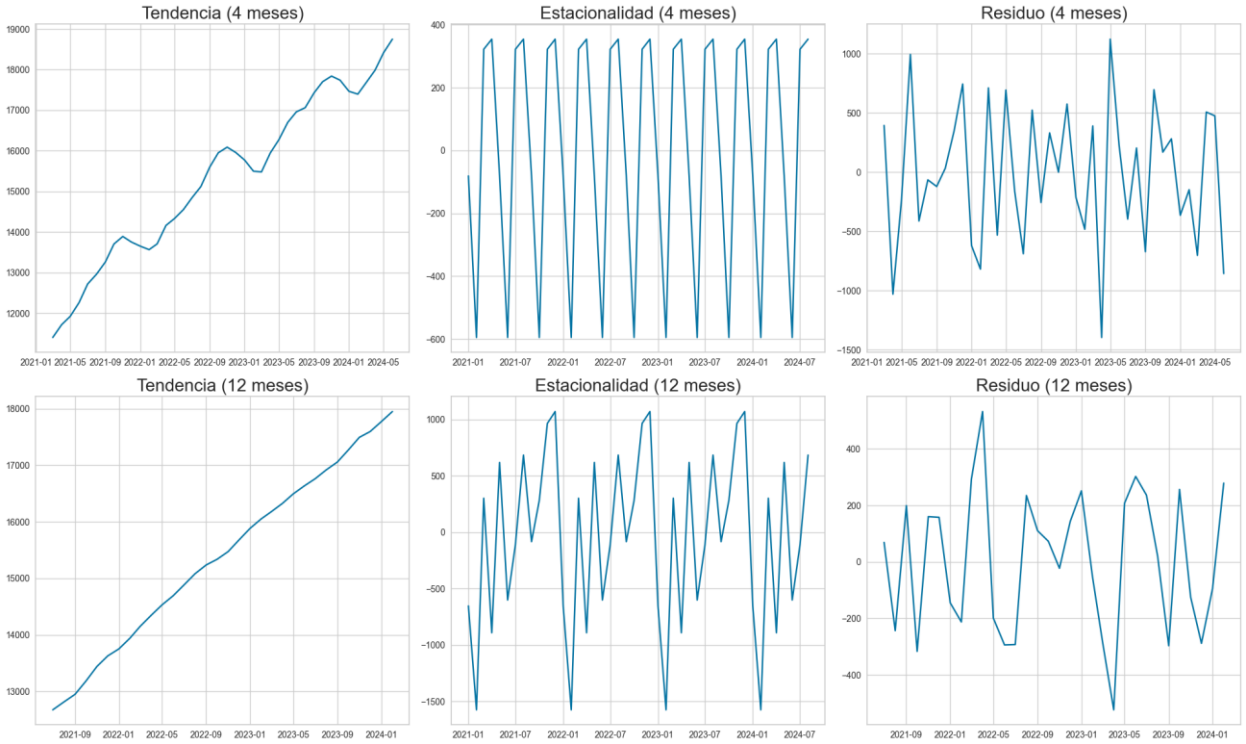


Figura 3.9. Análisis de Estacionalidad, Tendencia y Periodicidad (Elaboración Propia).

Para validar estadísticamente la presencia de estacionalidad en la serie de tiempo, se realizaron dos pruebas comunes en el análisis de series temporales: la prueba de Dickey-Fuller Aumentada (Dickey & Fuller, 1979)y la prueba KPSS (Kwiatkowski et al., 1992). La prueba ADF da como resultado un estadístico de -1.069 y un valor p de 0.7271, lo que indica que no podemos rechazar la hipótesis nula de no estacionariedad. Esto sugiere que la serie no es estacionaria según esta prueba. Por otro lado, la prueba KPSS mostró un estadístico de 0.975 y un valor p de 0.01, permitiéndonos rechazar la hipótesis nula de estacionariedad, lo cual también indica que la serie no es estacionaria. En conjunto, ambos resultados sugieren que la serie no es estacionaria, lo que implica la posible presencia de estacionalidad o tendencia, y, por lo tanto, podría ser necesario aplicar transformaciones como la diferenciación estacional para lograr estacionariedad en el análisis y en el modelado.

El análisis realizado es fundamental para la parametrización del modelo de predicción en una serie temporal. Identificar con precisión los patrones estacionales y la tendencia permite configurar el modelo con parámetros específicos que reflejan la estructura real de los datos.

3.2.3 Selección del modelo

En el proceso de configuración de un modelo de series temporales en PyCaret mediante la función pycaret.time_series.setup(), se especifican y ajustan varios parámetros clave como se muestra en Tabla 3.1, definen cómo el modelo analizará y estructurará los datos temporales. Estos parámetros determinan cómo se manejan aspectos como la estacionalidad, la validación cruzada, y la estructura general de la serie.

El parámetro Target especifica la columna objetivo en los datos en este caso la columna Total. Además, el Approach define el tipo de análisis que se llevará a cabo. En este caso, se utiliza un enfoque univariado, lo que significa que se está considerando únicamente la variable Total sin incluir variables exógenas adicionales en el modelo.

En términos de validación, Fold Generator y Fold Number son parámetros fundamentales para el modelado de series temporales en PyCaret. El generador de folds utilizado es el ExpandingWindowSplitter, el cual permite evaluar el modelo en diferentes segmentos del tiempo, respetando la estructura temporal y evitando que el modelo entrene con datos futuros. Además, Fold Number=4 indica que se realizarán cuatro divisiones o validaciones cruzadas, lo que ayuda a evaluar la estabilidad del modelo en diferentes períodos históricos.

Uno de los parámetros más críticos es el Seasonal Period, que define el ciclo estacional esperado en la serie. En el análisis anterior, se estableció seasonal_period=4, lo cual llevó a PyCaret a detectar una estacionalidad significativa en ciclos de 4 períodos (posiblemente trimestrales). Sin embargo, al definir seasonal_period=12, PyCaret no detectó estacionalidad significativa. Esto confirma el análisis de estacionalidad realizado previamente, donde se observó que el ciclo anual de 12 períodos es menos pronunciado y consistente que el ciclo de 4 períodos. Esta configuración implica que el modelo se ajustará mejor al patrón estacional de corto plazo (cuatro meses), reflejando variaciones cíclicas trimestrales en los datos.

Tabla 3.1. Parámetros de Configuración del Modelo de Predicción (Elaboración Propia).

Parámetro	Valor	Descripción
session_id	123	Identificador de la sesión para reproducibilidad.
Target	Total	Columna objetivo en los datos que se intenta predecir.
Approach	Univariate	Enfoque del modelo; "Univariate" indica que se analiza una sola variable de tiempo.
Exogenous Variables	Not Present	Indica si hay variables exógenas en el modelo; "Not Present" significa que no se usan variables adicionales.
Original data shape	(44, 1)	Tamaño de los datos originales (número de filas y columnas).
Transformed data shape	(44, 1)	Tamaño de los datos después de transformación (filas y columnas).
Transformed train set shape	(41, 1)	Tamaño del conjunto de entrenamiento transformado.
Transformed test set shape	(3, 1)	Tamaño del conjunto de prueba transformado.
Rows with missing values	0.0%	Porcentaje de filas con valores faltantes en los datos.
Fold Generator	ExpandingWindowSplitter	Método de generación de divisiones (folds) para validación cruzada.
Fold Number	4	Número de divisiones (folds) usadas en la validación cruzada.
Enforce Prediction Interval	False	Indica si se debe aplicar un intervalo de predicción.
Splits used for hyperparameters	all	Conjuntos de datos utilizados para ajustar hiperparámetros.
User Defined Seasonal Period(s)	4	Período de estacionalidad especificado manualmente por el usuario (en este caso, cada 4 períodos).
Ignore Seasonality Test	False	Indica si se debe ignorar la prueba automática de estacionalidad.
Seasonality Detection Algo	user_defined	Algoritmo utilizado para detectar la estacionalidad; en este caso, se usa el período especificado por el usuario.
Max Period to Consider	60	Máximo período de estacionalidad considerado en la detección automática.
Seasonal Period(s) Tested	[4]	Período(s) estacional(es) evaluado(s) para detectar patrones repetitivos.
Significant Seasonal Period(s)	[4]	Período(s) estacional(es) significativo(s) detectado(s) en los datos.

Significant Seasonal Period(s) without Harmonics	[4]	Período(s) estacional(es) significativo(s) sin armónicos.
Remove Harmonics	False	Indica si se deben eliminar los armónicos en la estacionalidad.
Harmonics Order Method	harmonic_max	Método utilizado para ordenar los armónicos en el análisis de estacionalidad.
Num Seasonalities to Use	1	Número de períodos estacionales que se utilizarán en el modelo.
All Seasonalities to Use	[4]	Lista de todos los períodos estacionales a usar.
Primary Seasonality	4	Período de estacionalidad principal detectado en los datos.
Seasonality Present	True	Indica si se detectó estacionalidad en la serie temporal. Para 12 periodos no detecta se establece en False.
Seasonality Type	mul	Tipo de estacionalidad detectada; "mul" indica una estacionalidad multiplicativa.
Target Strictly Positive	True	Indica si los valores de la variable objetivo son estrictamente positivos.
Target White Noise	No	Indica si la serie es ruido blanco (sin patrones); "No" significa que existen patrones.
Recommended d	1	Grado de diferenciación no estacional recomendado para la serie temporal.
Recommended Seasonal D	0	Grado de diferenciación estacional recomendado.
Preprocess	False	Indica si se realizaron preprocesamientos adicionales.
CPU Jobs	-1	Número de procesadores utilizados; "-1" indica que se usan todos los disponibles.
Use GPU	False	Indica si se utiliza GPU para acelerar el procesamiento.
Log Experiment	MlflowLogger	Herramienta de registro de experimentos, en este caso, se usa MlflowLogger.
Experiment Name	ts-default-name	Nombre asignado al experimento para el registro de resultados.
USI	4647	Identificador único de sesión, utilizado para rastrear experimentos en la plataforma.

3.2.4 Entrenamiento del modelo

PyCaret construye y entrena el modelo automáticamente. El usuario pude seleccionar el modelo o permite que compare_models() seleccione el mejor modelo. PyCaret maneja la configuración del entrenamiento, como la selección de hiperparámetros y la división de los datos. Se utiliza la función pycaret.time_series.compare_models(), esta función entrena y evalúa el rendimiento de todos los estimadores disponibles en la biblioteca de modelos utilizando validación cruzada. El resultado de esta función es una tabla de puntuaciones con los promedios de las puntuaciones de las validaciones cruzadas.

En la Figura 3.10 se comparan diversos modelos de pronóstico aplicados sobre una serie temporal, evaluando su rendimiento mediante métricas clave como: MASE (Mean Absolute Scaled Error), RMSE (Root Mean Squared Error), MAE (Mean Absolute Error) y R^2 (Coeficiente de determinación).

Se configura para elegir al mejor modelo que logre maximizar R^2, lo cual indica su eficacia en la captura de la variabilidad presente en los datos.

La Figura 3.10 resalta que el modelo ETS (Error, Trend, Seasonality) es la opción más robusta debido a su rendimiento en la métrica R^2, siendo el menos negativo (-0.1059) y, por tanto, el de mayor capacidad explicativa relativa en comparación con los demás. Aunque todos los modelos presentan valores negativos, lo que limita su capacidad para capturar completamente la variabilidad de la serie de tiempo, ETS logra un equilibrio destacado entre precisión y ajuste, con un RMSE de 701,524.92 y un MASE de 0.5190, ambos competitivos. Otros modelos, como el Polynomial Trend Forecaster (polytrend), muestran errores absolutos similares, pero con un R^2 más bajo (-0.1214), por lo tanto, el modelo ETS tiene es el mejor modelo al combinar precisión y la mejor capacidad explicativa disponible para este conjunto de datos.

	Model	MASE	RMSSE	MAE	RMSE	MAPE	SMAPE	R2	TT (Sec)
ets	ETS	0.5190	0.4984	628610.7723	701524.9205	0.0359	0.0360	-0.1059	0.0400
polytrend	Polynomial Trend Forecaster	0.5035	0.4856	611599.6137	684579.2862	0.0355	0.0348	-0.1214	0.0150
theta	Theta Forecaster	0.5409	0.5079	653722.0367	714359.3115	0.0372	0.0373	-0.1359	0.0675
exp_smooth	Exponential Smoothing	0.5384	0.5262	648892.2725	738151.8500	0.0371	0.0369	-0.2537	0.0450
br_cds_dt	Bayesian Ridge w/ Cond. Deseasonalize & Detrending	0.5170	0.5354	627504.3698	753619.0581	0.0366	0.0357	-0.3294	0.0525
lightgbm_cds_dt	Light Gradient Boosting w/ Cond. Deseasonalize & Detrending	0.5178	0.5358	628438.0077	754202.4146	0.0367	0.0357	-0.3315	0.2700
huber_cds_dt	Huber w/ Cond. Deseasonalize & Detrending	0.5275	0.5411	639277.5497	760024.6307	0.0374	0.0365	-0.3792	0.0575
omp_cds_dt	Orthogonal Matching Pursuit w/ Cond. Deseasonalize & Detrending	0.5201	0.5500	631025.7637	773445.3966	0.0369	0.0360	-0.3922	0.0525
ridge_cds_dt	Ridge w/ Cond. Deseasonalize & Detrending	0.5492	0.5733	665075.7054	805452.1654	0.0390	0.0380	-0.5405	0.0650
lasso_cds_dt	Lasso w/ Cond. Deseasonalize & Detrending	0.5492	0.5733	665075.7054	805452.1654	0.0390	0.0380	-0.5405	0.0525
llar_cds_dt	Lasso Least Angular Regressor w/ Cond. Deseasonalize & Detrending	0.5492	0.5733	665075.7054	805452.1654	0.0390	0.0380	-0.5405	0.0625
en_cds_dt	Elastic Net w/ Cond. Deseasonalize & Detrending	0.5492	0.5733	665075.7054	805452.1654	0.0390	0.0380	-0.5405	0.3100
lr_cds_dt	Linear w/ Cond. Deseasonalize & Detrending	0.5492	0.5733	665075.7054	805452.1654	0.0390	0.0380	-0.5405	0.2900
rf_cds_dt	Random Forest w/ Cond. Deseasonalize & Detrending	0.5675	0.6333	688393.9560	892621.6926	0.0405	0.0390	-1.0316	0.1225
ada_cds_dt	AdaBoost w/ Cond. Deseasonalize & Detrending	0.6072	0.6563	736096.0992	923912.6078	0.0429	0.0414	-1.1167	0.0825
et_cds_dt	Extra Trees w/ Cond. Deseasonalize & Detrending	0.5906	0.6612	717025.1727	931536.0365	0.0421	0.0404	-1.1887	0.1075
knn_cds_dt	K Neighbors w/ Cond. Deseasonalize & Detrending	0.6500	0.6836	789299.0955	964140.7064	0.0463	0.0445	-1.2515	0.1350
catboost_cds_dt	CatBoost Regressor w/ Cond. Deseasonalize & Detrending	0.6738	0.7003	816771.0401	985876.4036	0.0477	0.0459	-1.4128	0.9975
xgboost_cds_dt	Extreme Gradient Boosting w/ Cond. Deseasonalize & Detrending	0.6579	0.6908	795300.6572	969980.5350	0.0462	0.0457	-1.4155	0.4200
gbr_cds_dt	Gradient Boosting w/ Cond. Deseasonalize & Detrending	0.6926	0.6977	835319.5007	979406.1368	0.0486	0.0481	-1.6148	0.0825
stlf	STLF	0.7743	0.7784	932816.7676	1092829.2744	0.0542	0.0538	-1.8883	0.0300
naive	Naive Forecaster	0.8010	0.7813	965005.5000	1094348.6903	0.0547	0.0551	-1.8938	2.0825
auto_arima	Auto ARIMA	0.8010	0.7813	965005.5000	1094348.6903	0.0547	0.0551	-1.8938	0.1950
dt_cds_dt	Decision Tree w/ Cond. Deseasonalize & Detrending	0.7971	0.7721	961677.3964	1085074.1011	0.0551	0.0548	-2.2579	0.0525
arima	ARIMA	0.8044	0.8433	969340.4334	1181589.5608	0.0561	0.0558	-2.4940	0.1350
snaive	Seasonal Naive Forecaster	1.0005	0.9389	1201264.0000	1311378.2876	0.0685	0.0709	-4.2868	0.0625
croston	Croston	1.5182	1.3952	1829392.2332	1956590.7437	0.1029	0.1094	-8.5060	0.0150
grand_means	Grand Means Forecaster	2.6016	2.2883	3138926.7085	3210941.0375	0.1773	0.1954	-23.0150	1.5275

Figura 3.10. Comparación de Rendimiento de Modelos de Series de Tiempo (Elaboración Propia).

3.2.5 Optimización de hiperparámetros

La optimización de hiperparámetros es una etapa crucial en el proceso de modelado, ya que permite ajustar los parámetros del modelo con el objetivo de mejorar su rendimiento. La función pycaret.time_series.tune_model() en PyCaret permite realizar el ajuste automático de hiperparámetros para optimizar el rendimiento del modelo de series temporales. Al utilizar esta función, PyCaret evalúa diferentes combinaciones de hiperparámetros clave del modelo seleccionado con el objetivo de minimizar o maximizar una métrica específica de error, como el RMSE, MAE, MAPE, entre otras.

La siguiente Figura 3.11 ilustra las métricas de desempeño del modelo de series temporales en distintos puntos de corte después del proceso de ajuste de hiperparámetros mediante tune_model(). En términos de precisión, el modelo ajustado presenta un MASE promedio de 0.8516 y un RMSE de 871,104.35, lo que indica un nivel de error moderado en las predicciones. El MAE promedio es de 772,334.27, mientras que las métricas de error porcentual, MAPE y SMAPE, son de 0.0486 y 0.0471, respectivamente. Esto sugiere que el modelo ajustado mantiene un error porcentual relativamente bajo en comparación con los valores reales, lo cual indica una precisión aceptable en términos absolutos.

En cuanto al coeficiente de determinación (R^2), el valor promedio después del ajuste es de -1.0533. Aunque este valor sigue siendo negativo, lo cual indica una capacidad limitada del modelo para explicar la variabilidad de la serie temporal, representa un ligero avance en comparación del valor de entrenameinto. Esto implica que el modelo ajustado ha mejorado en su habilidad explicativa, aunque de manera limitada, y sugiere que el ajuste de hiperparámetros ha permitido optimizar ciertos aspectos del modelo.

	cutoff	MASE	RMSSE	MAE	RMSE	MAPE	SMAPE	R2
0	2022-04	0.7556	0.6627	668264.2500	739678.0747	0.0460	0.0451	0.1488
1	2022-08	0.4109	0.4316	380311.8741	482219.8130	0.0231	0.0235	0.4133
2	2022-12	1.8517	1.6755	1605956.1332	1753656.7754	0.1074	0.1008	-6.0532
3	2023-04	0.4554	0.4400	429538.0000	487077.0829	0.0251	0.0253	-0.0138
4	2023-08	0.4805	0.4438	467316.8832	508520.3588	0.0266	0.0265	0.4493
5	2023-12	1.1555	1.1308	1082618.4714	1255473.9687	0.0638	0.0612	-1.2645
Mean	NaT	0.8516	0.7974	772334.2687	871104.3456	0.0486	0.0471	-1.0533
SD	NaT	0.5149	0.4631	440881.4492	478554.5822	0.0299	0.0275	2.3086

Figura 3.11. Resultado de la Optimización de Hiperparametros (Elaboración Propia).

3.3 Evaluación del modelo

La función pycaret.time_series.plot_model() es una herramienta útil para evaluar el rendimiento del modelo de manera visual. La tendencia de las predicciones en la Figura 3.12 refleja un crecimiento moderado en los valores futuros, alineándose con el patrón ascendente que se observa en la serie original. Esto sugiere que el modelo

ETS ha capturado de manera adecuada la tendencia subyacente en los datos históricos, proyectando una continuidad en el comportamiento ascendente de la serie.

El intervalo de predicción, representado por la zona sombreada en azul claro, proporciona un rango de confianza alrededor de las predicciones centrales, mostrando la variabilidad esperada en los valores futuros. Este intervalo es relativamente estrecho, lo cual indica un nivel de confianza razonable en las proyecciones, aunque también admite cierta variabilidad, que es común en modelos de series temporales. Esto significa que, aunque las predicciones del modelo son generalmente confiables, existe un margen de error inherente que debe ser considerado en la interpretación de los resultados.

En términos generales, el modelo ETS parece ajustarse bien a la tendencia de largo plazo de la serie y a ciertos patrones cíclicos, aunque es posible que no capture todas las fluctuaciones de corto plazo observadas en los datos históricos. Este comportamiento es esperado en modelos de series temporales que buscan captar patrones amplios, más que variaciones menores. En conjunto, el gráfico muestra que el modelo ETS proporciona predicciones razonables para el futuro cercano, ofreciendo una visión confiable de los valores proyectados para los próximos meses y permitiendo una mejor planificación y toma de decisiones informadas dentro de los rangos de incertidumbre indicados.

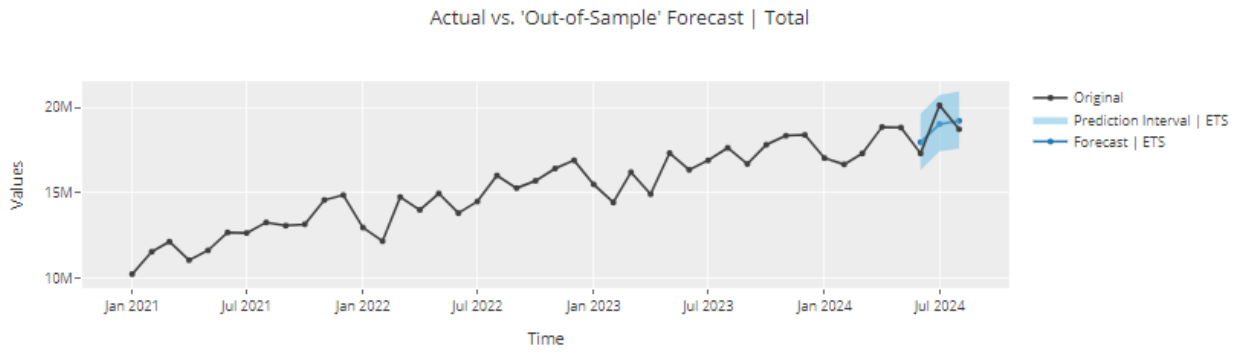


Figura 3.12. Comparación de Valores Reales y Predicciones del Modelo (Elaboración Propia).

Los valores predichos con los datos de validación del modelo se utilizan para evaluar el desempeño del modelo en un conjunto de datos que no fue utilizado en el

entrenamiento inicial, permitiendo así medir la precisión de las predicciones en un contexto "fuera de muestra.

Estos valores han sido generados mediante el modelo ETS ajustado y representan las predicciones para el período de validación, proporcionando una estimación de cómo se comportaría el modelo en datos futuros. La precisión de estas predicciones se evaluaría mediante métricas de desempeño como RMSE, MAE y MAPE, permitiendo confirmar la confiabilidad del modelo en escenarios reales de predicción y verificar si captura adecuadamente la tendencia y estacionalidad observadas en la serie temporal.

A continuación en laTabla 3.2, se presentan los valores predichos para los meses de validación.

Tabla 3.2. Valores Reales y Predichos con el Modelo (Elaboración Propia).

Año - Mes	Valor Real	Predicción
2024-06	17,299,736	17,898,940
2024-07	20,113,790	18,980,840
2024-08	18,710,862	19,236,020

La Tabla 3.3 muestra las métricas de desempeño del modelo ETS (Error, Trend, Seasonality) en la predicción de la serie temporal. En términos de precisión, el modelo presenta un MASE de 0.6315 y un RMSE de 788,089.40, lo cual indica un nivel de error moderado en las predicciones. El MAE (error absoluto medio) es de 746,174.83, sugiriendo que, en promedio, las predicciones del modelo difieren de los valores reales por esta cantidad en unidades absolutas.

Además, los valores de MAPE y SMAPE son bajos, con 0.0395 y 0.0396, respectivamente, lo que implica que el error porcentual medio en relación con los valores reales es bajo, evidenciando una buena precisión relativa del modelo en términos de error porcentual.

Es particularmente destacable que el modelo presenta un coeficiente de determinación R^2 de 0.5294, lo cual indica que el modelo explica aproximadamente

el 52.94% de la variabilidad observada en la serie temporal. Este valor positivo y relativamente alto para un modelo de series temporales sugiere que el modelo ETS no solo logra un ajuste adecuado en términos de error absoluto, sino que también captura una porción considerable de la estructura subyacente en los datos, incrementando así su capacidad explicativa.

En general, estas métricas indican que el modelo ETS es una opción sólida para la predicción de esta serie temporal, proporcionando predicciones precisas y un buen nivel de ajuste a los patrones presentes en los datos.

Tabla 3.3. Resultados de Metricas de Evaluación del Modelo (Elaboración Propia).

Modelo	MASE	RMSSE	MAE	RMSE	MAPE	SMAPE	R^2
ETS	0.6315	0.5755	746174.8308	788089.3960	0.0395	0.0396	0.5294

La Figura 3.13 muestra una comparación entre los valores reales (Total), las predicciones generadas por el modelo (ETS) y los residuos (Error) a lo largo del tiempo, desde enero de 2021 hasta agosto de 2024. La línea azul representa los valores reales, mientras que la línea naranja corresponde a las predicciones del modelo ETS. Los residuos, que son la diferencia entre los valores reales y las predicciones, se visualizan en rojo.

Es importante destacar que los residuos se mantienen en su mayoría cercanos a cero, lo que indica que el modelo ha logrado capturar de manera precisa la tendencia de los datos a lo largo del tiempo. Esta proximidad al cero sugiere que las predicciones del modelo son confiables y que los errores de estimación son generalmente bajos. La cercanía de los residuos a cero implica que las desviaciones entre los valores reales y las predicciones son mínimas, lo cual es un indicador de un buen ajuste del modelo.

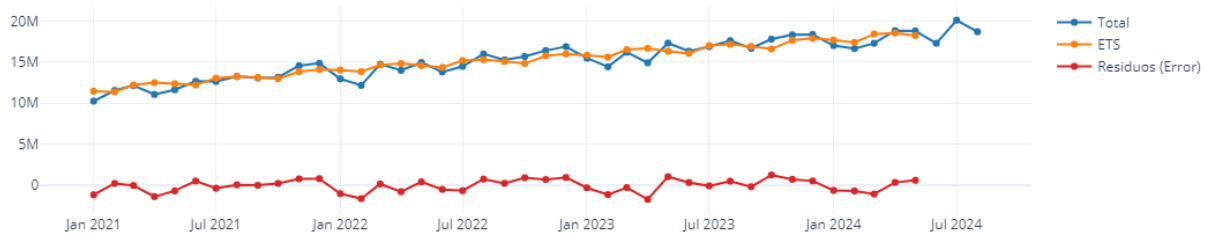


Figura 3.13. Análisis de Valores Reales, Predicciones y Residuos en la Cantidad de Tarjetas de Crédito Emitidas (Elaboración Propia).

3.4 Predicción con el modelo

La función pycaret.time_series.finalize_model() en PyCaret entrena el modelo seleccionado utilizando todo el conjunto de datos disponible, incluyendo los datos que se usaron previamente para validación. Esto se realiza en la fase final del modelado, cuando se ha identificado el mejor modelo y se desea preparar para su uso en producción. Al entrenar el modelo con todos los datos, finalize_model permite aprovechar toda la información disponible, mejorando potencialmente la precisión de las predicciones futuras. Este paso deshabilita la validación cruzada, ya que el modelo ahora utiliza la totalidad del conjunto de datos, y está listo para realizar predicciones en nuevos datos en un entorno de producción.

La Figura 3.14, muestra el comportamiento de la serie temporal histórica (en línea negra) junto con las predicciones generadas fuera de muestra por el modelo ETS para los próximos tres meses (en línea azul). Estas predicciones vienen acompañadas de un intervalo de predicción en un área sombreada en azul claro, que representa el rango de confianza en el cual se espera que caigan los valores futuros.

El intervalo de predicción, representado por la zona sombreada en azul claro, proporciona un margen de confianza para las proyecciones futuras. Este intervalo relativamente estrecho indica un nivel de certeza razonable en las predicciones del modelo ETS, aunque también permite visualizar la incertidumbre inevitable en cualquier proyección de series temporales. La estabilidad del intervalo refuerza la idea de que el modelo espera una continuidad en el comportamiento de la serie, sin

grandes desviaciones, lo cual es un indicador de confianza en las proyecciones realizadas para el próximo trimestre.

El modelo ETS finalizado, entrenado con todos los datos históricos disponibles, proporciona predicciones fiables para los próximos meses, capturando tanto la tendencia general ascendente de la serie como variaciones menores en el corto plazo. Las predicciones específicas para septiembre, octubre y noviembre de 2024 ofrecen una base sólida para la planificación, permitiendo anticipar el comportamiento de la serie con una visión clara y fundamentada. En conjunto, el gráfico y las predicciones sugieren que el modelo ETS es una herramienta adecuada para pronósticos a corto plazo, brindando información útil para la toma de decisiones estratégicas.

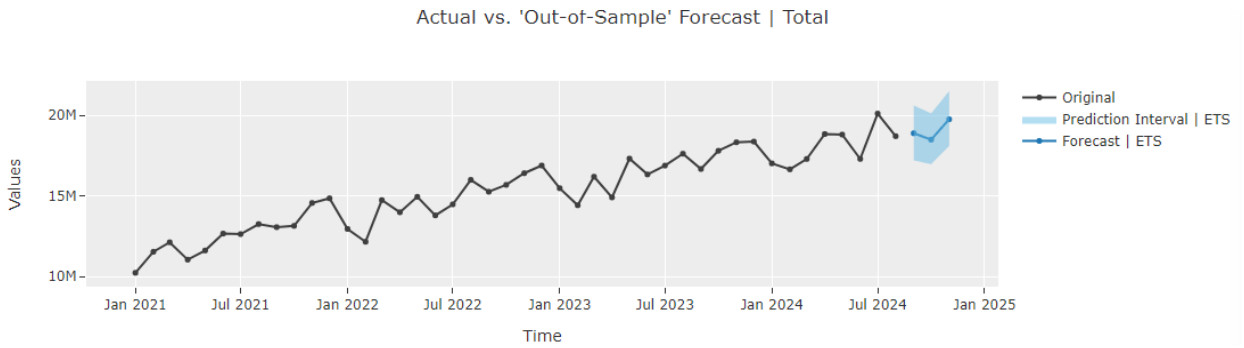


Figura 3.14. Predicción de la Cantidad de Tarjetas de Crédito Emitidas para los Siguientes 3 Meses (Elaboración Propia).

La Tabla 3.4 muestra que, el modelo predice para septiembre de 2024 un valor de 18892582, seguido por una ligera disminución en octubre de 2024 a 18495000 y un posterior aumento en noviembre de 2024 a 19761607. Este patrón de variación a corto plazo, donde se observa una leve caída en octubre seguida de una recuperación en noviembre, sugiere que el modelo ha capturado algún nivel de estacionalidad o ciclo en la serie. Esto es indicativo de un modelo que, además de reflejar la tendencia de crecimiento general, también responde a patrones temporales más sutiles dentro de los datos históricos.

Tabla 3.4. Valores Predichos con el Modelo (Elaboración Propia).

Año- Mes	Predicción
2024-09	18,892,582
2024-10	18,495,000
2024-11	19,761,607

Una vez completadas las etapas de entrenamiento, ajuste (tuning) y finalización del modelo en PyCaret, se procede a utilizar pycaret.time_series.save_model() para exportarlo a un entorno externo. Esto permite que el modelo pueda cargarse posteriormente mediante la función pycaret.time_series.load_model(), asegurando así su disponibilidad para futuros análisis o aplicaciones en producción. Este enfoque optimiza el flujo de trabajo al reducir el tiempo de procesamiento y garantizar la replicabilidad del modelo entrenado en diferentes entornos o dispositivos.

3.5 Aplicación de pipeline en entidad financiera.

En la siguiente sección, se implementa el pipeline desarrollado para dos entidades representativas del conjunto de datos: Diners Club, la entidad financiera con la mayor cantidad de tarjetas de crédito, y Produbanco, una entidad financiera que se caracteriza por tener una cantidad promedio de tarjetas de crédito en el mercado.

3.5.1 Diners Club.

Según la Figura 3.15, indica que el modelo predice de manera precisa, manteniéndose cercano a los valores reales y dentro del intervalo de confianza en las proyecciones. La capacidad del Theta Forecaster para capturar la tendencia de crecimiento y ajustarse bien a los datos pasados, con un bajo margen de error, lo convierte en una herramienta confiable para realizar predicciones en series temporales similares. Este ajuste preciso y la inclusión del intervalo de predicción refuerzan la confiabilidad del modelo para estudios prospectivos y aplicaciones en el análisis de datos de series temporales.

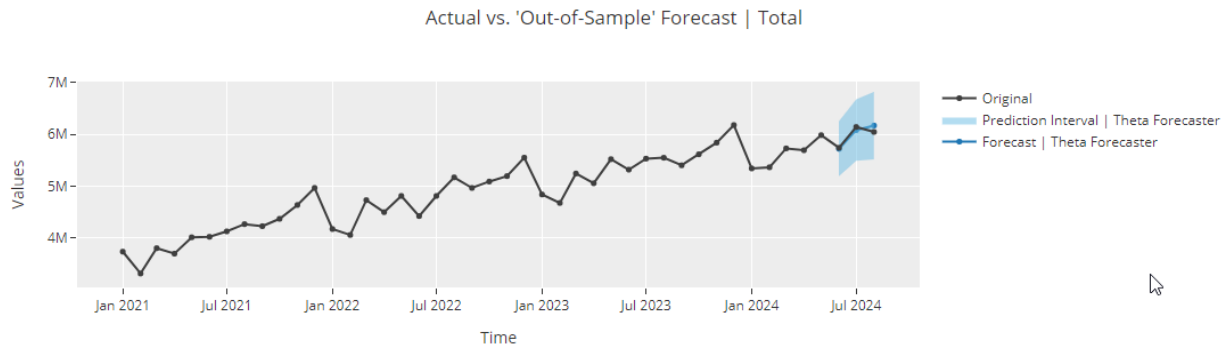


Figura 3.15. Predicción de Cantidad de Tarjetas de Crédito de Diners Club (Elaboración Propia).

La Figura 3.16 muestra una comparación entre datos reales, predicciones del modelo y residuos (errores) a lo largo del tiempo, indicando que el modelo predictivo es bastante preciso. La línea de datos reales (azul) presenta una tendencia de crecimiento constante desde 2021 hasta 2024, mientras que la línea de predicciones (naranja) sigue de cerca esta tendencia, con pequeñas diferencias. Los residuos (línea roja) son bajos y estables, sin grandes variaciones, lo cual sugiere que la precisión del modelo se mantiene constante. En general, el modelo logra capturar eficazmente la tendencia de crecimiento en los datos con un margen de error relativamente bajo, demostrando ser confiable para futuras predicciones.

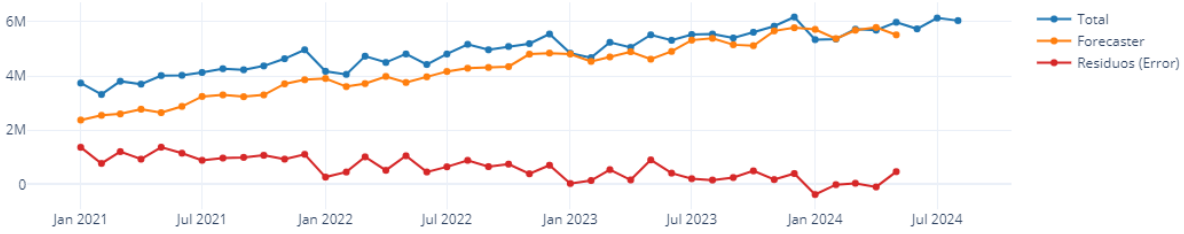


Figura 3.16. Análisis de Valores Reales, Predicciones y Residuos en la Cantidad de Tarjetas de Crédito Emitidas en Diners Club (Elaboración Propia).

El modelo Theta Forecaster detallado en la Tabla 3.5 muestra un buen desempeño en términos de la métrica R^2, con un valor de 0.7822, lo que indica que el modelo explica aproximadamente el 78.22% de la variabilidad de los datos observados. Este alto valor de R^2 sugiere que el modelo es efectivo para capturar la tendencia de los datos y hacer predicciones cercanas a los valores reales. Otros indicadores, como el MAPE (0.0112) y SMAPE (0.0111), reflejan un bajo porcentaje de error, reforzando la conclusión de que el modelo es preciso y confiable en su capacidad predictiva.

Tabla 3.5. Resultados de Metricas de Evaluación del Modelo para Diners Club (Elaboración Propia).

Model	MASE	RMSSE	MAE	RMSE	MAPE	SMAPE	R^2
Theta Forecaster	0.1785	0.1842	67486.9448	79833.4617	0.0112	0.0111	0.7822

La Tabla 3.6 presenta los valores reales y las predicciones del modelo que en el periodo de junio a agosto de 2024. Durante este periodo el modelo muestra predicciones muy cercanas a los valores reales. En junio, la predicción es de 5,720,425 frente a un valor real de 5,739,995; en julio, predice 6,081,442 frente a un valor real de 6,141,165; y en agosto, la predicción es de 6,168,656, con un valor real de 6,045,488. Estas pequeñas diferencias sugieren que el modelo tiene una buena capacidad para capturar la tendencia real, manteniéndose consistentemente cercano a los valores observados.

Tabla 3.6. Valores Reales y Predichos con el Modelo para Diners Club (Elaboración Propia).

Año - Mes	Valor Real	Predicción
2024-06	5,739,995	5,720,425
2024-07	6,141,165	6,081,442
2024-08	6,045,488	6,168,656

3.5.2 Produbanco.

El modelo parece capturar la tendencia de los datos históricos, proyectando un aumento que se mantiene dentro del intervalo de predicción. La banda azul indica la incertidumbre en las predicciones, ampliándose en los puntos finales, lo cual es común en los modelos de series temporales debido al aumento de la incertidumbre conforme se proyecta más lejos en el futuro.

La Figura 3.17 ilustra la precisión del Theta Forecaster para realizar predicciones fuera de la muestra (out-of-sample), lo que permite evaluar su efectividad y la confianza en sus proyecciones futuras, especialmente útil para anticipar tendencias bajo un margen de error definido.

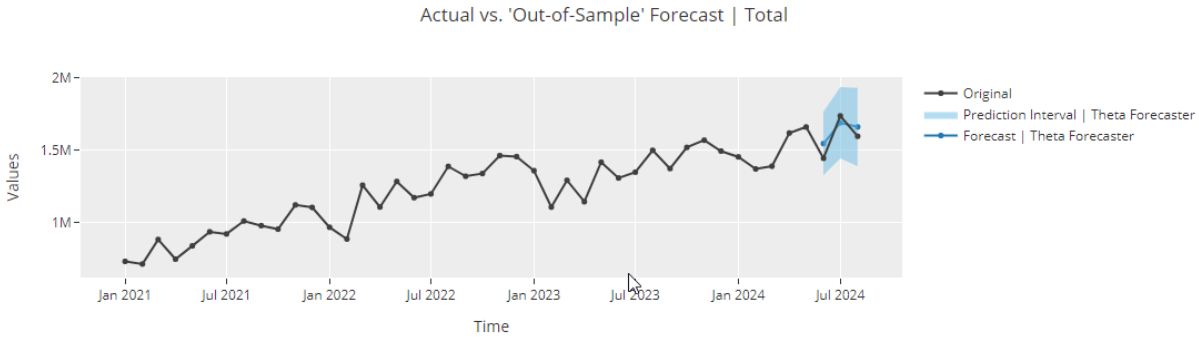


Figura 3.17. Predicción de Cantidad de Tarjetas de Crédito de Produbanco (Elaboración Propia).

La Figura 3.18 muestra la serie temporal de valores observados (línea azul), pronosticados (línea naranja) y los residuos o errores (línea roja) entre enero de 2021 y julio de 2024. Los valores observados presentan una tendencia de crecimiento constante con variaciones cíclicas, mientras que los pronósticos siguen de cerca esta tendencia, lo que indica la precisión del modelo predictivo. Los residuos oscilan alrededor de cero, reflejando un bajo error y la ausencia de un sesgo significativo en el modelo. Los picos ocasionales en los residuos sugieren momentos de mayor desviación entre los valores observados y pronosticados, destacando posibles áreas de mejora en el modelo.

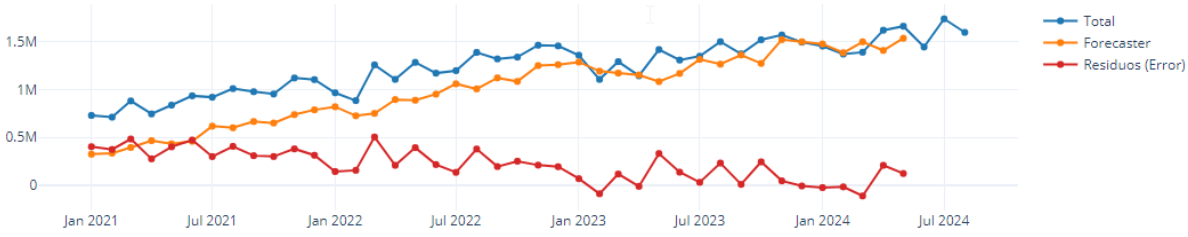


Figura 3.18. Análisis de Valores Reales, Predicciones y Residuos en la Cantidad de Tarjetas de Crédito Emitidas en Produbanco (Elaboración Propia).

El modelo Theta Forecaster presenta resultados de evaluación sólidos en la Tabla 3.7, con un bajo error absoluto y porcentual, reflejado en métricas como MASE (0.4832), RMSSE (0.4200), MAPE (4.59%) y SMAPE (4.50%). Estas métricas indican que el modelo tiene una alta precisión en las predicciones en relación con la escala de los datos. El MAE y RMSE, con valores de 70,922 y 74,455 respectivamente, sugieren una baja magnitud de error en términos absolutos y

cuadrados. Sin embargo, el coeficiente de determinación R^2 de 0.6127 indica que el modelo explica aproximadamente el 61.27% de la variabilidad de los datos, lo cual, aunque aceptable, sugiere que existe margen para mejorar la capacidad predictiva. En conjunto, el Theta Forecaster demuestra un buen ajuste y precisión en las predicciones, siendo una opción válida para la serie temporal analizada.

Tabla 3.7. Resultados de Metricas de Evaluación del Modelo para Produbanco (Elaboración Propia).

Modelo	MASE	RMSSE	MAE	RMSE	MAPE	SMAPE	R^2
Theta Forecaster	0.4832	0.4200	70922.4768	74455.2131	0.0459	0.0450	0.6127

La Tabla 3.8 muestra que en los meses de junio, julio y agosto de 2024, el modelo Theta Forecaster muestra predicciones cercanas, pero no exactas a los valores reales, con una sobreestimación en junio (101,461 unidades) y agosto (64,075 unidades) y una ligera subestimación en julio (47,231 unidades). Estas desviaciones indican que, aunque el modelo sigue de cerca la tendencia general de los datos, aún presenta margen de mejora para capturar con mayor precisión las variaciones mensuales, especialmente en los picos. En general, el modelo ofrece un desempeño adecuado, con errores moderados en sus estimaciones.

Tabla 3.8. Valores Reales y Predichos con el Modelo para Produbanco (Elaboración Propia).

Año - Mes	Valor Real	Predicción
2024-06	1,442,622	1,544,083
2024-07	1,735,609	1,688,378
2024-08	1,595,036	1,659,111

3.6 Prototipo para usuario final.

Para la implementación del modelo de predicción y facilitar su uso por parte del usuario final, se utilizó la biblioteca Streamlit, una herramienta en Python diseñada para la creación de aplicaciones web interactivas. Streamlit permite desplegar visualizaciones de datos, gráficos de predicción y métricas de evaluación de manera amigable y accesible. En este caso, se desarrolló una aplicación donde el usuario

puede seleccionar una entidad financiera y un horizonte de predicción para obtener las proyecciones de la cantidad de tarjetas de crédito.

El pipeline de solución desarrollado permite generar modelos predictivos para varias entidades financieras. Cada modelo se almacena como un archivo en formato pkl, el cual es cargado posteriormente en el aplicativo web para realizar predicciones. La interfaz del sistema muestra los valores históricos reales y proyecta las predicciones futuras generadas por el modelo, como se observa en la Figura 3.19. En esta pantalla, el usuario puede configurar tanto la entidad financiera como el horizonte de predicción deseado.

Figura 3.19. Pantalla de Aplicación Web para Configurar el Modelo y Periodo a Predecir (Elaboración Propia).

Una vez seleccionada la entidad financiera y definido el horizonte de predicción, la aplicación web proporciona una experiencia de usuario intuitiva y eficiente. Permite a los interesados visualizar las proyecciones tanto de forma gráfica como en formato tabular, facilitando una interpretación rápida y clara de los resultados. La Figura 3.20 muestra la pantalla de la aplicación web, utilizada para la configuración del modelo y el período de predicción.

Figura 3.20 Resultados de Predicción desde la Página Web (Elaboración Propia).

Se integra un dashboard interactivo desarrollado en Power BI dentro del aplicativo web, lo que permite al usuario final explorar los datos de manera intuitiva y detallada. La Figura 3.21 presenta el dashboard del conjunto de datos, diseñado para facilitar el análisis visual de la emisión de tarjetas de crédito por entidades financieras en Ecuador. La aplicación ha sido publicada en Streamlit Cloud, lo que permite a los usuarios acceder al sistema a través de un enlace web, eliminando la necesidad de instalar software adicional. Esto garantiza una experiencia accesible y eficiente para todos los involucrados.

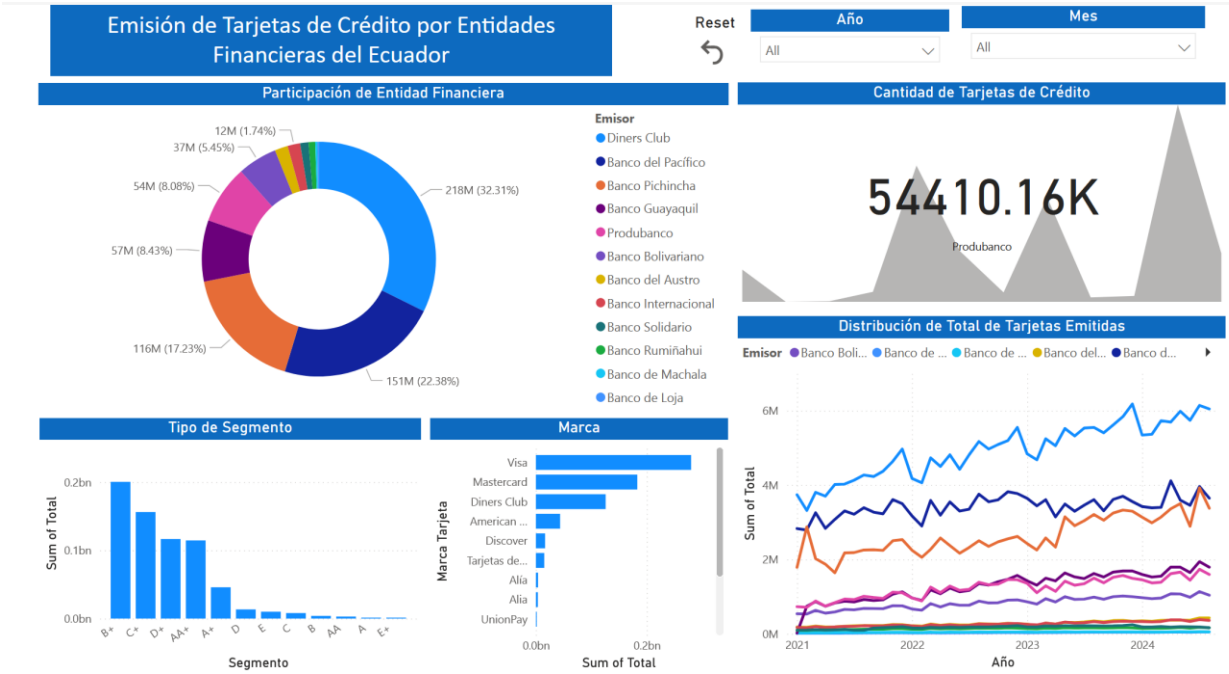


Figura 3.21. Dashboard del Conjunto de Datos (Elaboración Propia).

CAPÍTULO 4

4. EVALUACION DE RESULTADOS

En este capítulo se presenta la evaluación de los modelos predictivos desarrollados para la proyección de la cantidad de tarjetas de crédito emitidas, con un enfoque comparativo entre un modelo base y modelos avanzado. Se analizan los resultados obtenidos, tanto en términos de precisión de las predicciones como en su capacidad para capturar patrones complejos en las series temporales. Además, se exploran los resultados de la validación con un conjunto de datos independientes. Finalmente, se discuten los sesgos potenciales y los costos asociados con la implementación de la solución propuesta.

4.1 Evaluación del modelo frente a un modelo básico

El modelo Naive es una herramienta simple y efectiva para realizar predicciones en series temporales, basado únicamente en el último valor observado. Su principal fortaleza radica en su simplicidad: asume que el futuro será igual al dato más reciente, lo que lo convierte en un excelente punto de partida o referencia (línea base) para comparar con modelos más avanzados. Además, es rápido de implementar, con bajo costo computacional y altamente interpretable, lo que facilita su uso en entornos donde la rapidez y la claridad son esenciales.

En la Figura 4.1, se observa cómo el modelo Naive (línea naranja) sigue la forma general de la serie original, aunque con un desfase, ya que utiliza el último valor conocido para predecir el siguiente período. Por ejemplo, el valor predicho para junio de 2024 corresponde al valor observado en mayo de 2024.

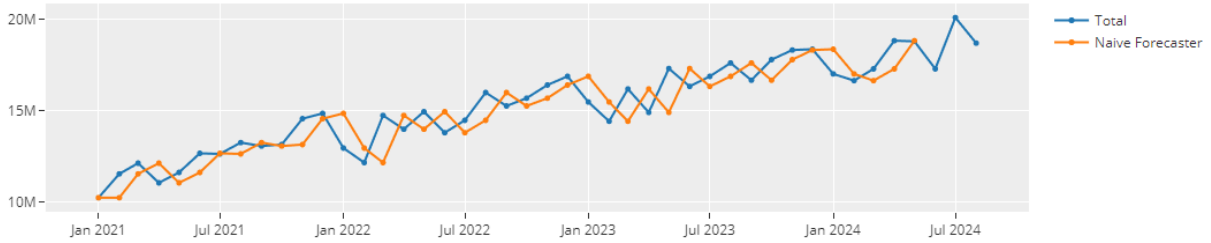


Figura 4.1 Comportamiento del Modelo Base Naive y Valores Reales de la Serie De Tiempo (Elaboración Propia).

Se observa en la Figura 4.2 que las predicciones del modelo ETS (línea verde) se alinean más estrechamente con la dirección de los valores reales (línea azul) hacia el final del período, capturando la tendencia y oscilaciones de los datos históricos. En contraste, el modelo Naive (linea naranja) es más sencillo y no intenta capturar patrones adicionales, lo que lo convierte en una referencia básica para evaluar el desempeño de modelos más avanzados. Esta comparación ilustra la importancia de considerar la complejidad y la estructura de la serie al seleccionar un modelo predictivo adecuado para escenarios específicos.

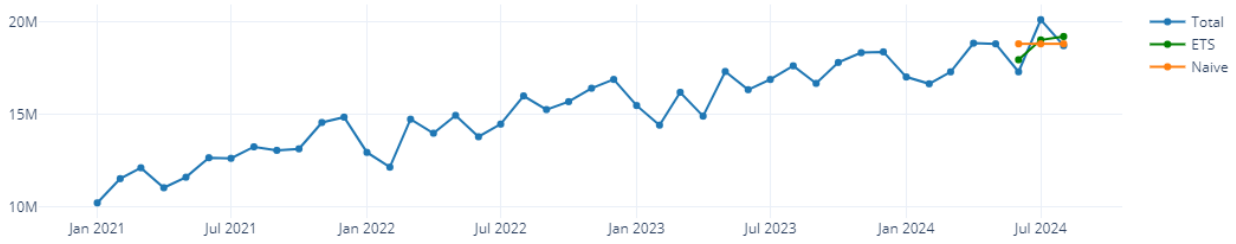


Figura 4.2 Comparación de Valores Predichos con el Modelo Base y el Modelo Avanzado (Elaboración Propia).

El análisis de los modelos detallado en la Tabla 4.1 muestra que el ETS, con un $R2$=0.5294, es capaz de explicar el 52.94% de la variabilidad de los datos, demostrando un desempeño moderado al capturar tendencias y estacionalidades de la serie temporal. En contraste, el modelo Naive obtuvo un $R2$=−0.0078, lo que indica que no logra explicar la variabilidad de los datos y tiene un rendimiento inferior incluso a un modelo trivial. Esto resalta que el ETS es más adecuado para series complejas, mientras que el Naive sirve únicamente como referencia básica.

Tabla 4.1 Comparación de Métrica de Rendimiento (Elaboración Propia).

Modelo	R^2
ETS	0.5294
Naive	-0.0078

En la comparación de los modelos ETS y Naive en la Tabla 4.2 para los meses de junio a agosto de 2024, se observa que el modelo ETS logra un mejor desempeño general al reducir los errores absolutos en dos de los tres meses evaluados, con valores más cercanos a los reales gracias a su capacidad para capturar tendencias y patrones en la serie. Sin embargo, en agosto de 2024, el modelo Naive presenta un menor error absoluto debido a que los valores reales están más cerca del último dato observado, resaltando su utilidad en escenarios con menor variabilidad. Esto confirma que el modelo ETS es más adecuado para series complejas, mientras que el Naive se limita a ser una referencia básica.

Tabla 4.2 Comparación de Predicciones del Valores Real, Modelo Base y Modelo Avanzado (Elaboración Propia).

Año - Mes	Valor Real	ETS	Naive
2024-06	17,299,736	17,898,940	18,809,455
2024-07	20,113,790	18,980,840	18,809,455
2024-08	18,710,862	19,236,020	18,809,455

El modelo avanzado demuestra ser superior al modelo base debido a su capacidad para capturar patrones complejos en series temporales, como tendencias y estacionalidades, mientras que el modelo Naive se limita a replicar el último valor observado. Esta diferencia se traduce en un desempeño consistentemente mejor del modelo avanzado, como se observa en su mayor precisión y menor error absoluto en la mayoría de los períodos evaluados. Además, el modelo avanzado es más robusto frente a cambios en la estructura de la serie, lo que lo convierte en una opción más confiable para predicciones en escenarios con mayor variabilidad y dinámicas complejas. Por tanto, la aplicación del modelo avanzado es más adecuada cuando se busca un modelo que explique mejor la variabilidad de los datos y ofrezca predicciones más precisas, mientras que el modelo base se limita a ser una herramienta básica de comparación.

4.2 Validación con un conjunto de datos independientes

La validación del modelo se llevó a cabo utilizando un conjunto de datos independientes correspondiente al mes de septiembre de 2024, lo que permite evaluar el desempeño de los modelos en un escenario real, fuera de los datos de entrenamiento y prueba. Este análisis comparó los valores predichos por un Modelo Avanzado y un Modelo Base (Naive) contra los valores reales para dos entidades financieras: Diners Club y Produbanco, como se muestra en la Tabla 4.3.

Para Diners Club en septiembre de 2024, la predicción del Modelo Base coincide exactamente con los valores reportados para los meses anteriores, reflejando su incapacidad para adaptarse a cambios en la dinámica de los datos. Sin embargo, esta simplicidad le permitió generar una predicción más cercana al valor real en el caso de Diners Club, con un error absoluto de 16,215, menor que el del Modelo Avanzado. Este resultado demuestra que, aunque limitado, el Modelo Base puede ser útil en escenarios con datos estables o predecibles.

En el caso de Produbanco, el Modelo Avanzado mostró un mejor desempeño al reducir el error absoluto en un 38% respecto al Modelo Base. Esto evidencia que el Modelo Avanzado logra capturar patrones o tendencias que el Modelo Base no puede.

Tabla 4.3 Comparación de Valores Reales con Valores Predichos para Fechas Independientes (Elaboración Propia).

Año - Mes	Entidad	Valor Real	Modelo Avanzado	Modelo Base
2024-09	Diners Club	5,967,879	5,927,902	5,984,094
2024-09	Produbanco	1,588,762	1,632,340	1,658,974

4.3 Análisis de Resultados de Predicciones

En el análisis de los valores de predicción observados en la Figura 4.3, se evidencia un potencial crecimiento moderado hacia los últimos meses del período analizado, lo que podría traducirse en beneficios significativos para las instituciones financieras emisoras de tarjetas de crédito. La proyección positiva en septiembre (1.0%) y el repunte en noviembre (6.8%) sugieren un aumento en la emisión de crédito, lo cual

fortalece la liquidez de las entidades y genera mayores ingresos derivados de intereses, comisiones y transacciones relacionadas con el uso de estas tarjetas.

Este crecimiento en la cantidad de emisión de tarjetas y, en consecuencia, en el crédito otorgado, contribuye a robustecer la base de depósitos del sistema financiero. Un mayor volumen de transacciones y pagos incrementa los fondos disponibles para cumplir con los requisitos de encaje exigidos por el Banco Central, fortaleciendo así la capacidad de las instituciones para mantener operaciones fluidas y cumplir con sus obligaciones regulatorias (Política y Regulación Monetaria, 2023).

Además, un aumento sostenido en la emisión de crédito mediante tarjetas representa una señal de recuperación económica y confianza tanto de los consumidores como de las instituciones financieras. Esto crea un círculo virtuoso en el que la expansión del crédito impulsa el consumo interno, lo que a su vez beneficia al sistema financiero y genera una mayor contribución al encaje bancario, garantizando la estabilidad y sostenibilidad del sistema.

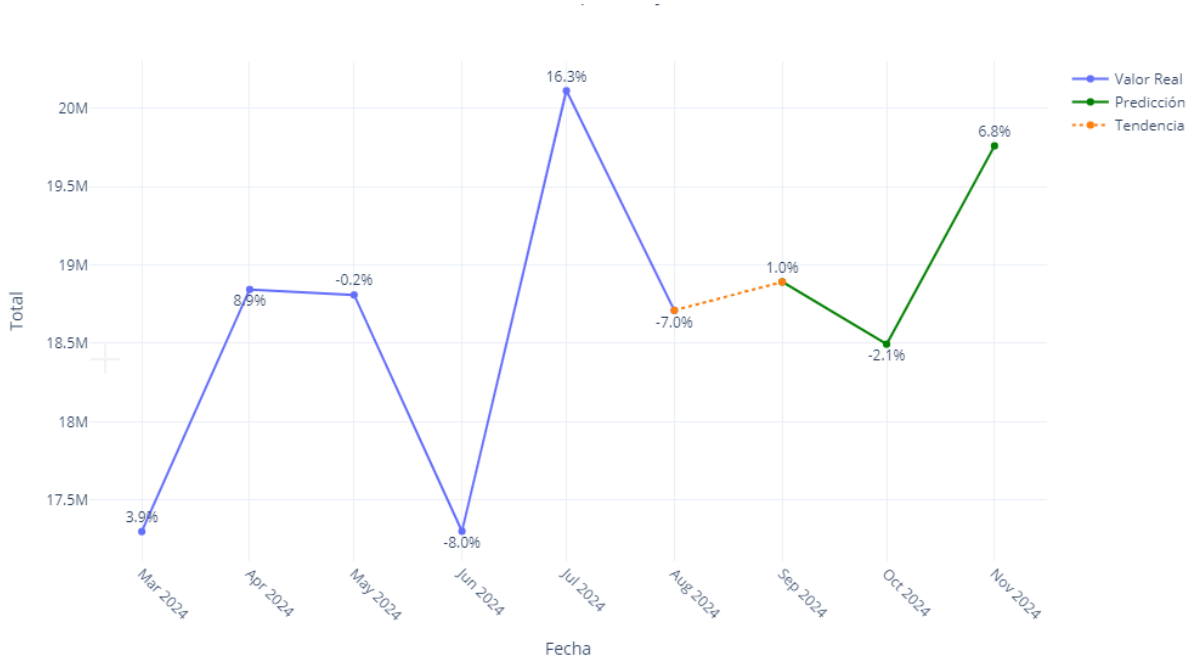


Figura 4.3. Evolución de la Cantidad de Tarjetas de Crédito Emitidas y Variación Porcentual Para el Sistema Financiero del Ecuador (Elaboración Propia).

4.3.1 Resultados de predicciones para Diners Club

Durante el periodo comprendido entre septiembre y noviembre de 2024, se observa en la Figura 4.4 un comportamiento diferenciado en la cantidad de tarjetas activas de Diners Club, caracterizado por dos meses consecutivos de contracción (septiembre y octubre) seguidos de una notable recuperación en noviembre. Este análisis destaca las fluctuaciones proyectadas, las causas potenciales y las implicaciones estratégicas para la entidad financiera en el contexto del mercado de tarjetas de crédito.

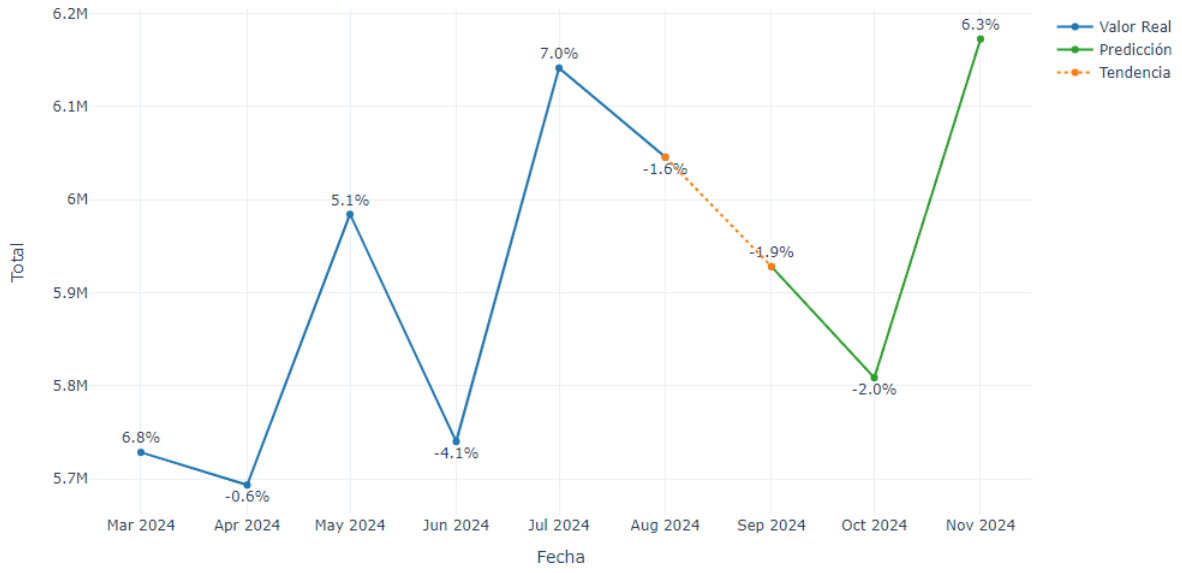


Figura 4.4. Evolución de la Cantidad de Tarjetas de Crédito Emitidas y Variación Porcentual para Diners Club (Elaboración Propia).

En la Tabla 4.4, el verde indica aumentos, el rojo disminuciones, y los tonos intermedios (amarillo/anaranjado) reflejan cambios moderados. Los valores predichos futuros están resaltados en azul celeste, diferenciándose de los históricos para facilitar su interpretación. Para el mes de septiembre de 2024, se proyecta una disminución del -1.945% en la cantidad de tarjetas activas. Este decrecimiento puede estar vinculado a factores operativos internos, como el bloqueo de tarjetas debido a incumplimientos de pago o medidas de seguridad, la caducidad de tarjetas sin renovación, y la cancelación voluntaria de cuentas por parte de clientes que perciben menor valor en el producto.

En octubre de 2024, la tendencia negativa continúa con una contracción ligeramente mayor del -2.014%, lo que representa el segundo mes consecutivo de disminución en la cantidad de tarjetas activas. Este comportamiento puede relacionarse, además de los factores operativos mencionados, con el impacto acumulativo del encaje bancario sobre las operaciones de la entidad. Un mayor encaje puede restringir la capacidad de Diners Club para otorgar líneas de crédito a nuevos clientes o ampliar los límites de crédito de los clientes existentes, generando una percepción de rigidez en los beneficios ofrecidos. Asimismo, la presión de liquidez podría limitar las promociones dirigidas a reactivar clientes inactivos, afectando la competitividad de las tarjetas frente a productos de otras instituciones financieras.

En noviembre de 2024, se proyecta una recuperación significativa del +6.264% en la cantidad de tarjetas activas, marcando un punto de inflexión en la tendencia observada. Este crecimiento puede explicarse por varios factores, como la reactivación de clientes que renovaron tarjetas caducadas en los meses anteriores, la influencia de la temporada alta de consumo (incluyendo eventos como el Black Friday y las compras navideñas), y el efecto positivo de estrategias de captación y retención aplicadas previamente.

Tabla 4.4 Calculo del Valor y Porcentaje de Variación entre Fechas para Diners Club (Elaboración Propia).

Fecha	Valor Absoluto de Predicción	% de Variación
01/04/2024	-35.068	-0,612
01/05/2024	290.812	5,108
01/06/2024	-244.099	-4,079
01/07/2024	401.170	6,989
01/08/2024	-95.677	-1,558
01/09/2024	-117.585	-1,945
01/10/2024	-119.373	-2,014
01/11/2024	363.835	6,264

4.3.2 Resultados de predicciones para Produbanco

El análisis del comportamiento de las tarjetas de crédito activas de Produbanco durante el periodo comprendido entre septiembre y noviembre de 2024 evidencia una dinámica de fluctuaciones marcada por un leve crecimiento en septiembre, una contracción en octubre y una recuperación significativa en noviembre como se muestra en la Figura 4.5. Este periodo pone de manifiesto no solo los retos propios de la gestión de portafolios financieros, sino también el impacto de una tendencia creciente: la resistencia de los consumidores hacia la aceptación y uso de tarjetas de crédito.

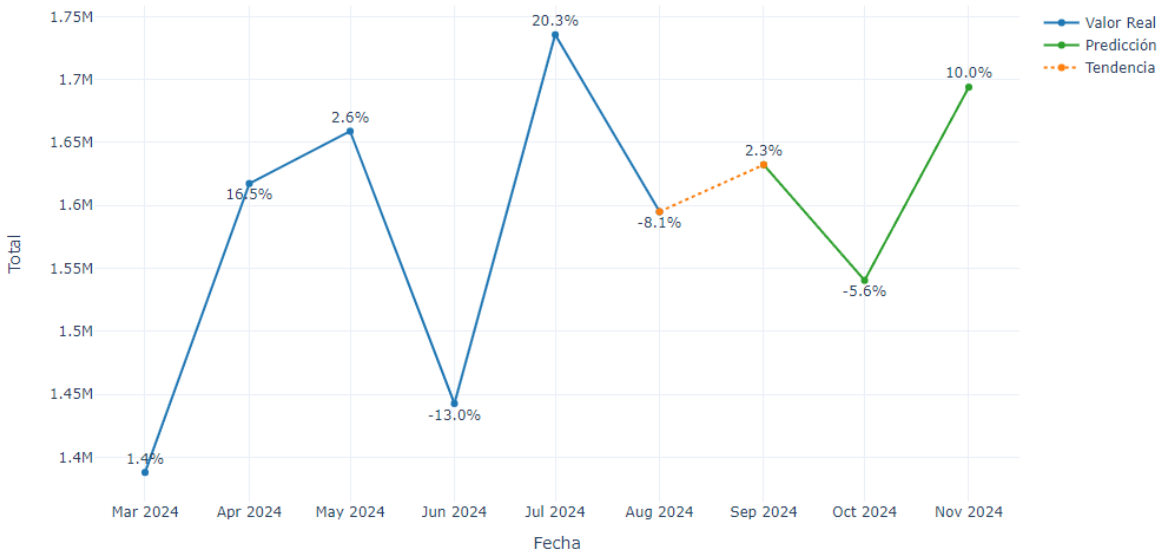


Figura 4.5. Evolución de la Cantidad de Tarjetas de Crédito Emitidas y Variación Porcentual para Produbanco (Elaboración Propia).

En la Tabla 4.5, el verde indica aumentos, el rojo disminuciones, y los tonos intermedios (amarillo/anaranjado) reflejan cambios moderados. Los valores predichos futuros están resaltados en azul celeste, diferenciándose de los históricos para facilitar su interpretación. Para septiembre de 2024, se registra un incremento del 2.34% en la cantidad de tarjetas activas. Aunque este crecimiento es moderado, refleja una recuperación leve en comparación con meses anteriores que presentaron contracciones significativas, como agosto (-8.10%). Las causas de este aumento pueden estar relacionadas con campañas específicas de renovación de tarjetas caducadas y estrategias promocionales que incentivaron el uso de las tarjetas. Sin embargo, este comportamiento positivo no necesariamente representa un cambio de

tendencia, ya que los consumidores continúan mostrando resistencia hacia las tarjetas de crédito debido a su percepción de alto costo, el auge de métodos de pago digitales y una desconfianza generalizada hacia el endeudamiento.

El panorama cambia en octubre de 2024, cuando Produbanco enfrentara una contracción del -5.63% en la cantidad de tarjetas activas, revirtiendo el leve crecimiento observado en septiembre. Este retroceso puede atribuirse a factores acumulativos, como bloqueos de tarjetas por incumplimiento de pagos, cancelaciones voluntarias y el impacto de alternativas financieras más modernas, como billeteras electrónicas y plataformas de pago digital. Además, factores macroeconómicos, como posibles ajustes en el encaje bancario, podrían haber restringido la capacidad del banco para ofrecer condiciones más competitivas. Este mes refleja un desafío estratégico para Produbanco, ya que el rechazo de los consumidores hacia las tarjetas de crédito sigue siendo un obstáculo importante.

En noviembre de 2024, se proyecta una recuperación significativa del 10.00% en la cantidad de tarjetas activas, marcando un punto de inflexión positivo en el desempeño de Produbanco. Este crecimiento puede explicarse por el efecto de la temporada alta de consumo, que incluye eventos como el Black Friday y las compras navideñas, donde los consumidores tienden a usar más sus tarjetas de crédito para aprovechar promociones y descuentos. Además, es probable que las estrategias de captación y fidelización implementadas en meses anteriores comiencen a reflejar resultados positivos en este periodo. Este comportamiento sugiere que, a pesar de la resistencia hacia las tarjetas de crédito, existe una demanda estacional que puede ser capitalizada mediante campañas estratégicas.

Tabla 4.5 Calculo del Valor y Porcentaje de Variación entre Fechas para Produbanco (Elaboración Propia).

Fecha	Valor Absoluto de Predicción	% de Variación
01/04/2024	229537	16,538654
01/05/2024	41555	2,569217
01/06/2024	-216352	-13,041313
01/07/2024	292987	20,30934
01/08/2024	-140573	-8,099347
01/09/2024	37304,2127	2,338769
01/10/2024	-91850,7824	-5,626939
01/11/2024	153473,0882	9,962619

En general, este análisis con la predicción de la cantidad de tarjetas de crédito a emitir, destaca tanto los desafíos como las oportunidades que enfrentan las entidades financieras al utilizar herramientas para evaluar el mercado de tarjetas de crédito Si bien septiembre y octubre presentan retos significativos debido a la contracción de la cartera activa y el rechazo de los consumidores hacia el producto, noviembre destaca como un mes de recuperación que puede ser aprovechado estratégicamente para fortalecer la relación con los clientes.

4.4 Análisis de sesgo

El análisis de sesgo es crucial para identificar y mitigar las posibles limitaciones que pueden influir en la calidad y precisión de los resultados obtenidos. A continuación, se describen los diferentes tipos de sesgo considerados en este proyecto:

4.4.1 Sesgo de los Datos

Los datos utilizados en este proyecto son de carácter público e incluyen exclusivamente a las entidades financieras supervisadas por la Superintendencia de Bancos del Ecuador. No se consideran cooperativas de ahorro y crédito, mutualistas u otras instituciones que están bajo la supervisión de la Superintendencia de Economía Popular y Solidaria. Esto se debe a que, para los fines de esta investigación, se ha optado por emplear únicamente la información proporcionada por la Superintendencia de Bancos, dado su rigor en la publicación regular de datos y la homogeneidad en las estructuras de información suministradas por las entidades financieras bajo su control.

4.4.2 Sesgo Temporal

A medida que el sector financiero evoluciona con el tiempo, la forma y estructura de los informes emitidos por las entidades han experimentado transformaciones significativas. El sitio web proporciona de manera clara y detallada los lineamientos necesarios para adaptarse a estos cambios y cumplir con las normativas actualizadas.

4.4.3 Sesgo de Estacionalidad

Se realizó un análisis de estacionalidad para mitigar el riesgo de sobreestimar patrones repetitivos o subestimar comportamientos erráticos e impredecibles. Este enfoque permite identificar tendencias estacionales relevantes sin dejar de lado posibles variaciones atípicas que puedan tener un impacto significativo en el análisis.

4.4.4 Sesgo Técnico

Se utiliza un análisis de estacionalidad para mitigar la presencia excesiva de patrones recurrentes, evitando así subestimar comportamientos erráticos o impredecibles que puedan surgir en los datos. Este enfoque busca equilibrar la identificación de tendencias estacionales sin comprometer la detección de variaciones atípicas.

4.4.5 Sesgo de Selección del Modelo

Todos los modelos se entrenaron bajo las mismas características y condiciones, y su selección se realizó considerando múltiples métricas, como R^2, MAE, entre otras. Además, se empleó validación cruzada para calcular promedios representativos y garantizar una evaluación más robusta. Finalmente, el modelo seleccionado se validó utilizando un conjunto de datos independiente, ajeno al proceso de construcción, para asegurar su capacidad de generalización.

4.5 Resultados Esperados

El pipeline desarrollado ofrece modelos específicos adaptados a las características de las series de tiempo de cada entidad financiera, proporcionando estimaciones útiles y confiables para la toma de decisiones estratégicas. Los tres modelos implementados, optimizados y evaluados presentan un desempeño satisfactorio, con coeficientes de determinación (R^2) de 0.5294, 0.7822 y 0.6127, respectivamente. En promedio, los modelos logran explicar el 64.14% de la variabilidad en los datos, lo

que demuestra su eficacia para generar predicciones precisas y alineadas con las necesidades de planificación operativa, gestión de riesgos y diseño de estrategias comerciales de las entidades financieras. Este enfoque garantiza que las instituciones puedan confiar en estas herramientas para optimizar recursos y maximizar la eficiencia en la emisión de tarjetas de crédito.

4.6 Análisis de costos y beneficio

Existen dos enfoques principales para la implementación del proyecto, cada uno con sus propias características, ventajas y desafíos a considerar como se describe en la Tabla 4.6.

El primer enfoque es la implementación on-premise, que implica desplegar la solución dentro de la infraestructura local de la organización. Este modelo es viable para el proyecto debido a que requiere un nivel moderado de procesamiento y almacenamiento, lo que permite utilizar hardware existente o realizar una inversión inicial en servidores dedicados. Además, el uso de herramientas libres y de código abierto, como Anaconda, MySQL y Streamlit, elimina la necesidad de adquirir licencias de software, reduciendo los costos a la inversión en hardware y su mantenimiento. Este enfoque ofrece ventajas significativas, como un mayor control sobre los datos sensibles, flexibilidad para personalizar el entorno según las necesidades específicas y la ausencia de costos recurrentes asociados al uso de servicios externos.

El segundo enfoque es la implementación en la nube, que aprovecha servicios contratados de almacenamiento y procesamiento en plataformas como AWS, Azure o Google Cloud. Este modelo es especialmente adecuado si se busca escalabilidad, flexibilidad y tiempos de implementación más rápidos. La nube permite ajustar los recursos de manera dinámica según las necesidades del proyecto, garantizando eficiencia en el uso de recursos y evitando grandes inversiones iniciales. Al optar por este modelo, se accede a servicios administrados que simplifican tareas como la gestión de bases de datos, el almacenamiento distribuido, la replicación automática de datos y el escalado automático para manejar aumentos en la carga de trabajo. Sin embargo, este enfoque tiene costos recurrentes basados en el consumo de

recursos, y puede haber consideraciones de dependencia del proveedor y cumplimiento con las políticas de privacidad de datos.

Tabla 4.6 Comparación de Costos de la Implementación de Proyecto (Elaboración Propia).

Componente	Costo Inicial (USD)	Costo Mensual (USD)	Observación
Hardware (Servidor)	$1,500 - $5,000	$0 (excepto mantenimiento)	Si se cuenta con servidor de propósito general, se puede reutilizar.
Backup y Redundancia	$500 - $2,000	$0	
Software (Open Source)	$0	$0	Considerar utilizar servicios en la nube.
Científico de Datos	$0	$1,000 - $3,000	Considerar todo el flujo, Ing de Datos.
Energía	$0	$20 - $50	
Desarrollo Web		$1,000 - $3,000	

La implementación de un modelo predictivo para prever la emisión de tarjetas de crédito representa una solución costo-efectiva que optimiza los recursos financieros de la entidad. Al proporcionar estimaciones precisas, permite reducir gastos asociados a estrategias de prueba y error, como campañas de marketing ineficaces o un exceso de inventario de tarjetas físicas. Además, el uso de herramientas de código abierto y tecnologías escalables minimiza significativamente los costos operativos iniciales y recurrentes, maximizando el retorno de inversión. Esta capacidad para anticipar la demanda y ajustar las estrategias operativas en función de las proyecciones asegura un uso más eficiente del presupuesto, favoreciendo la sostenibilidad y rentabilidad a largo plazo.

4.7 Conclusiones

- Mediante el uso de técnicas avanzadas de aprendizaje automático y herramientas tecnológicas, se desarrolla un pipeline automatizado diseñado para entrenar,

optimizar y evaluar modelos predictivos de series temporales. Este pipeline está orientado a satisfacer las necesidades de las entidades financieras, permitiéndoles proyectar con precisión la cantidad de tarjetas de crédito que emitirán en un período determinado, facilitando así la planificación estratégica y operativa.

- El enfoque de AutoML permitió automatizar el entrenamiento de múltiples modelos, evaluándolos mediante métricas específicas para seleccionar el más adecuado según las características de la serie de tiempo de cada entidad financiera. Este proceso garantiza la implementación del modelo con el mejor rendimiento, optimizando la precisión y la capacidad predictiva para satisfacer las necesidades particulares de cada institución.
- El prototipo ofrece al usuario final una experiencia interactiva y funcional mediante un tablero diseñado en Power BI. Este tablero permite explorar las características del conjunto de datos con filtros dinámicos por año, mes, entidad financiera, categorías, entre otros, proporcionando una visualización detallada de la información. Además, incluye la opción de seleccionar el modelo predictivo deseado (Diners Club, Produbanco o un modelo general) y ajustar, mediante un control deslizante, el número de períodos (meses) a predecir. Esta funcionalidad dota al usuario experto de una herramienta simple para generar predicciones personalizadas, optimizando la toma de decisiones estratégicas basadas en datos.

4.8 Recomendaciones

- Se recomienda emplear la automatización de la comparación de modelos mediante herramientas como PyCaret, ya que facilita la evaluación práctica de múltiples modelos predictivos y la comparación de métricas clave para seleccionar el más adecuado. Es esencial considerar parámetros como la estacionalidad y la validación cruzada con múltiples folds, ya que impactan directamente en la robustez y generalización de los resultados.
- Se recomienda evaluar la contratación de un servicio web proporcionado por la Superintendencia de Bancos que permita la obtención automática de datos, optimizando el proceso actualmente realizado mediante documentos Excel.
- Monitoreo continuo del rendimiento de los modelos y planificar un reentrenamiento periódico de los modelos conforme se disponga de nuevos datos, garantizando

que estos se mantengan actualizados, relevantes y capaces de adaptarse a posibles cambios en los patrones de las series temporales o en las condiciones del entorno.

BIBLIOGRAFÍA

Ali, M. (2020). *PyCaret: An open source, low-code machine learning library in Python*. https://www.pycaret.org

Angwin, J., Larson, J., Mattu, S., & Kirchner, L. (2016). Machine Bias: There's software used across the country to predict future criminals. And it's biased against blacks. *ProPublica*. https://www.propublica.org/article/machine-bias-risk-assessments-in-criminal-sentencing

Bhatore, S., Mohan, L., & Reddy, Y. R. (2020). Machine learning techniques for credit risk evaluation: a systematic literature review. *Journal of Banking and Financial Technology*, *4*(1), 111–138. https://doi.org/10.1007/s42786-020-00020-3

Boustani, N., Emrouznejad, A., Gholami, R., Despic, O., & Ioannou, A. (2023). Improving the predictive accuracy of the cross-selling of consumer loans using deep learning networks. *Annals of Operations Research*. https://doi.org/10.1007/s10479-023-05209-5

Bujlow, T., Riaz, T., & Pedersen, J. M. (2012). A method for classification of network traffic based on C5.0 machine learning algorithm. *2012 International Conference on Computing, Networking and Communications, ICNC'12*, 237 – 241. https://doi.org/10.1109/ICCNC.2012.6167418

Caruso, G., Gattone, S. A., Fortuna, F., & Di Battista, T. (2021). Cluster Analysis for mixed data: An application to credit risk evaluation. *Socio-Economic Planning Sciences*, *73*. https://doi.org/10.1016/j.seps.2020.100850

Chen, Y., Dong, G., Han, J., Wah, B. W., & Wang, J. (2002). Chapter 29 - Multi-Dimensional Regression Analysis of Time-Series Data Streams**The work was supported in part by grants from U.S. National Science Foundation, the University of Illinois, and Microsoft Research. In P. A. Bernstein, Y. E. Ioannidis, R. Ramakrishnan, & D. Papadias (Eds.), *VLDB '02: Proceedings of the 28th International Conference on Very Large Databases* (pp. 323–334). Morgan Kaufmann. https://doi.org/https://doi.org/10.1016/B978-155860869-6/50036-6

Deng, C., Yi, F., Li, X., Tang, J., & Sun, G. (2023). Performance Analysis of CHAID Algorithm for Accuracy. *Proceedings - 2023 International Conference on Pattern*

Recognition, Machine Vision and Intelligent Algorithms, PRMVIA 2023, 182 – 186. https://doi.org/10.1109/PRMVIA58252.2023.00036

Dickey, D., & Fuller, W. (1979). Distribution of the Estimators for Autoregressive Time Series With a Unit Root. *JASA. Journal of the American Statistical Association*, *74*. https://doi.org/10.2307/2286348

Gao, X., Wen, J., & Zhang, C. (2021). Forecasting of credit card default based on incremental random forest. *Journal of Physics: Conference Series*, *1828*(1). https://doi.org/10.1088/1742-6596/1828/1/012122

Grodzicki, D. (n.d.). *The Evolution of Competition in the Credit Card Market **. https://ssrn.com/abstract=4493211

Hemdan Ezz El-Din and Manjaiah, D. H. (2022). Anomaly Credit Card Fraud Detection Using Deep Learning. In A. and Z. N. Acharjya Debi Prasanna and Mitra (Ed.), *Deep Learning in Data Analytics: Recent Techniques, Practices and Applications* (pp. 207–217). Springer International Publishing. https://doi.org/10.1007/978-3-030-75855-4_12

Kwiatkowski, D., Phillips, P. C. B., Schmidt, P., & Shin, Y. (1992). Testing the null hypothesis of stationarity against the alternative of a unit root: How sure are we that economic time series have a unit root? *Journal of Econometrics*, *54*(1), 159–178. https://doi.org/https://doi.org/10.1016/0304-4076(92)90104-Y

Li, W., Wu, X., Sun, Y., & Zhang, Q. (2010). Credit card customer segmentation and target marketing based on data mining. *Proceedings - 2010 International Conference on Computational Intelligence and Security, CIS 2010*, 73–76. https://doi.org/10.1109/CIS.2010.23

Mavrogiorgos, K., Kiourtis, A., Mavrogiorgou, A., Menychtas, A., & Kyriazis, D. (2024). Bias in Machine Learning: A Literature Review. In *Applied Sciences (Switzerland)* (Vol. 14, Issue 19). Multidisciplinary Digital Publishing Institute (MDPI). https://doi.org/10.3390/app14198860

Mehrabi, N., Morstatter, F., Saxena, N., Lerman, K., & Galstyan, A. (2021). A Survey on Bias and Fairness in Machine Learning. *ACM Comput. Surv.*, *54*(6). https://doi.org/10.1145/3457607

Olson, R. S., Bartley, N., Urbanowicz, R. J., & Moore, J. H. (2016). Evaluation of a Tree-based Pipeline Optimization Tool for Automating Data Science. *Proceedings of the Genetic and Evolutionary Computation Conference 2016*, 485–492. https://doi.org/10.1145/2908812.2908918

Política y Regulación Monetaria. (2023). *Resolución Nro. JPRM-2023-026-M: Reformar la Regulación del Porcentaje de Encaje y Reservas de Liquidez de las Entidades de los Sectores Financieros Público, Privado y Popular y Solidario*. https://www.bce.fin.ec/images/SISTEMA_N_PAGOS/GestionRiesgo/JPRM-2023-026-M.pdf

Quispe, J. O. Q., Quispe, A. C. F., Calvo, N. C. L., & Toledo, O. C. (2024). Analysis and Selection of Multiple Machine Learning Methodologies in PyCaret for Monthly Electricity Consumption Demand Forecasting. *Materials Proceedings*, *18*(1). https://doi.org/10.3390/materproc2024018005

Rajamohamed, R., & Manokaran, J. (2018). Improved credit card churn prediction based on rough clustering and supervised learning techniques. *Cluster Computing*, *21*(1), 65–77. https://doi.org/10.1007/s10586-017-0933-1

Rodríguez del Águila, M. M., & Benítez-Parejo, N. (2011). Simple linear and multivariate regression models. *Allergologia et Immunopathologia*, *39*(3), 159–173. https://doi.org/https://doi.org/10.1016/j.aller.2011.02.001

Sargeant, H. (2023). Algorithmic decision-making in financial services: economic and normative outcomes in consumer credit. *AI and Ethics*, *3*(4), 1295–1311. https://doi.org/10.1007/s43681-022-00236-7

Singh, A., Singh, A., Aggarwal, A., & Chauhan, A. (2022). Design and Implementation of Different Machine Learning Algorithms for Credit Card Fraud Detection. *2022 International Conference on Electrical, Computer, Communications and Mechatronics Engineering (ICECCME)*, 1–6. https://doi.org/10.1109/ICECCME55909.2022.9988588

Song, Z., Zhang, Y., & King, I. (2023). Towards Fair Financial Services for All: A Temporal GNN Approach for Individual Fairness on Transaction Networks. *Proceedings of the 32nd ACM International Conference on Information and Knowledge Management*, 2331–2341. https://doi.org/10.1145/3583780.3615091

Steinberg, D. (2009). CART: Classification and Regression Trees. In *The Top Ten Algorithms in Data Mining*. https://doi.org/10.1201/9781420089653-17

Stewart, R. T. (2011). A profit-based scoring system in consumer credit: making acquisition decisions for credit cards. *Journal of the Operational Research Society*, *62*(9), 1719–1725. https://doi.org/10.1057/jors.2010.135

Superintendencia de Bancos de Ecuador. (2024). *Servicios Financieros y Tarjetas – Portal Estadístico*. https://www.superbancos.gob.ec/estadisticas/portalestudios/servicios-financieros/

Wu, Y., Meng, X., Zhang, J., He, Y., Romo, J. A., Dong, Y., & Lu, D. (2024). Effective LSTMs with seasonal-trend decomposition and adaptive learning and niching-based backtracking search algorithm for time series forecasting. *Expert Systems with Applications*, *236*, 121202. https://doi.org/https://doi.org/10.1016/j.eswa.2023.121202

Xu, T., & Qu, M. (2024). Novel embedding model predicting the credit card's default using neural network optimized by harmony search algorithm and vortex search algorithm. *Heliyon*, *10*(9). https://doi.org/10.1016/j.heliyon.2024.e30134

Zywicki, T. J. (2010). *THE ECONOMICS OF PAYMENT CARD INTERCHANGE FEES AND THE LIMITS OF REGULATION ICLE Financial Regulatory Program White Paper Series*. http://ssrn.com/abstract_id=1624002Electroniccopyavailableat:https://ssrn.com/abstract=1624002Electroniccopyavailableat:https://ssrn.com/abstract=1624002

APÉNDICES

APÉNDICE A - Código Fuente Pipeline de Solución.

APÉNDICE B - Código Fuente Aplicación Web de Prototipo.

APÉNDICE A

Codigo Fuente - Tesis

December 5, 2024

1 Pipeline de Prediccion - Series de Tiempo

1.0.1 Importar librerias necesarias

[1]:
```
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
from matplotlib.ticker import FuncFormatter
import matplotlib.pyplot as plt
import seaborn as sns
import torch
import torch.nn as nn
from sklearn.preprocessing import MinMaxScaler
import numpy as np
from sklearn.metrics import mean_squared_error
from sklearn.metrics import mean_absolute_error
import matplotlib.ticker as ticker
import pycaret

#import pycaret time series and init setup
from pycaret.time_series import *
```

2 Entrada de Conjunto de Datos

[2]:
```
# Leer el archivo Excel
file_path = 'datos/ConsumosTarjetasConsolidado2021.xlsx'
file_path2 = 'datos/ConsumosTarjetasConsolidado2022.xlsx'
file_path3 = 'datos/ConsumosTarjetasConsolidado2023.xlsx'
file_path4 = 'datos/ConsumosTarjetasConsolidadoAGOSTO2024.xlsx'


df2021 = pd.read_excel(file_path, sheet_name='Tarjetas - Diciembre 2021',␣
 ↪header=None,skiprows=13)
df2022 = pd.read_excel(file_path2, sheet_name='Tabla', header=None,skiprows=18)
df2023 = pd.read_excel(file_path3, sheet_name='Data', header=0)
df2024 = pd.read_excel(file_path4, sheet_name='Data', header=0)
```

[3]:
```python
def aplanar_excel (df,nombre_file):
    # Crear listas para almacenar los datos procesados
    entidades = []
    marcas = []
    meses = []
    segmentos = []
    tipos = []
    totales = []

    # Variables auxiliares para almacenar la entidad, marca y mes actuales
    current_entidad = None
    current_marca = None
    current_segmento = None
    current_tipo = None

    # Procesar fila por fila
    for i, row in df.iterrows():
        # Si la fila indica una entidad financiera
        if not pd.isna(row[0]):
            current_entidad = row[0]

        # Si la fila indica una marca de tarjeta
        if not pd.isna(row[1]):
            current_marca = row[1]

        # Si la fila indica un mes
        if not pd.isna(row[2]):
            current_segmento = row[2]

        # Si la fila indica un mes
        if not pd.isna(row[3]):
            current_tipo = row[3]

        # Si la fila tiene un segmento de tarjeta y total
        if not pd.isna(row[4]):
            mes = row[4]
            total = row[5]

            # Agregar los datos a las listas
            entidades.append(current_entidad)
            marcas.append(current_marca)
            segmentos.append(current_segmento)
            tipos.append(current_tipo)
            meses.append(mes)
            totales.append(total)

    # Crear un DataFrame a partir de las listas
```

```
    df_processed = pd.DataFrame({
        'Emisor': entidades,
        'Marca Tarjeta': marcas,
        'Segmento':segmentos ,
        'Tipo Tarjeta':tipos ,
        'Mes': meses,
        'Total': totales
    })


    # Guardar el DataFrame procesado a un nuevo archivo  CSV
    # df_processed.to_csv('datos/aplanar_'+nombre_file+'.csv', index=False)
    return df_processed
```

[4]:
```
#procesar archivo 2021
nombre=file_path.split('/')[1].split('.')[0]
dftc2021=aplanar_excel (df2021,nombre)
dftc2021['Anio']='2021'
```

[5]:
```
#procesar archivo 2022
nombre=file_path2.split('/')[1].split('.')[0]
dftc2022=aplanar_excel (df2022,nombre)
dftc2022['Anio']='2022'
```

[6]:
```
#procesar archivo 2023
dftc2023=df2023[['Emisor','Marca Tarjeta','Segmento','Tipo␣
↪Tarjeta','Mes','Número de transacciones','Año']]
dftc2023=dftc2023.rename(columns={'Número de transacciones': 'Total','Año':␣
↪'Anio'})
```

[7]:
```
#procesar archivo 2024
dftc2024=df2024[['Emisor','Marca Tarjeta','Segmento','Tipo␣
↪Tarjeta','Mes','Número de transacciones','Año']]
dftc2024=dftc2024.rename(columns={'Número de transacciones': 'Total','Año':␣
↪'Anio'})
```

[8]:
```
dftc2024
```

[8]:

	Emisor	Marca Tarjeta	Segmento
0	Banco Amazonas S.A	Otras marcas	D
1	Banco Amazonas S.A	Visa	C+
2	Banco Amazonas S.A	Visa	C+
3	Banco Amazonas S.A	Visa	C+
4	Banco Amazonas S.A	Visa	C+
…	…	…	…
14790	Banco Visionfund Ecuador S.A	Visa	NaN
14791	BanEcuador B.P	Mastercard	NaN

```
14792                BanEcuador B.P    Mastercard      NaN
14793                BanEcuador B.P    Mastercard      NaN
14794                BanEcuador B.P    Mastercard      NaN

             Tipo Tarjeta     Mes  Total  Anio
0      Tarjeta de crédito   enero      2  2024
1      Tarjeta de crédito   enero    181  2024
2      Tarjeta de crédito   enero    149  2024
3      Tarjeta de crédito   enero     35  2024
4      Tarjeta de crédito   enero      3  2024
…                     …      …    …    …
14790   Tarjeta de débito  agosto   1061  2024
14791   Tarjeta de débito  agosto   6559  2024
14792   Tarjeta de débito  agosto    402  2024
14793   Tarjeta de débito  agosto   2093  2024
14794   Tarjeta de débito  agosto  12671  2024

[14795 rows x 7 columns]
```

[9]:
```
#Unir todos los años en solo DF
df_tc = pd.concat([dftc2021, dftc2022,dftc2023,dftc2024], axis=0)  # axis=0␣
 ↪significa que se concatenan filas
```

[10]:
```
# Nombres de Mes - homogeneos
df_tc['Mes']=df_tc['Mes'].str.strip().str.lower().str.title()
```

[11]:
```
df_tc['Mes'].value_counts()
```

[11]:
```
Mes
Agosto        4038
Mayo          4023
Junio         4006
Julio         4006
Abril         3999
Enero         3875
Marzo         3871
Febrero       3826
Septiembre    2178
Octubre       2172
Noviembre     2165
Diciembre     2160
Name: count, dtype: int64
```

[12]:
```
# Diccionario de mapeo para homologar los nombres
homologacion = {
    'Banco Pichincha C.A': 'Banco Pichincha',
    'Banco de la Producción S.A Produbanco': 'Produbanco',
```

```
    'Banco Diners Club del Ecuador S.A': 'Diners Club',
    'Banco Bolivariano C.A': 'Banco Bolivariano',
    'Banco del Pacífico S.A': 'Banco del Pacífico',
    'Banco General Rumiñahui  S.A': 'Banco Rumiñahui',
    'Banco General Rumiñahui S.A': 'Banco Rumiñahui',
    'Banco del Austro S.A': 'Banco del Austro',
    'Banco Internacional S.A': 'Banco Internacional',
    'Banco de Machala S.A': 'Banco de Machala',
    'Banco Comercial de Manabi S.A': 'Banco Comercial de Manabí',
    'Banco Comercial de Manabí': 'Banco Comercial de Manabí',
    'Banco Produbanco Grupo Promerica': 'Produbanco',
    'Banco Guayaquil S.A': 'Banco Guayaquil',
    'Banco de Guayaquil':  'Banco Guayaquil',
    'Banco Procredit S.A':  'Banco Procredit',
    'Banco Solidario S.A':  'Banco Solidario',
    'Banco Amazonas S.A':  'Banco Amazonas',
    'Banco de Loja S.A':  'Banco de Loja',
    'Banco General Rumiñahui':  'Banco Rumiñahui'
}

df_tc['Emisor'] = df_tc['Emisor'].replace(homologacion)
```

[13]:
```
segmento_counts = df_tc['Emisor'].value_counts()

# Mostrar los resultados
print(segmento_counts)
```

```
Emisor
Banco Pichincha                                  6579
Produbanco                                       5409
Diners Club                                      4756
Banco Bolivariano                                4574
Banco del Pacífico                               4561
Banco del Austro                                 2734
Banco Rumiñahui                                  2272
Banco Internacional                              2256
Banco Guayaquil                                  1820
Banco de Machala                                 1378
Banco Amazonas                                   1166
Banco Solidario                                  1117
Banco de Loja                                     924
Banco Comercial de Manabí                         280
Banco Procredit                                   153
Banco Desarrollo de los Pueblos S.A. Codesarrollo  91
Banco del Litoral S.A                              80
BanEcuador B.P                                     52
Banco Visionfund Ecuador S.A                       40
Banco Amibank S.A                                  38
```

```
Banco D-miro S.A                                20
Banco Dmiro                                     12
Banco del Litoral                                4
Banco Desarrollo de los Pueblos                  3
Name: count, dtype: int64
```

[14]:
```python
# Obtener el conteo de cada valor en Segmento de la tarjeta
segmento_counts = df_tc['Marca Tarjeta'].value_counts()

# Mostrar los resultados
print(segmento_counts)
```

```
Marca Tarjeta
Visa                            21748
Mastercard                      13154
Diners Club                      1913
Alía                              981
American Express                  679
Discover                          660
UnionPay                          437
Tarjetas de Sistemas Cerrados     340
Alia                              128
Otras marcas                      103
Union Pay                          74
Discover                           72
Alia/Cuota Fácil                   16
Cuota Fácil                        13
0                                   1
Name: count, dtype: int64
```

[15]:
```python
# Obtener el conteo de cada valor del segmento
segmento_counts = df_tc['Segmento'].value_counts()

# Mostrar los resultados
print(segmento_counts)
```

```
Segmento
C+     7758
B+     6838
D+     5839
AA+    5086
C      3148
D      2817
A+     2480
E      1570
AA     1340
B       977
A       485
E+      238
```

```
Name: count, dtype: int64
```

[16]:
```
# Filtrar el DataFrame para eliminar valores en blanco o nulos en 'Segmento de la tarjeta'
df_tc = df_tc[df_tc['Segmento'].notna() & (df_tc['Segmento'] != '(en blanco)')]

# Obtener el conteo de cada valor en 'Segmento de la tarjeta' nuevamente
segmento_counts = df_tc['Segmento'].value_counts()

# Mostrar los resultados
print(segmento_counts)
```

```
Segmento
C+     7758
B+     6838
D+     5839
AA+    5086
C      3148
D      2817
A+     2480
E      1570
AA     1340
B       977
A       485
E+      238
Name: count, dtype: int64
```

[17]:
```
df_tc['Tipo Tarjeta'].value_counts()
```

[17]:
```
Tipo Tarjeta
Tarjeta de crédito    38329
Tarjeta de débito       185
Tarjeta prepago          62
Name: count, dtype: int64
```

[18]:
```
#Vamos a utilizar solo TC
df_tc=df_tc[df_tc['Tipo Tarjeta']=='Tarjeta de crédito']
df_tc
```

[18]:
```
                Emisor Marca Tarjeta Segmento        Tipo Tarjeta      Mes  \
12         Diners Club   Diners Club       A+  Tarjeta de crédito    Enero
13         Diners Club   Diners Club       A+  Tarjeta de crédito  Febrero
14         Diners Club   Diners Club       A+  Tarjeta de crédito    Marzo
15         Diners Club   Diners Club       A+  Tarjeta de crédito    Abril
16         Diners Club   Diners Club       A+  Tarjeta de crédito     Mayo
…                  …             …      …                   …      …
14782  Banco Solidario          Alía       D+  Tarjeta de crédito   Agosto
14783  Banco Solidario          Alía       D+  Tarjeta de crédito   Agosto
```

```
14784  Banco Solidario          Alía       D+  Tarjeta de crédito   Agosto
14785  Banco Solidario          Alía       D+  Tarjeta de crédito   Agosto
14786  Banco Solidario          Alía       D+  Tarjeta de crédito   Agosto

       Total  Anio
12     83681  2021
13     72585  2021
14     80254  2021
15     73573  2021
16     76015  2021
…      …    …
14782   1675  2024
14783      1  2024
14784  14390  2024
14785   2307  2024
14786   4307  2024

[38329 rows x 7 columns]
```

[19]:
```
df_emisor_agrupado=df_tc.groupby(['Emisor','Anio','Mes'])['Total'].sum().
↪reset_index()
```

[20]:
```
df_emisor_agrupado[df_emisor_agrupado['Emisor']=='Banco Pichincha']
```

[20]:

	Emisor	Anio	Mes	Total
220	Banco Pichincha	2023	Abril	2331948
221	Banco Pichincha	2023	Agosto	3208838
222	Banco Pichincha	2023	Diciembre	3295645
223	Banco Pichincha	2023	Enero	2426988
224	Banco Pichincha	2023	Febrero	2253880
225	Banco Pichincha	2023	Julio	3039645
226	Banco Pichincha	2023	Junio	2905346
227	Banco Pichincha	2023	Marzo	2578632
228	Banco Pichincha	2023	Mayo	3146365
229	Banco Pichincha	2023	Noviembre	3326788
230	Banco Pichincha	2023	Octubre	3246904
231	Banco Pichincha	2023	Septiembre	3053500
232	Banco Pichincha	2024	Abril	3361609
233	Banco Pichincha	2024	Agosto	3376290
234	Banco Pichincha	2024	Enero	3135782
235	Banco Pichincha	2024	Febrero	2983973
236	Banco Pichincha	2024	Julio	3924316
237	Banco Pichincha	2024	Junio	2891370
238	Banco Pichincha	2024	Marzo	3138028
239	Banco Pichincha	2024	Mayo	3499944
240	Banco Pichincha	2021	Abril	1872752
241	Banco Pichincha	2021	Agosto	2253517

```
242  Banco Pichincha  2021   Diciembre  2532638
243  Banco Pichincha  2021       Enero  1787445
244  Banco Pichincha  2021     Febrero  2873356
245  Banco Pichincha  2021       Julio  2183933
246  Banco Pichincha  2021       Junio  2175501
247  Banco Pichincha  2021       Marzo  2019490
248  Banco Pichincha  2021        Mayo  1641248
249  Banco Pichincha  2021   Noviembre  2503053
250  Banco Pichincha  2021     Octubre  2243823
251  Banco Pichincha  2021  Septiembre  2260458
252  Banco Pichincha  2022       Abril  2577245
253  Banco Pichincha  2022      Agosto  2502281
254  Banco Pichincha  2022   Diciembre  2619740
255  Banco Pichincha  2022       Enero  2245811
256  Banco Pichincha  2022     Febrero  2057239
257  Banco Pichincha  2022       Julio  2320826
258  Banco Pichincha  2022       Junio  2167602
259  Banco Pichincha  2022       Marzo  2272793
260  Banco Pichincha  2022        Mayo  2374486
261  Banco Pichincha  2022   Noviembre  2552827
262  Banco Pichincha  2022     Octubre  2473665
263  Banco Pichincha  2022  Septiembre  2350978
```

[21]:
```
df_emisor_agrupado.to_csv('datos/entidades_agrupadas.csv', index=False)
```

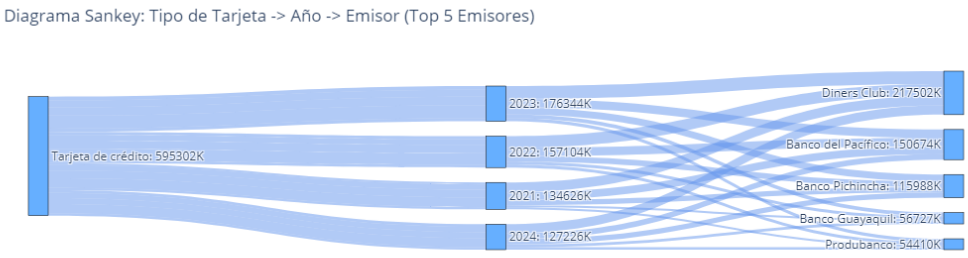
3 Análisis exploratorio de los datos

3.0.1 Grafico: sankey para la distrbucion de TC entre año y emisor

[22]:
```
import pandas as pd
import plotly.graph_objects as go

# Filtrar los 5 emisores más grandes por el total transaccionado
top_emisores = df_tc.groupby('Emisor')['Total'].sum().nlargest(5).index
df_top = df_tc[df_tc['Emisor'].isin(top_emisores)]

# Agrupar por Tipo de Tarjeta, Año y Emisor para sumar los totales
df_agrupado = df_top.groupby(['Tipo Tarjeta', 'Anio', 'Emisor'])['Total'].sum().
 ↪reset_index()

# Calcular el total por Tipo de Tarjeta para obtener los porcentajes
totales_tarjeta = df_agrupado.groupby('Tipo Tarjeta')['Total'].transform('sum')
df_agrupado['Porcentaje'] = df_agrupado['Total'] / totales_tarjeta * 100

# Crear las listas de nodos y enlaces para el diagrama Sankey
tarjetas = df_agrupado['Tipo Tarjeta'].unique().tolist()
```

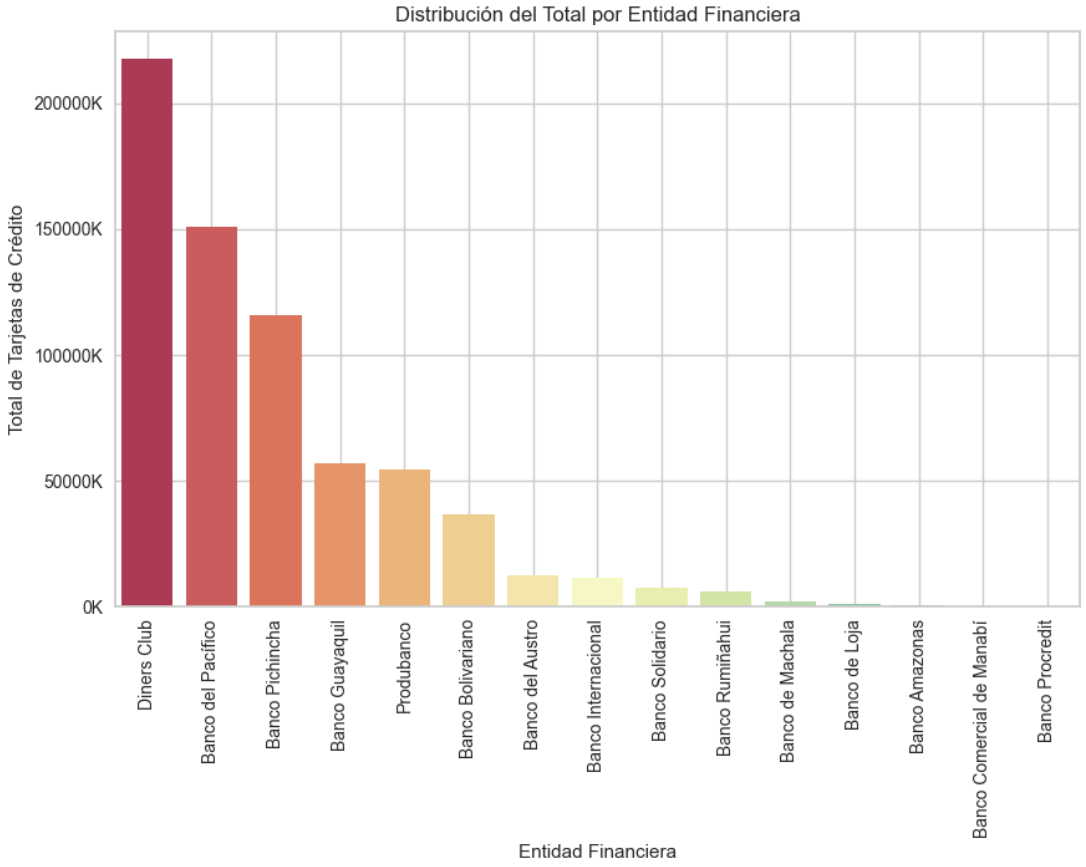
```
anios = df_agrupado['Anio'].astype(str).unique().tolist()  # Convertimos años a␣
 ↪string para los nodos
emisores = df_agrupado['Emisor'].unique().tolist()

# Crear nodos para Tipo de Tarjeta, Años y Emisores
nodos = tarjetas + anios + emisores

# Crear un diccionario de índices para los nodos
nodos_indices = {nodo: i for i, nodo in enumerate(nodos)}

# Preparar los flujos (links) para el diagrama Sankey
source = (
    df_agrupado['Tipo Tarjeta'].map(nodos_indices).tolist() +  # De Tipo de␣
 ↪Tarjeta a Año
    df_agrupado['Anio'].astype(str).map(nodos_indices).tolist()  # De Año a␣
 ↪Emisor
)
target = (
    df_agrupado['Anio'].astype(str).map(nodos_indices).tolist() +  # De Tipo de␣
 ↪Tarjeta a Año
    df_agrupado['Emisor'].map(nodos_indices).tolist()  # De Año a Emisor
)
values = (df_agrupado['Total'] / 1_000).tolist() * 2  # Convertimos los valores␣
 ↪a miles

# Crear etiquetas de nodos mostrando valores en miles sin decimales
node_labels = [
    f"{nodo}: {int(df_agrupado[df_agrupado['Tipo Tarjeta'] == nodo]['Total'].
 ↪sum() / 1_000)}K"
    if nodo in tarjetas else
    f"{nodo}: {int(df_agrupado[df_agrupado['Anio'].astype(str) ==␣
 ↪nodo]['Total'].sum() / 1_000)}K"
    if nodo in anios else
    f"{nodo}: {int(df_agrupado[df_agrupado['Emisor'] == nodo]['Total'].sum() /␣
 ↪1_000)}K"
    for nodo in nodos
]

# Crear texto personalizado para hover labels con porcentaje y valor en miles␣
 ↪sin decimales
hover_text = (
    [f"{row['Tipo Tarjeta']} → {row['Anio']}: {int(row['Total'] / 1_000)}K␣
 ↪({row['Porcentaje']:.2f}%)"
     for _, row in df_agrupado.iterrows()] +
    [f"{row['Anio']} → {row['Emisor']}: {int(row['Total'] / 1_000)}K␣
 ↪({row['Porcentaje']:.2f}%)"
```

```
    for _, row in df_agrupado.iterrows()]
)

# Crear el diagrama Sankey con valores en miles y hover personalizado
fig = go.Figure(go.Sankey(
    node=dict(
        pad=15,
        thickness=20,
        line=dict(color="black", width=0.5),
        label=node_labels,  # Nodos con etiquetas visibles y totales en miles␣
 ↪sin decimales
        color="rgba(0, 123, 255, 0.6)"  # Color de los nodos
    ),
    link=dict(
        source=source,  # Nodo fuente
        target=target,  # Nodo destino
        value=values,   # Valor del flujo en miles
        color="rgba(100, 149, 237, 0.5)",  # Color de los enlaces
        customdata=hover_text,  # Datos personalizados para hover
        hovertemplate='%{customdata}<extra></extra>',  # Mostrar texto␣
 ↪personalizado en hover
    )
))

# Título del gráfico
fig.update_layout(
    title_text="Diagrama Sankey: Tipo de Tarjeta -> Año -> Emisor (Top 5␣
 ↪Emisores)",
    font_size=12
)

# Mostrar el diagrama
fig.show()
```



3.0.2 Grafico:Distrbucion total entidad

[23]:
```
# Función para formatear los valores del eje Y en miles
def thousands_formatter(x, pos):
    return f'{int(x/1000)}K'

# Ordenar el DataFrame por el total agregado por cada "Entidad financiera"
df_ordered = df_tc.groupby('Emisor')['Total'].sum().reset_index().
 ↪sort_values(by='Total', ascending=False)

plt.figure(figsize=(10, 6))
sns.barplot(data=df_ordered, x='Emisor', y='Total', palette='Spectral')
plt.title('Distribución del Total por Entidad Financiera')
plt.ylabel('Total de Tarjetas de Crédito')
plt.xlabel('Entidad Financiera')
plt.xticks(rotation=90)
plt.gca().yaxis.set_major_formatter(FuncFormatter(thousands_formatter))  #␣
 ↪Aplicar formateo en millones
plt.grid(True)
plt.savefig("graficos/distribucion_total_entidades.png", bbox_inches='tight',␣
 ↪format='png')
plt.show()
```



[24]:
```
# Eliminar las entidades que no tiene una gran produccion, para analizar las␣
 ↪que tienen mas impactos
df_tc = df_tc[~df_tc['Emisor'].isin(['Banco Procredit', 'Banco Amazonas','Banco␣
 ↪Comercial de Manabí','Banco Desarrollo de los Pueblos','Banco del␣
 ↪Litoral','Banco Visionfund','Banco Dmiro','Banco Finca'])]

# Verificar las primeras filas del nuevo DataFrame filtrado
df_tc.head()
```

[24]:
```
         Emisor Marca Tarjeta Segmento        Tipo Tarjeta      Mes  Total  \
12  Diners Club   Diners Club       A+  Tarjeta de crédito    Enero  83681
13  Diners Club   Diners Club       A+  Tarjeta de crédito  Febrero  72585
14  Diners Club   Diners Club       A+  Tarjeta de crédito    Marzo  80254
15  Diners Club   Diners Club       A+  Tarjeta de crédito    Abril  73573
16  Diners Club   Diners Club       A+  Tarjeta de crédito     Mayo  76015

    Anio
12  2021
13  2021
14  2021
15  2021
16  2021
```

[25]:
```
df_tc
```

[25]:
```
                Emisor Marca Tarjeta Segmento        Tipo Tarjeta      Mes  \
12         Diners Club   Diners Club       A+  Tarjeta de crédito    Enero
13         Diners Club   Diners Club       A+  Tarjeta de crédito  Febrero
14         Diners Club   Diners Club       A+  Tarjeta de crédito    Marzo
15         Diners Club   Diners Club       A+  Tarjeta de crédito    Abril
16         Diners Club   Diners Club       A+  Tarjeta de crédito     Mayo
…                  …             …      …                   …      …
14782  Banco Solidario          Alía       D+  Tarjeta de crédito   Agosto
14783  Banco Solidario          Alía       D+  Tarjeta de crédito   Agosto
14784  Banco Solidario          Alía       D+  Tarjeta de crédito   Agosto
14785  Banco Solidario          Alía       D+  Tarjeta de crédito   Agosto
14786  Banco Solidario          Alía       D+  Tarjeta de crédito   Agosto

       Total  Anio
12     83681  2021
13     72585  2021
14     80254  2021
15     73573  2021
16     76015  2021
```

```
…       …   …
14782   1675  2024
14783      1  2024
14784  14390  2024
14785   2307  2024
14786   4307  2024

[37018 rows x 7 columns]
```

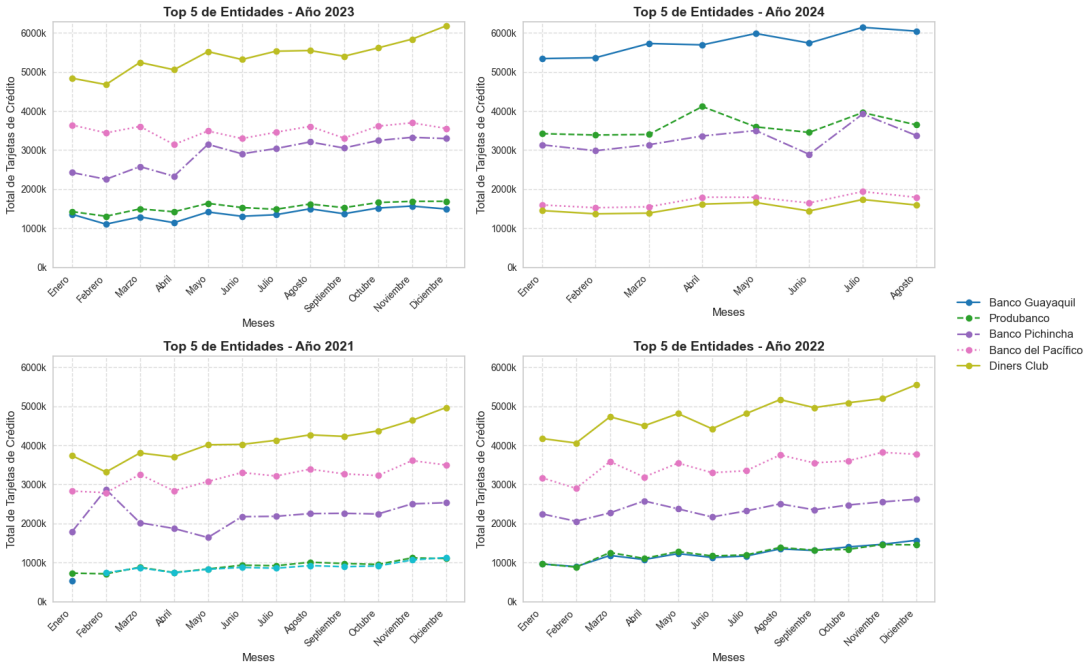
#df_tc.to_csv('datos/final_todasEntidades.csv', index=False)

3.1 Grafico: Top 5 entidades

[26]:
```
# Get the top 5 entities by year and month with the highest total value
grouped_entidades = df_tc.groupby(['Anio', 'Mes', 'Emisor'],␣
 ↪as_index=False)['Total'].sum()

# For each combination of year and month, we will get the top 5 entities by␣
 ↪total value
top_5_per_year_month = grouped_entidades.groupby(['Anio', 'Mes']).apply(lambda␣
 ↪x: x.nlargest(5, 'Total')).reset_index(drop=True)
```

[27]:
```
import pandas as pd
import matplotlib.pyplot as plt
import matplotlib.ticker as ticker

# Define the month order
month_order = ['Enero', 'Febrero', 'Marzo', 'Abril', 'Mayo', 'Junio', 'Julio',␣
 ↪'Agosto', 'Septiembre', 'Octubre', 'Noviembre', 'Diciembre']

# Adjust the legend position to be on the right of the charts
fig, axes = plt.subplots(nrows=2, ncols=2, figsize=(14, 10))
axes = axes.flatten()

# Ensure unique years and entities are used from the data
unique_years = top_5_per_year_month['Anio'].unique()
unique_entities = top_5_per_year_month['Emisor'].nunique()

# Generate a colormap and linestyles dynamically based on the number of unique␣
 ↪entities
colors = plt.cm.get_cmap('tab10', unique_entities).colors
linestyles = ['-', '--', '-.', ':'] * (unique_entities // 4 + 1)

# Calculate the maximum y-value for consistent y-axis limits across all plots
y_max = top_5_per_year_month['Total'].max()  # Calculate maximum value for the␣
 ↪y-axis
```

```
for i, year in enumerate(unique_years):
    ax = axes[i]
    year_data = top_5_per_year_month[top_5_per_year_month['Anio'] == year]
    year_data['Mes'] = pd.Categorical(year_data['Mes'], categories=month_order,␣
 ↪ordered=True)
    year_data = year_data.sort_values(by='Mes')

    for j, entity in enumerate(year_data['Emisor'].unique()):
        entity_data = year_data[year_data['Emisor'] == entity]
        ax.plot(entity_data['Mes'], entity_data['Total'], label=entity,␣
 ↪color=colors[j], linestyle=linestyles[j], marker='o')

    ax.set_title(f"Top 5 de Entidades - Año {year}", fontsize=14, weight='bold')
    ax.set_xlabel('Meses', fontsize=12)
    ax.set_ylabel('Total de Tarjetas de Crédito', fontsize=12)
    ax.set_xticklabels(month_order, rotation=45, ha='right')  # Use month_order␣
 ↪for consistent x-ticks
    ax.set_ylim(0, y_max + 100000)  # Ensure a bit more space at the top of the␣
 ↪y-axis
    ax.grid(True, linestyle='--', alpha=0.7)
    ax.yaxis.set_major_formatter(ticker.FuncFormatter(lambda x, pos: f'{int(x/
 ↪1000)}k'))

# Fetch the legend handles and labels from the last axis to use in the␣
 ↪consolidated legend
handles, labels = ax.get_legend_handles_labels()

# Consolidate the legend on the right
fig.legend(handles, labels, loc='center left', bbox_to_anchor=(1, 0.5),␣
 ↪fontsize=12)

plt.tight_layout()
plt.savefig("graficos/top_5_entities_chart_with_legend.png",␣
 ↪bbox_inches='tight', format='png')

# Show the plot
plt.show()
```



[28]:
```
grouped_entidades
```

[28]:

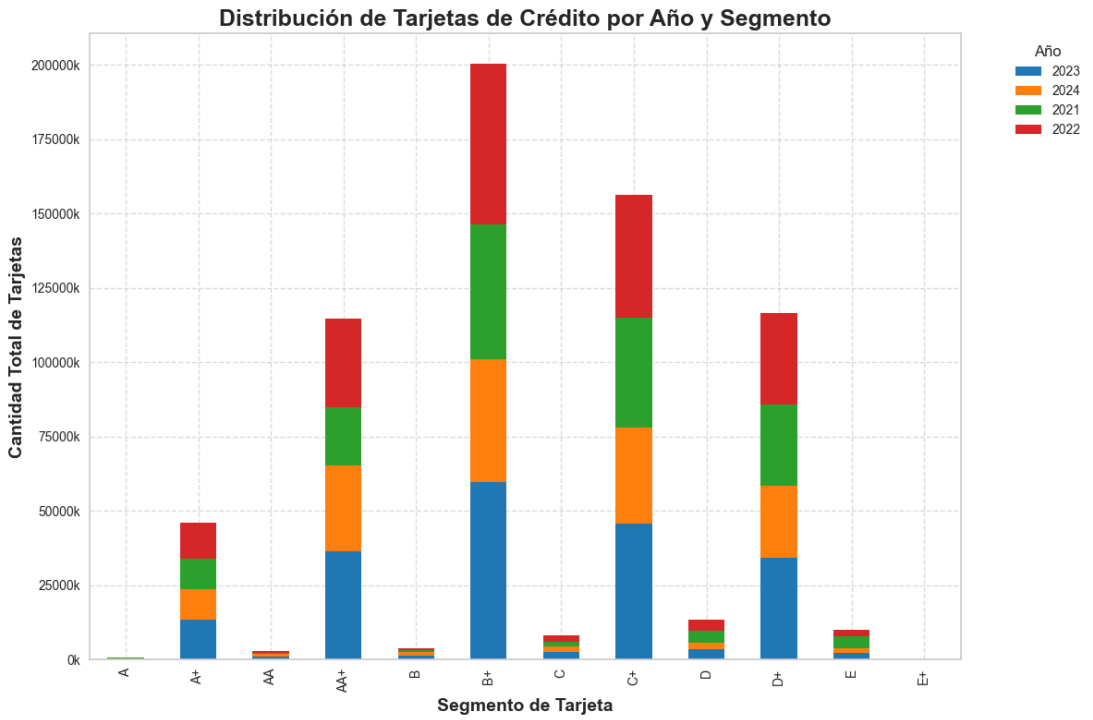
	Anio	Mes	Emisor	Total
0	2023	Abril	Banco Bolivariano	857069
1	2023	Abril	Banco Guayaquil	1422225
2	2023	Abril	Banco Internacional	262600
3	2023	Abril	Banco Pichincha	2331948
4	2023	Abril	Banco Rumiñahui	144571
..	…	…	…	…
521	2022	Septiembre	Banco de Machala	48083
522	2022	Septiembre	Banco del Austro	266384
523	2022	Septiembre	Banco del Pacífico	3550967
524	2022	Septiembre	Diners Club	4966401
525	2022	Septiembre	Produbanco	1318521

[526 rows x 4 columns]

3.2 Distribucion segmento tarjeta

[29]:
```
# Let's improve the previous code for a thesis-level presentation

# Group the data by year, segment, and sum totals
grouped_data = df_tc.groupby(['Anio', 'Segmento'], as_index=False)['Total'].
  ↪sum()
```

```
# Pivot the data to have years as columns and segments as the index
pivot_segment_by_year_x = grouped_data.pivot(index='Segmento', columns='Anio',␣
 ↪values='Total')

# Define a formal color palette for the years
formal_colors = ['#1f77b4', '#ff7f0e', '#2ca02c', '#d62728', '#9467bd']  #␣
 ↪Blue, orange, green, red, purple

# Create the stacked bar chart with formal colors
fig, ax = plt.subplots(figsize=(12, 8))  # Larger figure size for better␣
 ↪clarity in a thesis

# Plot the stacked bars for each segment with years in more formal colors
pivot_segment_by_year_x.plot(kind='bar', stacked=True, ax=ax,␣
 ↪color=formal_colors)

# Customize the title and labels for a thesis presentation
plt.title("Distribución de Tarjetas de Crédito por Año y Segmento",␣
 ↪fontsize=18, weight='bold')
plt.xlabel("Segmento de Tarjeta", fontsize=14, weight='bold')
plt.ylabel("Cantidad Total de Tarjetas", fontsize=14, weight='bold')

# Format the Y-axis in thousands (k)
ax.get_yaxis().set_major_formatter(plt.FuncFormatter(lambda x, loc: f'{int(x/
 ↪1000)}k'))

# Style the legend for a more formal presentation
ax.legend(title='Año', bbox_to_anchor=(1.05, 1), loc='upper left',␣
 ↪title_fontsize=12, fontsize=10)

# Display gridlines for better readability
ax.grid(True, linestyle='--', alpha=0.7)

# Ensure the layout is tight and save the figure
plt.tight_layout()
plt.savefig("graficos/distribucion_segmentosxanio_tesis.png",␣
 ↪bbox_inches='tight', format='png')

# Display the plot
plt.show()
```



4 Analisis de correlacion

[30]:
```
# Realizar One Hot Encoding
df_encoded = pd.get_dummies(df_tc, columns=['Emisor','Marca Tarjeta',␣
 ↪'Segmento','Mes','Anio'])
df_encoded = df_encoded.drop('Tipo Tarjeta', axis=1)
# Mostrar las primeras filas del DataFrame codificado
df_encoded.head()
```

[30]:
```
    Total  Emisor_Banco Bolivariano  Emisor_Banco Guayaquil  \
12  83681                     False                   False
13  72585                     False                   False
14  80254                     False                   False
15  73573                     False                   False
16  76015                     False                   False

    Emisor_Banco Internacional  Emisor_Banco Pichincha  \
12                       False                   False
13                       False                   False
14                       False                   False
15                       False                   False
16                       False                   False
```

```
    Emisor_Banco Rumiñahui  Emisor_Banco Solidario  Emisor_Banco de Loja  \
12                   False                   False                 False
13                   False                   False                 False
14                   False                   False                 False
15                   False                   False                 False
16                   False                   False                 False

    Emisor_Banco de Machala  Emisor_Banco del Austro  …  Mes_Junio  \
12                    False                   False  …      False
13                    False                   False  …      False
14                    False                   False  …      False
15                    False                   False  …      False
16                    False                   False  …      False

    Mes_Marzo  Mes_Mayo  Mes_Noviembre  Mes_Octubre  Mes_Septiembre  \
12      False     False          False        False           False
13      False     False          False        False           False
14       True     False          False        False           False
15      False     False          False        False           False
16      False      True          False        False           False

    Anio_2023  Anio_2024  Anio_2021  Anio_2022
12      False      False       True      False
13      False      False       True      False
14      False      False       True      False
15      False      False       True      False
16      False      False       True      False

[5 rows x 55 columns]
```

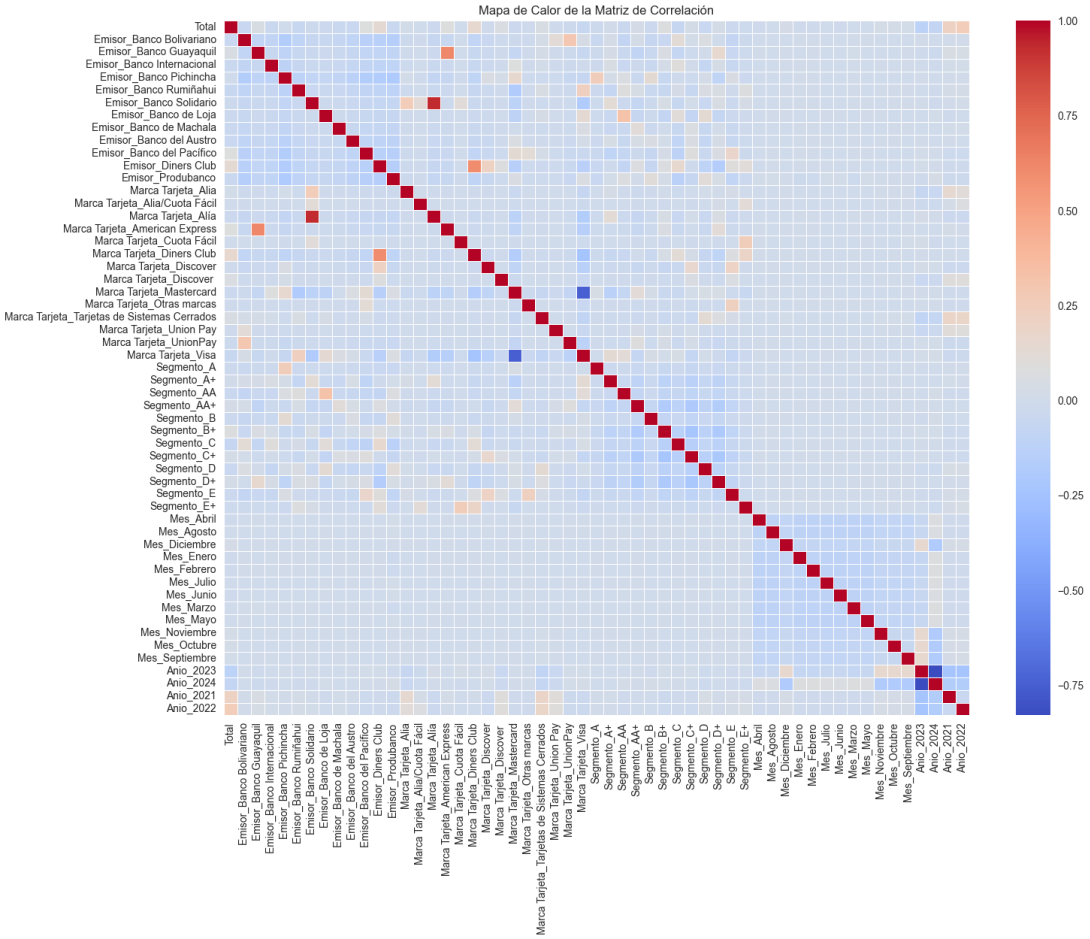
[31]:
```
correlation_matrix = df_encoded.corr()

# Mostrar la matriz de correlación
correlation_matrix.head()
```

[31]:
```
                              Total  Emisor_Banco Bolivariano  \
Total                      1.000000                 -0.049440
Emisor_Banco Bolivariano  -0.049440                  1.000000
Emisor_Banco Guayaquil     0.042943                 -0.082254
Emisor_Banco Internacional -0.041454                -0.091296
Emisor_Banco Pichincha    -0.001970                 -0.171178

                           Emisor_Banco Guayaquil  \
Total                                    0.042943
Emisor_Banco Bolivariano                -0.082254
Emisor_Banco Guayaquil                   1.000000
```

```
Emisor_Banco Internacional              -0.054513
Emisor_Banco Pichincha                  -0.102211

                          Emisor_Banco Internacional  \
Total                                      -0.041454
Emisor_Banco Bolivariano                   -0.091296
Emisor_Banco Guayaquil                     -0.054513
Emisor_Banco Internacional                  1.000000
Emisor_Banco Pichincha                     -0.113447

                          Emisor_Banco Pichincha  Emisor_Banco Rumiñahui  \
Total                                  -0.001970               -0.051770
Emisor_Banco Bolivariano               -0.171178               -0.093746
Emisor_Banco Guayaquil                 -0.102211               -0.055976
Emisor_Banco Internacional             -0.113447               -0.062129
Emisor_Banco Pichincha                  1.000000               -0.116491

                          Emisor_Banco Solidario  Emisor_Banco de Loja  \
Total                                  -0.024980             -0.034028
Emisor_Banco Bolivariano               -0.063787             -0.056967
Emisor_Banco Guayaquil                 -0.038087             -0.034015
Emisor_Banco Internacional             -0.042274             -0.037754
Emisor_Banco Pichincha                 -0.079263             -0.070789

                          Emisor_Banco de Machala  Emisor_Banco del Austro  \
Total                                  -0.041148                -0.047821
Emisor_Banco Bolivariano               -0.069269                -0.100099
Emisor_Banco Guayaquil                 -0.041361                -0.059770
Emisor_Banco Internacional             -0.045908                -0.066340
Emisor_Banco Pichincha                 -0.086076                -0.124386

                          …  Mes_Junio  Mes_Marzo  Mes_Mayo  \
Total                     …  -0.008246  -0.005171 -0.005572
Emisor_Banco Bolivariano  …  -0.000775   0.001388  0.000181
Emisor_Banco Guayaquil    …  -0.001869   0.007691 -0.002709
Emisor_Banco Internacional …  -0.001584   0.000532 -0.000881
Emisor_Banco Pichincha    …  -0.001858   0.002406 -0.001541

                          Mes_Noviembre  Mes_Octubre  Mes_Septiembre  \
Total                          0.021207     0.016773        0.013932
Emisor_Banco Bolivariano      -0.002197    -0.000947       -0.000890
Emisor_Banco Guayaquil        -0.000447    -0.001147        0.005467
Emisor_Banco Internacional     0.000476    -0.000705        0.000122
Emisor_Banco Pichincha        -0.000204    -0.000795       -0.000865

                          Anio_2023  Anio_2024  Anio_2021  Anio_2022
Total                     -0.118060  -0.077081   0.207362   0.253794
```

```
Emisor_Banco Bolivariano      0.000689  -0.001195   0.000838   0.000284
Emisor_Banco Guayaquil       -0.014066  -0.024055   0.049196   0.040191
Emisor_Banco Internacional   -0.001331  -0.010245   0.013371   0.013549
Emisor_Banco Pichincha        0.000434  -0.003118   0.005667   0.000513

[5 rows x 55 columns]
```

[32]:
```
# Generar un mapa de calor para la matriz de correlación
plt.figure(figsize=(16, 12))
sns.heatmap(correlation_matrix, annot=False, cmap='coolwarm', fmt=".2f",␣
 ↪linewidths=.5)
plt.title('Mapa de Calor de la Matriz de Correlación')
plt.savefig("graficos/correlacion.png", bbox_inches='tight', format='png')
plt.show()
```



[33]:
```
# Convertir los meses a valores numéricos
meses_dict = {
```

```
    "Enero": 1, "Febrero": 2, "Marzo": 3, "Abril": 4, "Mayo": 5, "Junio": 6,
    "Julio": 7, "Agosto": 8, "Septiembre": 9, "Octubre": 10, "Noviembre": 11,␣
 ↪"Diciembre": 12
}
df_tc['Mes_num'] = df_tc['Mes'].map(meses_dict)
```

[34]:
```
# Crear una columna de fecha combinando año y mes
df_tc['Fecha'] = pd.to_datetime(df_tc['Anio'].astype(str) + '-' +␣
 ↪df_tc['Mes_num'].astype(str) + '-01')
```

[35]:
```
df_tc.head(5)
```

[35]:

	Emisor	Marca Tarjeta	Segmento	Tipo Tarjeta	Mes	Total	\
12	Diners Club	Diners Club	A+	Tarjeta de crédito	Enero	83681	
13	Diners Club	Diners Club	A+	Tarjeta de crédito	Febrero	72585	
14	Diners Club	Diners Club	A+	Tarjeta de crédito	Marzo	80254	
15	Diners Club	Diners Club	A+	Tarjeta de crédito	Abril	73573	
16	Diners Club	Diners Club	A+	Tarjeta de crédito	Mayo	76015	

	Anio	Mes_num	Fecha
12	2021	1	2021-01-01
13	2021	2	2021-02-01
14	2021	3	2021-03-01
15	2021	4	2021-04-01
16	2021	5	2021-05-01

[36]:
```
dfFinal=df_tc[["Emisor","Marca Tarjeta","Segmento","Fecha","Total"]]
```

[37]:
```
dfFinal.to_csv('datos/final_todos_campos.csv', index=False)
```

[38]:
```
#df_tc.groupby(['Fecha']).agg({'Total': 'sum'}).reset_index().to_csv('datos/
 ↪agrupadoTotal.csv', index=False)
```

[39]:
```
grouped_df = df_tc.groupby(['Fecha', 'Emisor']).agg({'Total': 'sum'}).
 ↪reset_index()
```

[40]:
```
grouped_df['Emisor'].unique()
```

[40]:
```
array(['Banco Bolivariano', 'Banco Guayaquil', 'Banco Internacional',
       'Banco Pichincha', 'Banco Rumiñahui', 'Banco Solidario',
       'Banco de Loja', 'Banco de Machala', 'Banco del Austro',
       'Banco del Pacífico', 'Diners Club', 'Produbanco'], dtype=object)
```

[41]:
```
grouped_df
```

[41]:

	Fecha	Emisor	Total
0	2021-01-01	Banco Bolivariano	538970
1	2021-01-01	Banco Guayaquil	20669

```
2   2021-01-01  Banco Internacional   169946
3   2021-01-01      Banco Pichincha  1787445
4   2021-01-01      Banco Rumiñahui    96846
..         …                   …      …
521 2024-08-01     Banco de Machala    54107
522 2024-08-01     Banco del Austro   427810
523 2024-08-01   Banco del Pacífico  3647063
524 2024-08-01          Diners Club  6045488
525 2024-08-01           Produbanco  1595036

[526 rows x 3 columns]
```

[42]:
```
#Validar que cada entidad este en todos los años
df_distinct = df_tc[['Anio', 'Emisor']].drop_duplicates()
df_distinct.groupby(['Emisor']).count()
```

[42]:

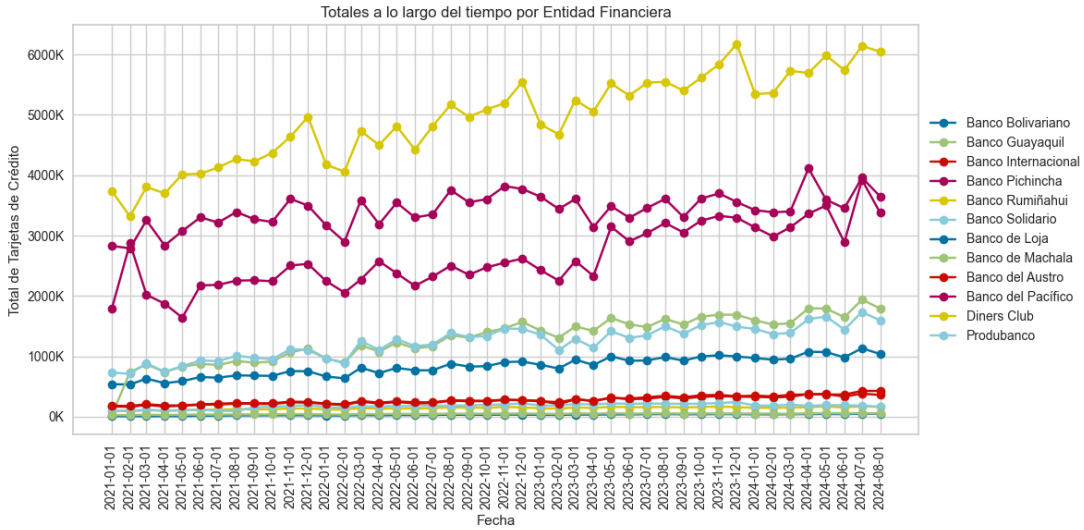
Emisor	Anio
Banco Bolivariano	4
Banco Guayaquil	4
Banco Internacional	4
Banco Pichincha	4
Banco Rumiñahui	4
Banco Solidario	4
Banco de Loja	4
Banco de Machala	4
Banco del Austro	4
Banco del Pacífico	4
Diners Club	4
Produbanco	4

4.1 Distribucion de entidades a los largo del tiempo

[43]:
```
import matplotlib.pyplot as plt
import pandas as pd
from matplotlib.ticker import FuncFormatter

# Supongo que grouped_df ya está cargado y contiene las columnas 'Emisor',␣
 ↪'Fecha', y 'Total'
# Convertir 'Fecha' a formato de fecha si no está ya en ese formato
grouped_df['Fecha'] = pd.to_datetime(grouped_df['Fecha'])

# Función para agregar "K" en los valores del eje Y
def thousands_formatter(x, pos):
    return f'{int(x)}K'

# Configurar la figura del gráfico
```

```
plt.figure(figsize=(12, 6))

# Graficar los totales a lo largo del tiempo para cada entidad financiera
for entidad in grouped_df['Emisor'].unique():
    entidad_data = grouped_df[grouped_df['Emisor'] == entidad]
    plt.plot(entidad_data['Fecha'], entidad_data['Total'] / 1000, marker='o',␣
 ↪label=entidad)  # Dividir por 1000 para mostrar en miles

# Configurar el formato del eje X para mostrar todos los meses y años
plt.xticks(pd.date_range(start=grouped_df['Fecha'].min(),␣
 ↪end=grouped_df['Fecha'].max(), freq='MS'), rotation=90)

# Añadir título y etiquetas
plt.title('Totales a lo largo del tiempo por Entidad Financiera')
plt.xlabel('Fecha')
plt.ylabel('Total de Tarjetas de Crédito')  # Actualizar la etiqueta del eje Y␣
 ↪para reflejar la escala en miles
plt.grid(True)

# Aplicar el formateador al eje Y
plt.gca().yaxis.set_major_formatter(FuncFormatter(thousands_formatter))

# Colocar la leyenda fuera del gráfico
plt.legend(loc='center left', bbox_to_anchor=(1, 0.5))
plt.tight_layout()  # Ajusta el diseño para que todo encaje bien

# Guardar la figura
plt.savefig("graficos/total_entidad_complete.png", bbox_inches='tight',␣
 ↪format='png')
plt
```

[43]: <module 'matplotlib.pyplot' from 'D:\\Anaconda\\envs\\MDC\\Lib\\site-packages\\matplotlib\\pyplot.py'>



5 IMPLEMENTACIÓN DEL MODELO DE ML

[44]:
```
# Normalizar la columna 'Total'
scaler = MinMaxScaler()
df_tc['Total_scaled'] = scaler.fit_transform(df_tc[['Total']])
```

5.0.1 Seleccionar la entidad que desear realizar el modelo

[45]:
```
#Menu para seleccionar entidad
from ipywidgets import interact, widgets


# Variable global para almacenar el DataFrame filtrado o la sumatoria
df_serie_filtrado = None

# Función para filtrar por emisor o calcular la sumatoria total
def obtener_df_serie(emisor):
    global df_serie_filtrado  # Declaramos la variable como global

    if emisor == 'Todo':
        # Agrupar por fecha y calcular la sumatoria de 'Total'
        df_serie_filtrado = grouped_df.groupby('Fecha')['Total'].sum().
↪reset_index()
    else:
        # Filtrar por emisor seleccionado
        df_serie_filtrado = grouped_df[['Fecha', 'Total']][grouped_df['Emisor']␣
↪== emisor]
```

```
    # Establecer la columna 'Fecha' como índice
    df_serie_filtrado.set_index('Fecha', inplace=True)

    # Mostrar el DataFrame resultante
    display(df_serie_filtrado)

# Crear un menú desplegable con los emisores únicos y la opción "Todo"
emisores = list(grouped_df['Emisor'].unique()) + ['Todo']
dropdown = widgets.Dropdown(
    options=emisores,
    description='Emisor:',
    value='Todo'  # Valor por defecto
)

# Vincular el menú con la función
interact(obtener_df_serie, emisor=dropdown)
```

```
interactive(children=(Dropdown(description='Emisor:', index=12, options=('Banco␣
↪Bolivariano', 'Banco Guayaquil…
```

[45]:
```
<function __main__.obtener_df_serie(emisor)>
```

[46]:
```
dropdown.value
#Para realizar de una entidad especifica
#df_serie=grouped_df[['Fecha', 'Total']][grouped_df['Emisor'] == 'Banco del␣
↪Pacífico']
# Establece la columna 'Fecha' como índice
#df_serie.set_index('Fecha', inplace=True)
```

[46]:
```
'Todo'
```

[47]:
```
df_serie_filtrado.to_csv('datos/Datos_'+dropdown.value+'.csv', index=True)
```

[48]:
```
df_serie=df_serie_filtrado
```

[49]:
```
df_serie
```

[49]:

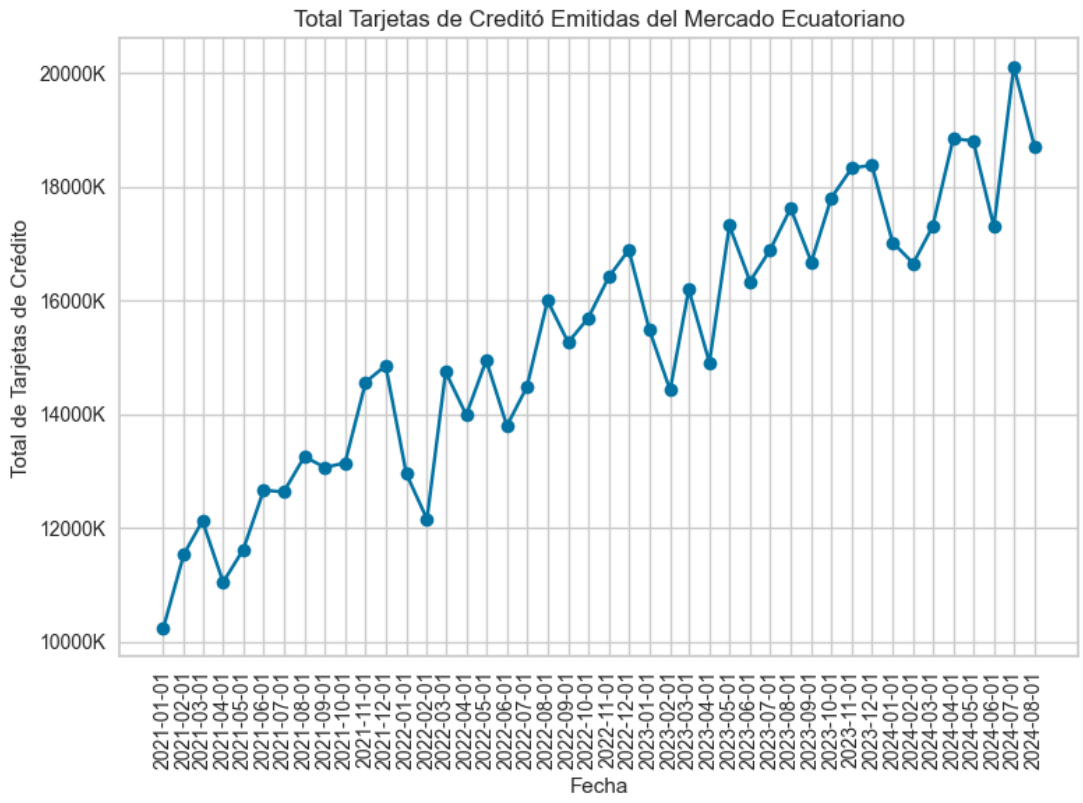
Fecha	Total
2021-01-01	10228509
2021-02-01	11535568
2021-03-01	12123085
2021-04-01	11045675
2021-05-01	11614115
2021-06-01	12664405
2021-07-01	12632293
2021-08-01	13252874
2021-09-01	13063143

```
2021-10-01  13144877
2021-11-01  14565273
2021-12-01  14854425
2022-01-01  12953198
2022-02-01  12155624
2022-03-01  14743210
2022-04-01  13987191
2022-05-01  14948784
2022-06-01  13796394
2022-07-01  14478792
2022-08-01  15999459
2022-09-01  15262781
2022-10-01  15689955
2022-11-01  16413438
2022-12-01  16889544
2023-01-01  15480735
2023-02-01  14422994
2023-03-01  16193813
2023-04-01  14909238
2023-05-01  17317785
2023-06-01  16335205
2023-07-01  16886289
2023-08-01  17621861
2023-09-01  16675890
2023-10-01  17804690
2023-11-01  18331790
2023-12-01  18376047
2024-01-01  17022283
2024-02-01  16653495
2024-03-01  17297934
2024-04-01  18844222
2024-05-01  18809455
2024-06-01  17299736
2024-07-01  20113790
2024-08-01  18710862
```

5.0.2 Grafico de serie de tiempo seleccionada

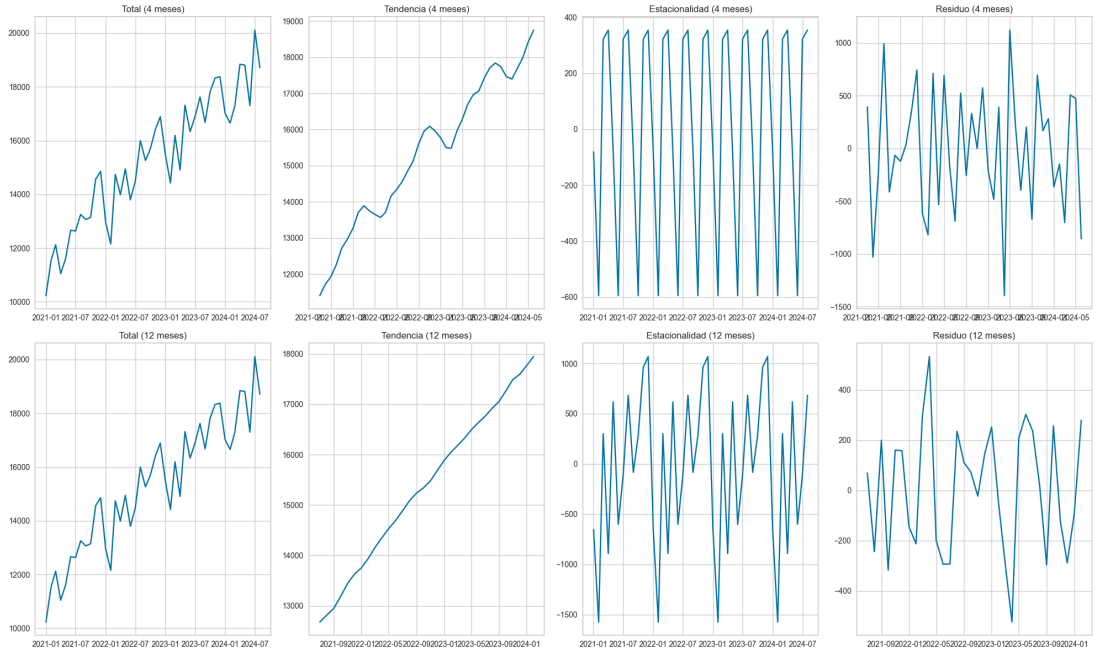
[50]:
```python
import matplotlib.pyplot as plt
import pandas as pd
from matplotlib.ticker import FuncFormatter


fechas = df_serie.index
totales = df_serie['Total']

# Crear la gráfica
```

```
plt.figure(figsize=(8,6))
plt.plot(fechas, totales, marker='o', linestyle='-', color='b')

# Formatear el eje Y para mostrar en miles
formatter = FuncFormatter(lambda x, pos: f'{int(x/1000)}K')
plt.gca().yaxis.set_major_formatter(formatter)

# Personalizar el gráfico
plt.title('Total Tarjetas de Creditó Emitidas del Mercado Ecuatoriano')
plt.xlabel('Fecha')
plt.ylabel('Total de Tarjetas de Crédito')
plt.grid(True)
plt.xticks(fechas, fechas.strftime('%Y-%m-%d'), rotation=90)

# Mostrar el gráfico
plt.tight_layout()
plt.savefig("graficos/serie_tiempo_modelado.png", bbox_inches='tight',␣
 ↪format='png')
plt.show()
```



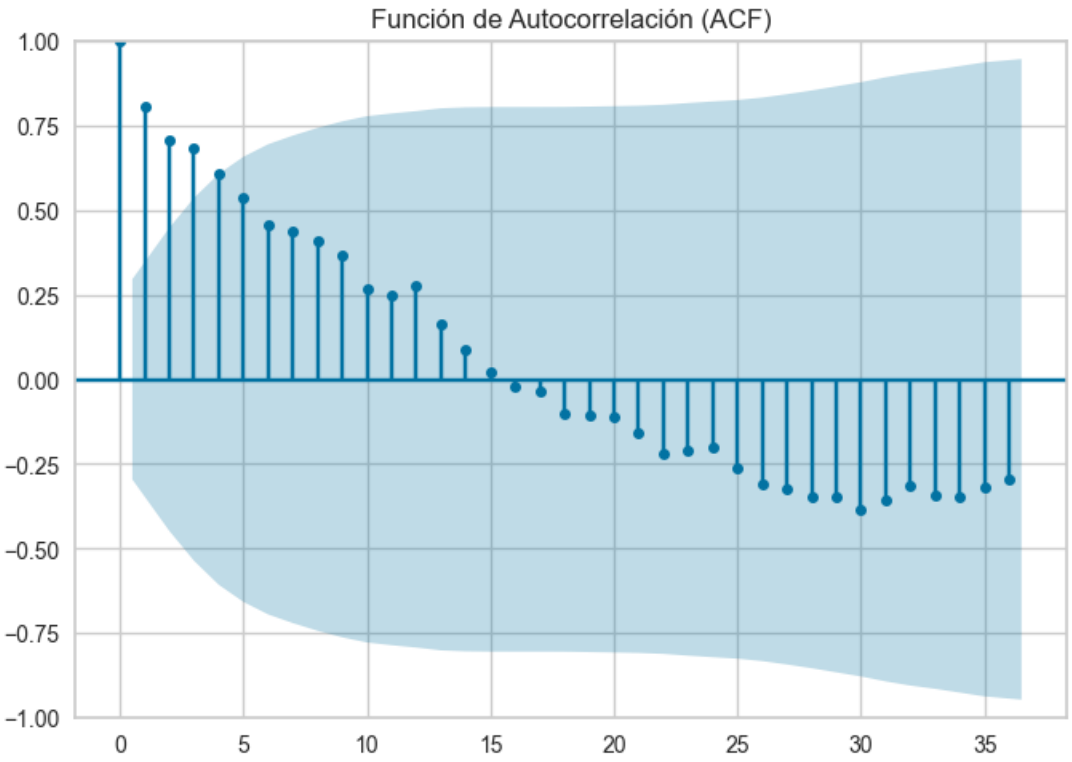
5.0.3 Analisis de estacionalidad, tendencia y periodo

[51]:
```
import pandas as pd
import matplotlib.pyplot as plt
from statsmodels.tsa.seasonal import seasonal_decompose

# Crear un DataFrame de paso y convertir los valores a miles
df_paso = df_serie.copy()
df_paso['Total'] = df_paso['Total'] / 1000

# Descomposición estacional con periodos de 4 y 12 meses en el DataFrame de paso
decomposition_4 = seasonal_decompose(df_paso['Total'], model='additive',␣
 ↪period=4)
decomposition_12 = seasonal_decompose(df_paso['Total'], model='additive',␣
 ↪period=12)

# Graficar ambas descomposiciones para comparar
fig, axs = plt.subplots(2, 4, figsize=(20, 12))

# Descomposición con periodo de 4 meses
axs[0, 0].plot(decomposition_4.observed)
axs[0, 0].set_title("Total (4 meses)")
axs[0, 1].plot(decomposition_4.trend)
axs[0, 1].set_title("Tendencia (4 meses)")
axs[0, 2].plot(decomposition_4.seasonal)
axs[0, 2].set_title("Estacionalidad (4 meses)")
axs[0, 3].plot(decomposition_4.resid)
axs[0, 3].set_title("Residuo (4 meses)")

# Descomposición con periodo de 12 meses
axs[1, 0].plot(decomposition_12.observed)
axs[1, 0].set_title("Total (12 meses)")
axs[1, 1].plot(decomposition_12.trend)
axs[1, 1].set_title("Tendencia (12 meses)")
axs[1, 2].plot(decomposition_12.seasonal)
axs[1, 2].set_title("Estacionalidad (12 meses)")
axs[1, 3].plot(decomposition_12.resid)
axs[1, 3].set_title("Residuo (12 meses)")

# Ajustar diseño
plt.tight_layout()
plt.savefig("graficos/estacionalidad.png", bbox_inches='tight', format='png')
plt.show()
```

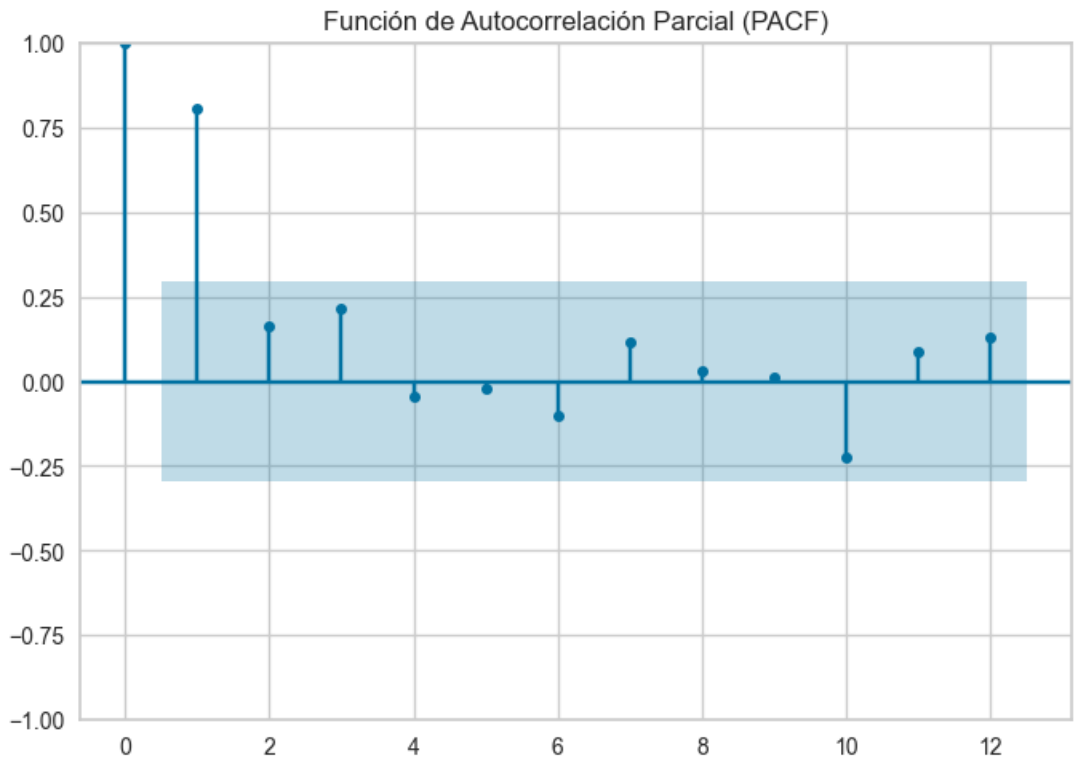


[52]:
```
from statsmodels.graphics.tsaplots import plot_acf, plot_pacf

plt.figure(figsize=(12, 6))
plot_acf(df_serie['Total'].dropna(), lags=36)
plt.title("Función de Autocorrelación (ACF)")
plt.show()

plt.figure(figsize=(12, 6))
plot_pacf(df_serie['Total'].dropna(), lags=12)
plt.title("Función de Autocorrelación Parcial (PACF)")
plt.show()
```

```
<Figure size 1200x600 with 0 Axes>
```




```
<Figure size 1200x600 with 0 Axes>
```



5.0.4 Métodos estadísticos para verificar si la estacionalidad es significativa

[53]:
```
from statsmodels.tsa.stattools import adfuller

# Realizar la prueba de Dickey-Fuller Aumentada
result = adfuller(df_serie['Total'].dropna())
print('ADF Statistic:', result[0])
print('p-value:', result[1])

# Evaluación del resultado
if result[1] < 0.05:
    print("La serie es estacionaria (rechazamos la hipótesis nula de no␣
 ↪estacionariedad)")
else:
    print("La serie no es estacionaria (no podemos rechazar la hipótesis nula)")
```

```
ADF Statistic: -1.0693984527352351
p-value: 0.7271199319905544
La serie no es estacionaria (no podemos rechazar la hipótesis nula)
```

[54]:
```
from statsmodels.tsa.stattools import kpss
```

```
result_kpss = kpss(df_serie['Total'].dropna(), regression='c')
print('KPSS Statistic:', result_kpss[0])
print('p-value:', result_kpss[1])

# Evaluación del resultado
if result_kpss[1] < 0.05:
    print("La serie no es estacionaria (rechazamos la hipótesis nula de␣
 ↪estacionariedad)")
else:
    print("La serie es estacionaria (no podemos rechazar la hipótesis nula)")
```

```
KPSS Statistic: 0.9749597088074229
p-value: 0.01
La serie no es estacionaria (rechazamos la hipótesis nula de estacionariedad)
```

5.0.5 Configuracion de entorno pycaret

[55]:
```
s = setup(df_serie,target='Total', fh = 3,fold=4, session_id =␣
 ↪123,log_experiment='mlflow',seasonal_period=4)
```

```
<pandas.io.formats.style.Styler at 0x207a91e1910>
```

5.0.6 Comparar modelos

[56]:
```
# compare baseline models
best = compare_models(exclude = ['naive'],sort='R2')
```

```
<IPython.core.display.HTML object>
```

```
<pandas.io.formats.style.Styler at 0x207a13c8050>
```

```
<IPython.core.display.HTML object>
```

```
2024/12/05 11:54:11 WARNING mlflow.models.model: Input example should be
provided to infer model signature if the model signature is not provided when
logging the model.
2024/12/05 11:54:13 WARNING mlflow.models.model: Input example should be
provided to infer model signature if the model signature is not provided when
logging the model.
2024/12/05 11:54:14 WARNING mlflow.models.model: Input example should be
provided to infer model signature if the model signature is not provided when
logging the model.
2024/12/05 11:54:15 WARNING mlflow.models.model: Input example should be
provided to infer model signature if the model signature is not provided when
logging the model.
2024/12/05 11:54:17 WARNING mlflow.models.model: Input example should be
provided to infer model signature if the model signature is not provided when
logging the model.
2024/12/05 11:54:19 WARNING mlflow.models.model: Input example should be
provided to infer model signature if the model signature is not provided when
```

```
logging the model.
2024/12/05 11:54:20 WARNING mlflow.models.model: Input example should be
provided to infer model signature if the model signature is not provided when
logging the model.
2024/12/05 11:54:22 WARNING mlflow.models.model: Input example should be
provided to infer model signature if the model signature is not provided when
logging the model.
2024/12/05 11:54:23 WARNING mlflow.models.model: Input example should be
provided to infer model signature if the model signature is not provided when
logging the model.
2024/12/05 11:54:24 WARNING mlflow.models.model: Input example should be
provided to infer model signature if the model signature is not provided when
logging the model.
2024/12/05 11:54:26 WARNING mlflow.models.model: Input example should be
provided to infer model signature if the model signature is not provided when
logging the model.
2024/12/05 11:54:27 WARNING mlflow.models.model: Input example should be
provided to infer model signature if the model signature is not provided when
logging the model.
2024/12/05 11:54:29 WARNING mlflow.models.model: Input example should be
provided to infer model signature if the model signature is not provided when
logging the model.
2024/12/05 11:54:30 WARNING mlflow.models.model: Input example should be
provided to infer model signature if the model signature is not provided when
logging the model.
2024/12/05 11:54:31 WARNING mlflow.models.model: Input example should be
provided to infer model signature if the model signature is not provided when
logging the model.
2024/12/05 11:54:33 WARNING mlflow.models.model: Input example should be
provided to infer model signature if the model signature is not provided when
logging the model.
2024/12/05 11:54:34 WARNING mlflow.models.model: Input example should be
provided to infer model signature if the model signature is not provided when
logging the model.
2024/12/05 11:54:36 WARNING mlflow.models.model: Input example should be
provided to infer model signature if the model signature is not provided when
logging the model.
2024/12/05 11:54:37 WARNING mlflow.models.model: Input example should be
provided to infer model signature if the model signature is not provided when
logging the model.
2024/12/05 11:54:39 WARNING mlflow.models.model: Input example should be
provided to infer model signature if the model signature is not provided when
logging the model.
2024/12/05 11:54:40 WARNING mlflow.models.model: Input example should be
provided to infer model signature if the model signature is not provided when
logging the model.
```

5.0.7 Tuning Model

[57]:
```
tuned_model = tune_model(best,optimize='R2',fold=6)
```

```
<IPython.core.display.HTML object>
```

```
<pandas.io.formats.style.Styler at 0x207a498bd50>
```

```
<IPython.core.display.HTML object>
```

```
Fitting 6 folds for each of 10 candidates, totalling 60 fits
```

```
[Parallel(n_jobs=-1)]: Using backend LokyBackend with 8 concurrent workers.
[Parallel(n_jobs=-1)]: Done  45 out of  60 | elapsed:    1.0s remaining:    0.3s
[Parallel(n_jobs=-1)]: Done  60 out of  60 | elapsed:    1.2s finished
```

[58]:
```
tuned_model
```

[58]:
```
AutoETS(error='mul', seasonal='mul', sp=4, trend='add')
```

5.0.8 Evaluate Model

[59]:
```
predicciones_evaluaciones = predict_model(tuned_model)
predicciones_evaluaciones['y_pred']
```

```
<pandas.io.formats.style.Styler at 0x207aa5f2790>
```

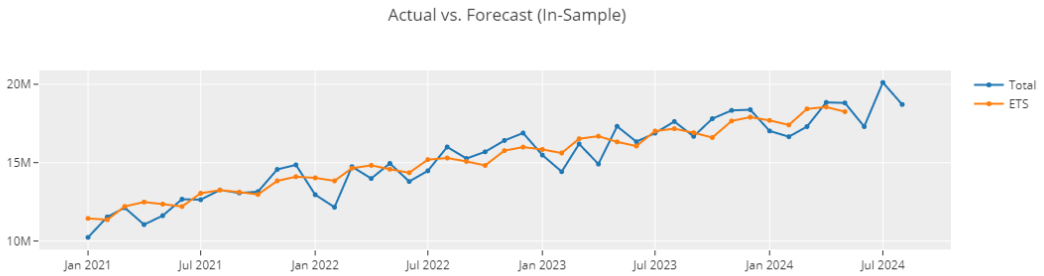
[59]:
```
2024-06    1.795179e+07
2024-07    1.902086e+07
2024-08    1.920440e+07
Freq: M, Name: y_pred, dtype: float64
```

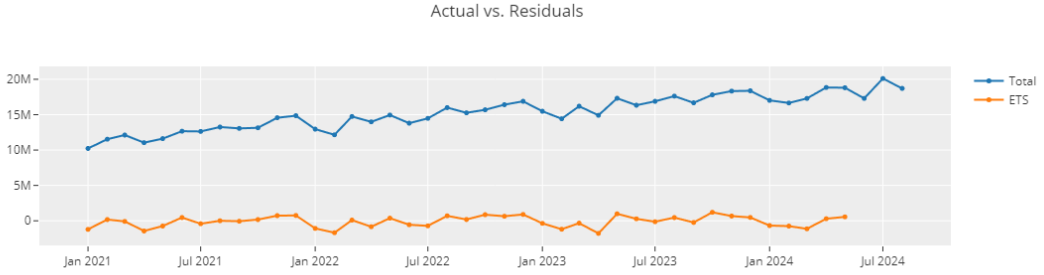
[60]:
```
test_data = get_config('y_test')
df_combinado = pd.concat([test_data, predicciones_evaluaciones['y_pred']],␣
 ↪axis=1)

# Renombra las columnas para mayor claridad
df_combinado.columns = ['Total Real', 'Predicción']

# Formatear ambas columnas para que tengan una presentación uniforme
df_combinado['Total Real'] = df_combinado['Total Real'].apply(lambda x: f"{x:,.
 ↪0f}")
df_combinado['Predicción'] = df_combinado['Predicción'].apply(lambda x: f"{x:,.
 ↪0f}")

# Convertir las columnas 'Total Real' y 'Predicción' a tipo numérico
# Elimina las comas y convierte a formato entero
df_combinado['Total Real'] = df_combinado['Total Real'].replace(',', '',␣
 ↪regex=True).astype(int)
df_combinado['Predicción'] = df_combinado['Predicción'].replace(',', '',␣
 ↪regex=True).astype(int)
```

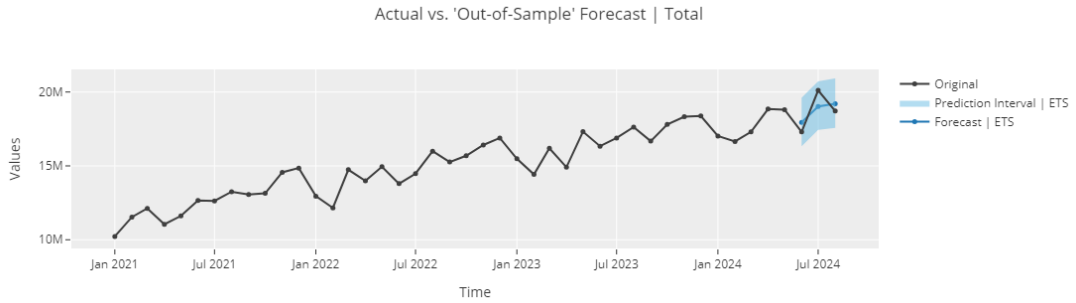
```
# Mostrar el resultado combinado con el formato deseado
print(df_combinado)
```

```
         Total Real  Predicción
2024-06    17299736    17951794
2024-07    20113790    19020859
2024-08    18710862    19204398
```

[61]:
```
predicciones_hist=plot_model(tuned_model, plot = 'insample',return_data=True)
residual=plot_model(tuned_model, plot = 'residuals',return_data=True)
valores_prediccion=plot_model(tuned_model,return_data=True)
```

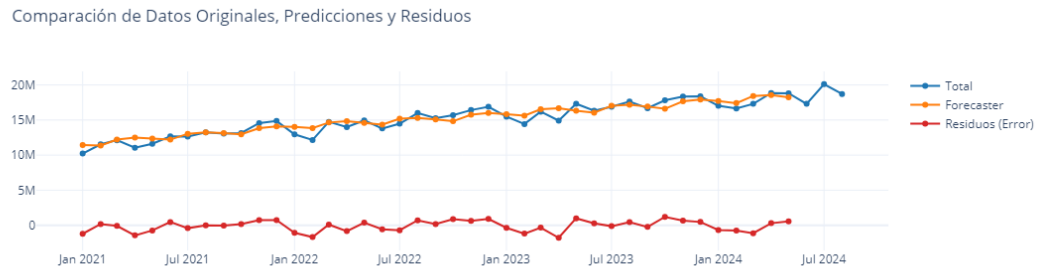






[62]:
```
import plotly.graph_objects as go


original_data = predicciones_hist['original_data']
predictions =   predicciones_hist['overlay_data'].iloc[:, 0]
residuals = residual['overlay_data'].iloc[:, 0]


if isinstance(original_data.index, pd.PeriodIndex):
    original_data.index = original_data.index.to_timestamp()

if isinstance(predictions.index, pd.PeriodIndex):
    predictions.index = predictions.index.to_timestamp()

if isinstance(residuals.index, pd.PeriodIndex):
    residuals.index = residuals.index.to_timestamp()

# Crear la figura
fig = go.Figure()

# Agregar los datos originales
fig.add_trace(go.Scatter(x=original_data.index, y=original_data.values,
                         mode='lines+markers', name='Total',␣
 ↪line=dict(color='#1f77b4')))

# Agregar las predicciones
fig.add_trace(go.Scatter(x=predictions.index, y=predictions.values,
                         mode='lines+markers', name='Forecaster',␣
 ↪line=dict(color='#ff7f0e')))

# Agregar los residuos
fig.add_trace(go.Scatter(x=residuals.index, y=residuals.values,
```

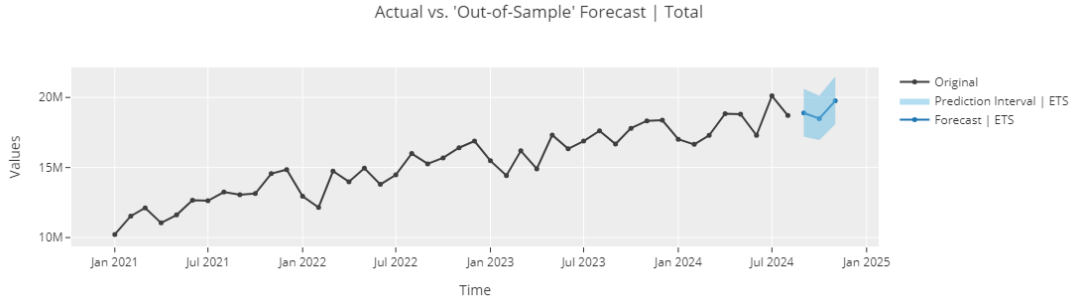
```
                         mode='lines+markers', name='Residuos (Error)',␣
 ↪line=dict(color='#d62728')))

# Configurar el diseño del gráfico
fig.update_layout(title='Comparación de Datos Originales, Predicciones y␣
 ↪Residuos',
                  #   xaxis_title='Fecha',
                   #  yaxis_title='Valor',
                    # legend_title='Series',
                     template='plotly_white')

# Mostrar el gráfico
fig.show()
```



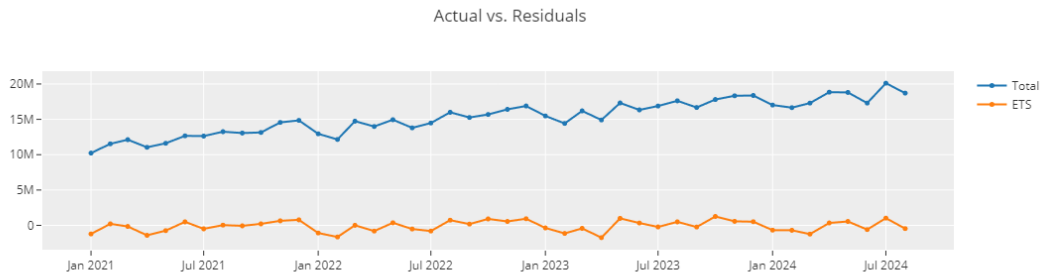
5.0.9 Finalize model

[63]:
```
final_model = finalize_model(tuned_model)
```

[64]:
```
# plot forecast
valoresNoConocidos_pred=plot_model(final_model, plot =␣
 ↪'forecast',return_data=True)
```



[65]:
```
# residuals plot
plot_model(final_model, plot = 'residuals')
```



[66]:
```
valores_futuros=predict_model(final_model,fh=4)
valores_futuros['y_pred'].astype(int)
```

[66]:
```
2024-09    18892582
2024-10    18495000
2024-11    19761607
2024-12    19789748
Freq: M, Name: y_pred, dtype: int32
```

5.0.10 Guardar Modelo

[67]:
```
# save pipeline
save_model(final_model, 'modelTesis_'+dropdown.value)

# load pipeline
loaded_best_pipeline = load_model( 'modelTesis_'+dropdown.value)
loaded_best_pipeline
```

```
Transformation Pipeline and Model Successfully Saved
Transformation Pipeline and Model Successfully Loaded
```

[67]:
```
ForecastingPipeline(steps=[('forecaster',
                            TransformedTargetForecaster(steps=[('model',
ForecastingPipeline(steps=[('forecaster',
             TransformedTargetForecaster(steps=[('model',
                                                 AutoETS(error='mul',
                                                         seasonal='mul',
                                                         sp=4,
                                                         trend='add'))]))]))]))])
```

5.1 Calcular variación de serie de tiempo

[68]:
```
# Calcular el porcentaje de variación de datos original
df_variacion = pd.DataFrame(original_data)
df_variacion["pct_variacion"] = df_variacion["Total"].pct_change() * 100
df_variacion_tail=df_variacion.tail(6)
df_variacion_tail
```

[68]:

	Total	pct_variacion
2024-03-01	17297934	3.869692
2024-04-01	18844222	8.939148
2024-05-01	18809455	-0.184497
2024-06-01	17299736	-8.026384
2024-07-01	20113790	16.266456
2024-08-01	18710862	-6.974956

[69]:
```
ultimo_valor_real = pd.DataFrame([df_variacion.iloc[-1]])
ultimo_valor_real=ultimo_valor_real.rename(columns={"Total": "y_pred"})
ultimo_valor_real
```

[69]:

	y_pred	pct_variacion
2024-08-01	18710862.0	-6.974956

[70]:
```
df_variacion_prediccion = pd.
↪DataFrame(valoresNoConocidos_pred['predictions'][0]['y_pred'])
df_unido_var = pd.concat([ultimo_valor_real, df_variacion_prediccion])
df_unido_var["pct_variacion"] = df_unido_var["y_pred"].pct_change() * 100
df_unido_var.index=pd.to_datetime(df_unido_var.index, format="%Y-%m")
df_linea_entre_real_y_pred=df_unido_var.head(2)
df_unido_var=df_unido_var.dropna()
df_unido_var
```

[70]:

	y_pred	pct_variacion
2024-09-01	1.889258e+07	0.971201
2024-10-01	1.849500e+07	-2.104432
2024-11-01	1.976161e+07	6.848375

[71]:
```
df_linea_entre_real_y_pred = df_linea_entre_real_y_pred.drop('pct_variacion',␣
↪axis=1)
df_linea_entre_real_y_pred
```

[71]:

	y_pred
2024-08-01	1.871086e+07
2024-09-01	1.889258e+07

[72]:
```
df_value_prc_variacion=pd.concat([df_variacion_tail, df_unido_var.
↪rename(columns={"y_pred": "Total"})])
df_value_prc_variacion["value"] = df_value_prc_variacion["Total"].diff()
```

```
df_value_prc_variacion
```

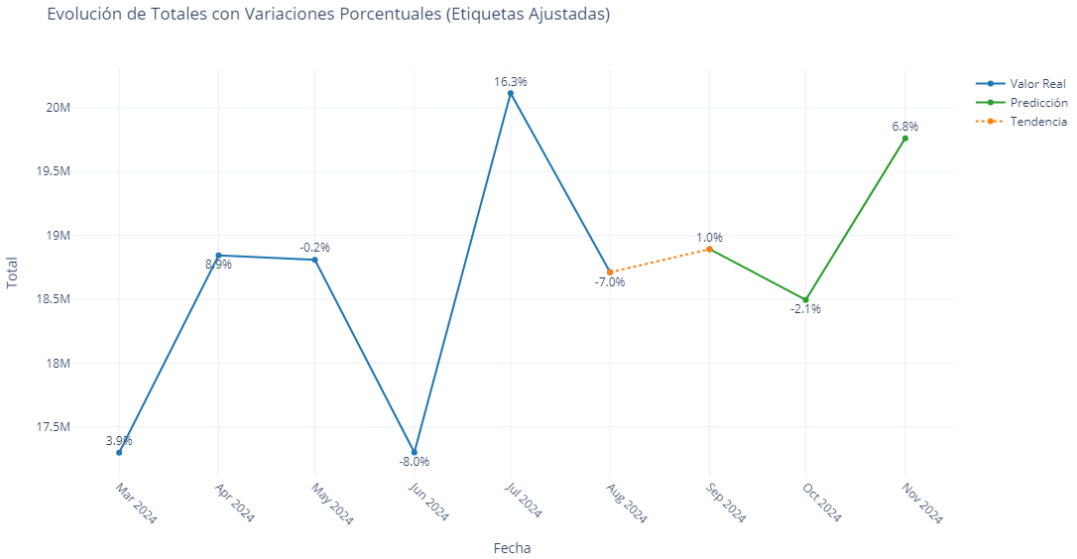
[72]:

	Total	pct_variacion	value
2024-03-01	1.729793e+07	3.869692	NaN
2024-04-01	1.884422e+07	8.939148	1.546288e+06
2024-05-01	1.880946e+07	-0.184497	-3.476700e+04
2024-06-01	1.729974e+07	-8.026384	-1.509719e+06
2024-07-01	2.011379e+07	16.266456	2.814054e+06
2024-08-01	1.871086e+07	-6.974956	-1.402928e+06
2024-09-01	1.889258e+07	0.971201	1.817202e+05
2024-10-01	1.849500e+07	-2.104432	-3.975815e+05
2024-11-01	1.976161e+07	6.848375	1.266607e+06

[73]:
```
# Preparar datos
fechas = df_variacion_tail.index
totales = df_variacion_tail["Total"]
variaciones = df_variacion_tail["pct_variacion"]

if isinstance(df_unido_var.index, pd.PeriodIndex):
    df_unido_var.index = df_unido_var.index.to_timestamp()

fechas_predic = df_unido_var.index
totales_predic = df_unido_var["y_pred"]
variaciones_predic = df_unido_var["pct_variacion"]


fechas_linea = df_linea_entre_real_y_pred.index
totales_linea = df_linea_entre_real_y_pred["y_pred"]
```

[74]:
```
import plotly.graph_objects as go
# Crear una figura mejorada para evitar superposición de etiquetas
fig = go.Figure()

# Alternar posición de etiquetas arriba y abajo para evitar superposición
positions = ['top center' if i % 2 == 0 else 'bottom center' for i in␣
 ↪range(len(fechas))]
positions_pred = ['top center' if i % 2 == 0 else 'bottom center' for i in␣
 ↪range(len(fechas_predic))]

# Añadir la línea de Total con etiquetas ajustadas
fig.add_trace(go.Scatter(
    x=fechas,
    y=totales,
    mode='lines+markers+text',
    text=[f"{pct:.1f}%" if pct is not None else "" for pct in variaciones],
    textposition=positions,
    name="Valor Real",
```

```
    line=dict(color='#1f77b4')
))

# Agregar las predicciones
fig.add_trace(go.Scatter(
    x=fechas_predic,
    y=totales_predic,
    text=[f"{pct:.1f}%" if pct is not None else "" for pct in␣
 ↪variaciones_predic],
    mode='lines+markers+text',
    name='Predicción',
    textposition=positions_pred,
    line=dict(color='#2ca02c')))

# Agregar las predicciones
fig.add_trace(go.Scatter(x=fechas_linea, y=totales_linea,
                         mode='lines+markers', name='Tendencia',␣
 ↪line=dict(dash='dot',color='#ff7f0e')))

# Ajustar diseño
fig.update_layout(
    title="Evolución de Totales con Variaciones Porcentuales (Etiquetas␣
 ↪Ajustadas)",
    xaxis_title="Fecha",
    yaxis_title="Total",
    xaxis=dict(tickangle=45),
    template="plotly_white",
    height=600,
    width=1000
)

# Mostrar el gráfico interactivo
fig.show()
```



5.2 Comparar con Modelo Base - Naive

[75]:
```
# Configurar el entorno para series temporales
exp_name = setup(df_serie,target='Total', fh = 3,fold=4, session_id =␣
↪123,log_experiment='mlflow',seasonal_period=4)

# Crear un modelo naive
naive_model = create_model('naive')
```

```
<pandas.io.formats.style.Styler at 0x207aaf52790>
```

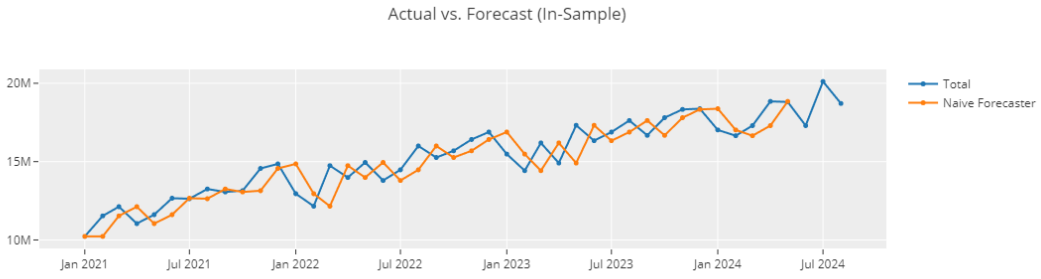
```
<IPython.core.display.HTML object>
```

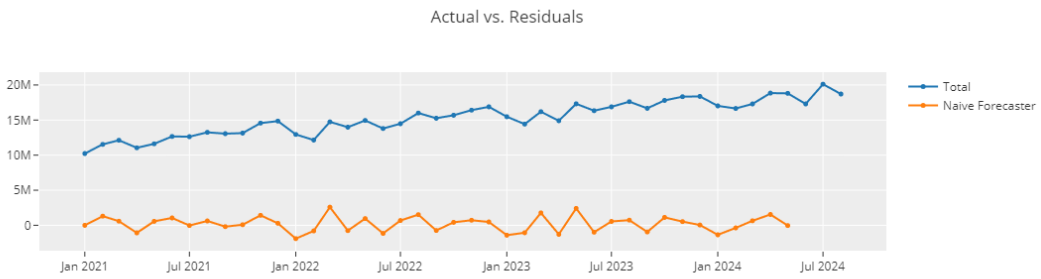
```
<pandas.io.formats.style.Styler at 0x207aabd0210>
```

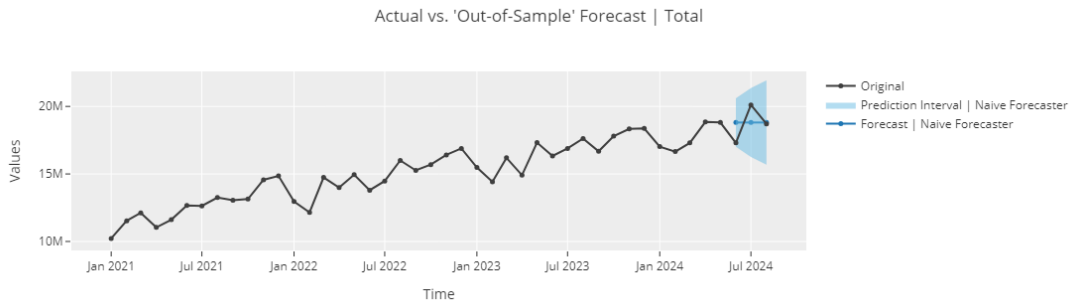
```
<IPython.core.display.HTML object>
```

```
2024/12/05 11:55:20 WARNING mlflow.models.model: Input example should be
provided to infer model signature if the model signature is not provided when
logging the model.
```

[76]:
```
# Evaluar el modelo naive
predicciones_naive=plot_model(naive_model, plot = 'insample',return_data=True)
residual_naive=plot_model(naive_model, plot = 'residuals',return_data=True)
valores_prediccion_naive=plot_model(naive_model, plot =␣
↪'forecast',return_data=True)
```

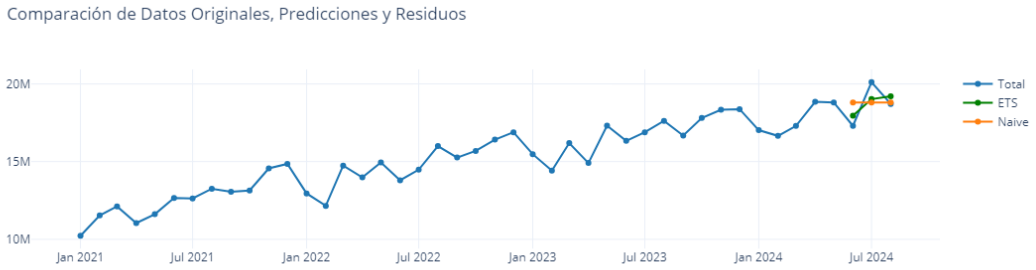






[77]:
```
import plotly.graph_objects as go


original_data_naive = predicciones_hist['original_data']
predictions_ETS =    valores_prediccion['predictions'][0]['y_pred']
predictions_naive = valores_prediccion_naive['predictions'][0]['y_pred']
```

```
if isinstance(original_data_naive.index, pd.PeriodIndex):
    original_data_naive.index = original_data_naive.index.to_timestamp()

if isinstance(predictions_ETS.index, pd.PeriodIndex):
    predictions_ETS.index = predictions_ETS.index.to_timestamp()

if isinstance(predictions_naive.index, pd.PeriodIndex):
    predictions_naive.index = predictions_naive.index.to_timestamp()

# Crear la figura
fig = go.Figure()

# Agregar los datos originales
fig.add_trace(go.Scatter(x=original_data_naive.index, y=original_data_naive.
 ↪values,
                         mode='lines+markers', name='Total',␣
 ↪line=dict(color='#1f77b4')))

# Agregar las predicciones
fig.add_trace(go.Scatter(x=predictions_ETS.index, y=predictions_ETS.values,
                         mode='lines+markers', name='ETS',␣
 ↪line=dict(color='green')))

# Agregar los residuos
fig.add_trace(go.Scatter(x=predictions_naive.index, y=predictions_naive.values,
                         mode='lines+markers', name='Naive',␣
 ↪line=dict(color='#ff7f0e')))

# Configurar el diseño del gráfico
fig.update_layout(title='Comparación de Datos Originales, Predicciones y␣
 ↪Residuos',
                  #   xaxis_title='Fecha',
                   #  yaxis_title='Valor',
                    # legend_title='Series',
                     template='plotly_white')

# Mostrar el gráfico
fig.show()
```



[78]:
```
future_predictions_naive = predict_model(naive_model,fh=4)
print(future_predictions_naive)
```

```
            y_pred
2024-06  18809455.0
2024-07  18809455.0
2024-08  18809455.0
2024-09  18809455.0
```

APÉNDICE B

app.py

```python
import streamlit as st
import pandas as pd
from pycaret.time_series import load_model, predict_model
from datetime import timedelta

# Configuración de la página en Streamlits
st.set_page_config(page_title="Predicción de Cantidad de Tarjetas de Crèdito", layout="wide")

# Título de la aplicación
st.title("Predicción de Cantidad de Tarjetas de Crédito")

# Menú de opciones
opcion = st.sidebar.selectbox("Seleccione una opción", ["DASHBOARD", "PREDICCIONES"])

# Enlace al dashboard de Power BI
power_bi_url = "https://app.powerbi.com/view?r=eyJrIjoiNzg1ZjE1OTktMDNiYS00NDA4LTg4MmMt↩
OTYxZWEzNzhiN2I5IiwidCI6ImI3YWY4Y2FmLTgzZDgtNDY0NC04NWFlLTMxN2M1NDUyMjNjMSIsImMiOjR9"  #
Reemplaza XXXX con tu enlace de embebido de Power BI

# Opciones del menú
if opcion == "DASHBOARD":
    # Mostrar el dashboard embebido
    st.subheader("Dashboard Interactivo")
    st.markdown(f'<iframe title="DashboardTesis" width="1024" height="612"
src="https://app.powerbi.com/view?r=eyJrIjoiNzg1ZjE1OTktMDNiYS00NDA4LTg4MmMt↩
OTYxZWEzNzhiN2I5IiwidCI6ImI3YWY4Y2FmLTgzZDgtNDY0NC04NWFlLTMxN2M1NDUyMjNjMSIsImMiOjR9"
frameborder="0" allowFullScreen="true"></iframe>', unsafe_allow_html=True)

elif opcion == "PREDICCIONES":
    # Título de la sección de predicciones
    st.subheader("Proyectar la cantidad de tarjetas")

    # Selección de entidad financiera
    st.sidebar.header("Seleccione la Entidad Financiera")
    entidad = st.sidebar.selectbox("Entidad Financiera", ["Todo", "Diners Club",
"Produbanco"])

    # Cargar el modelo y los datos correspondientes según la selección
    if entidad == "Todo":
        modelo_path = 'datos_deploy/modelTesis_Todo'
        datos_path = 'datos_deploy/Datos_Todo.csv'
    elif entidad == "Diners Club":
        modelo_path = 'datos_deploy/modelTesis_Diners Club'
        datos_path = 'datos_deploy/Datos_Diners Club.csv'
    elif entidad == "Produbanco":
        modelo_path = 'datos_deploy/modelTesis_Produbanco'
        datos_path = 'datos_deploy/Datos_Produbanco.csv'

    # Cargar el modelo seleccionado
    try:
        modelo = load_model(modelo_path)
        st.write(f"Modelo cargado: {modelo_path.split('/')[-1]}")
```

```
    except FileNotFoundError:
        st.error("Modelo no encontrado. Verifique la ruta y vuelva a intentarlo.")

    # Cargar los datos históricos seleccionados
    try:
        data = pd.read_csv(datos_path, index_col='Fecha', parse_dates=True)
        st.subheader(f"Valores reales de la entidad: {entidad}")
        st.line_chart(data)
    except FileNotFoundError:
        st.error("Datos no encontrados. Verifique la ruta y vuelva a intentarlo.")

    # Configuración del horizonte de predicción en la barra lateral
    st.sidebar.header("Parámetros de Predicción")
    periodos_prediccion = st.sidebar.slider("Horizonte de Predicción (número de periodos)", min_value=1, max_value=24, value=3)

    # Realizar la predicción cuando se presiona el botón
    if st.button("Generar Predicción"):
        try:
            # Realizar la predicción con el modelo cargado
            prediccion = predict_model(modelo, fh=periodos_prediccion)

            # Renombrar la columna 'y_pred' a 'Predicción'
            prediccion.rename(columns={'y_pred': 'Predicción'}, inplace=True)

            # Generar índice futuro para las predicciones, asegurándose de que las fechas sean el primer día de cada mes
            prediccion.index = pd.date_range(start=data.index[-1] + timedelta(days=1), periods=periodos_prediccion, freq='MS')  # 'MS' es Month Start

            # Mostrar la tabla con los valores predichos
            st.subheader("Valores Predichos")
            st.write(prediccion[['Predicción']])  # Mostrar solo las predicciones con la nueva etiqueta

            # Concatenar los datos históricos y las predicciones para visualización
            data_completa = pd.concat([data, prediccion[['Predicción']]], axis=0)

            # Mostrar las predicciones y los datos históricos juntos
            st.subheader("Predicciones y Datos de Entrenamiento")
            st.line_chart(data_completa)

        except Exception as e:
            st.error(f"Ha ocurrido un error durante la predicción: {e}")
# Sección de autor
st.markdown("""
    ---
    **Autor:** Edwin Maita 2024  xavimait@espol.edu.ec
    *Desarrollador de la solución de predicción de cantidad de tarjetas de crédito.*
    Esta aplicación utiliza modelos de predicción para proyectar la cantidad de tarjetas de crédito en diferentes entidades financieras.
""")
```