Prototipo software para la caracterización y predicción de la deserción en la UNAD a partir del uso de modelos de series de tiempo

Gabriel Elías Chanchí Golondrino

Asesora

Dayana Alejandra Barrera Buitrago

Universidad Nacional Abierta y a Distancia UNAD

Escuela de Ciencias Básicas, Tecnología e Ingeniería ECBTI

Especialización en Ciencia de Datos y Analítica

2024

Dedicatoria

A mi hijos Celeste y Samuel por ser la inspiración y motivación diaira. A Magaly por su apoyo constante. A mis padres y a Dios por su inmenso amor cada día.

Agradecimientos

Una vez culminado el trabajo quiero agradecer a la profesora Dayana Alejandra Barrera Buitrago por sus orientaciones y apoyo constante en el desarrollo del presente proyecto.

Un agradecimiento especial también para los profesores Rafael Gaitán Ospina y Miguel Angel Vargas Valencia por la motivación y conocimientos socializados en los cursos de la especialización.

Así mismo un agradecimiento especial a la UNAD por brindarme la posibilidad de formarme de manera flexible en un área tan interesante y desafiante como la ciencia de datos.

Resumen

La deserción es un indicador clave de la calidad educativa, por lo cual es imperativo para las Instituciones Educativas diseñar estrategias que la reduzcan, contribuyendo a la mejora de la retención estudiantil y a la consecución de los objetivos académicos de los estudiantes. Considerando que las investigaciones de deserción en Colombia se han enfocado en métodos de aprendizaje automático sobre conjuntos de datos de educación presencial, este proyecto presenta un enfoque diferente, basado en modelos de series de tiempo para el análisis de la deserción en la Universidad Nacional Abierta y a Distancia (UNAD). Específicamente, se propuso el desarrollo de un sistema software para la caracterización y predicción de la deserción en la UNAD a partir del uso de modelos de series de tiempo. Metodológicamente, se realizó una adaptación a cuatro fases de la metodología CRISP-DM: F1. Comprensión del modelo de negocio y de los datos, F2. Preparación de los datos, F3. Modelado y evaluación del modelo y F4. Despliegue del modelo. En la fase F1 se realizó la caracterización y reconocimiento de los datos de deserción obtenidos a partir del SPADIES. En la fase F2 se realizararon los procesos de preprocesamiento, división de conjuntos de entrenamiento y prueba, así como la determinación de los parámetros d, p y q del modelo ARIMA. En la fase F3 se ajustó el modelo con los datos de entrenamiento, teniendo en cuenta los parámetros determinados. Así mismo, se evaluó la efectividad del modelo teniendo en cuenta el conjunto de prueba. Por último en la fase 4, se realizó el despliegue del modelo en una herramienta software que tendrá por objetivo la predicción de los datos de deserción para la UNAD en semestres futuros. El modelo y el prototipo de software propuesto, pretenden erigirse como herramientas innovadoras útiles para para la toma de decisiones estratégicas con respecto a la retención estudiantil por parte de las autoridades académicas de la UNAD.

***Palabras clave*:** ARIMA, deserción, modelo predictivo, series de tiempo.

Abstract

Dropout is a key indicator of educational quality, making it imperative for Educational Institutions to design strategies to reduce it, thereby contributing to improved student retention and the achievement of students' academic goals. Considering that dropout research in Colombia has primarily focused on machine learning methods applied to datasets from face-to-face education, this project presents a different approach based on time series models for analyzing dropout at the National Open and Distance University (UNAD). Specifically, the project proposed the development of a software system for characterizing and predicting dropout at UNAD using time series models. Methodologically, an adaptation of the CRISP-DM methodology into four phases was carried out: F1. Business and data understanding, F2. Data preparation, F3. Modeling and model evaluation, and F4. Model deployment. In phase F1, the characterization and recognition of dropout data obtained from SPADIES were conducted. In phase F2, preprocessing processes, training and test set division, and the determination of the ARIMA model parameters d, p, and q were performed. In phase F3, the model was adjusted using the training data, taking into account the determined parameters. Additionally, the model's effectiveness was evaluated using the test set. Finally, in phase F4, the model was deployed in a software tool designed to predict dropout data for future semesters at UNAD. The proposed model and software prototype aim to serve as innovative tools to support strategic decision-making regarding student retention by UNAD's academic authorities.

***Keywords*:** ARIMA, dropout, predictive model, time series.

Tabla de Contenido

Lista de Tablas

Lista de Figuras

Lista de Apéndices

Introducción

La deserción académica se refiere al proceso en el que los estudiantes abandonan sus estudios antes de finalizar sus programas de grado. Este fenómeno representa un desafío importante para las universidades, ya que no solo impacta en su sostenibilidad financiera, sino también en el futuro académico y profesional de los estudiantes (Kok et al., 2024) . En este sentido es crucial entender y enfrentar esta problemática para diseñar estrategias efectivas de retención que mejoren las tasas de graduación (Nemtcan et al., 2020). Esto implica el análisis de múltiples factores que influyen en el abandono y la implementación de modelos predictivos que permitan identificar de manera temprana a los estudiantes con mayor riesgo de desertar.

De este modo, se ha identificado a partir del estado del arte que factores como la edad, el sexo, los antecedentes académicos y las capacidades financieras influyen significativamente en las tasas de deserción (Sani et al., 2022). Así mismo, se ha identificado que el bajo rendimiento en las materias de los primeros años es uno de los principales factores que influyen en la deserción, lo que pone de relieve la necesidad de una intervención temprana (Kok et al., 2024). En este mismo sentido, a nivel de los factores sociocognitivos se ha identificado que la autoeficacia académica y la integración de los estudiantes desempeñan un papel mediador en las intenciones de abandono escolar, lo que afecta a las decisiones de los estudiantes de abandonar los estudios o transferirse (Nemtcan et al., 2020).

A nivel del estado del arte son diferentes los modelos predictivos que se han utilizado para caracterizar la deserción académica, como en el caso de los aportes presentados en (Alcauter et al., 2023; Sani et al., 2022) donde se usan técnicas como Random Forest y Decision Trees para predecir el abandono de los alumnos, siendo el modelo Random Forest uno de los mas destacados en la predicción de la deserción en varios estudios. En este mismo orden de

ideas, el uso del aprendizaje supervisado ha sido ampliamente difundido en la clasificación de un estudiante como desertor o no no desertor o en la predicción de los niveles de riesgo de deserción, de tal modo que son escasos los trabajos enfocados en la caracterización de la tasa porcentual de deserción académica.

Dado que la deserción es monitoreada periódicamente puede ser caracterizada como una serie temporal, contexto en el cual los modelos ARIMA son uno de los enfoques más difundidos, al ser reconocidos como un modelo predicitivo eficaz para la caracterización y análisis de series temporales. Este modelo se basa en la premisa de que los valores futuros de la serie temporal pueden ser predichos a partir de la combinación lineal de los valores pasados y los errores pasados (Challa et al., 2020; Ugoh et al., 2022). A partir de lo anterior, en este trabajo, se propuso como principal aporte un modelo basado en series de tiempo tipo ARIMA para la caracterización y predicción de la tasa de deserción de la UNAD. Este modelo permitió generar pronósticos para los próximos años, los cuales pueden servir como base fundamental para que las autoridades educativas de la UNAD tomen decisiones estratégicas orientadas a la mitigación de la deserción. Estos pronósticos brindan una visión clara y anticipada de las tendencias futuras, facilitando la implementación de políticas más proactivas y dirigidas. Además, se desarrolló un prototipo software en Python con una interfaz gráfica que simplifica la carga de datos, el ajuste de los parámetros del modelo ARIMA, la evaluación de diferentes configuraciones del modelo y la predicción de la tasa de deserción. Esto proporciona a las autoridades una herramienta práctica para monitorear y gestionar de manera eficiente este indicador clave. La adaptación de la metodología CRISP-DM guió cada fase del desarrollo del modelo, asegurando un proceso estructurado y confiable, lo que refuerza aún más su utilidad para la toma de decisiones informadas y fundamentadas.

Tanto el modelo propuesto como el prototipo software implementado no solo pretenden ser una referencia para la comunidad científica en el estudio y caracterización de la deserción académica, sino que también presentan una alta versatilidad para ser aplicados en diferentes contextos educativos. Las fases y metodologías empleadas en este trabajo, como el uso del modelo ARIMA y la adaptación de la metodología CRISP-DM, son fácilmente extrapolables a otros niveles educativos e instituciones. Este enfoque permite ajustar el modelo a la particularidad de los datos disponibles, adaptando los parámetros del ARIMA según las características específicas de cada conjunto de datos. De este modo, el prototipo software se convierte en una herramienta flexible que puede ser utilizada por otras universidades, centros de formación o incluso en estudios comparativos entre distintos sistemas educativos, lo que facilita una mayor comprensión del fenómeno de la deserción y la implementación de estrategias más efectivas para reducirla.

Así, la principal novedad de este proyecto radica en la integración de modelos tipo ARIMA para la caracterización y predicción de la deserción en la UNAD, abordando un enfoque poco explorado en el estado del arte. Mientras que los trabajos previos se han centrado en identificar si un estudiante es un potencial desertor mediante el análisis de variables sociales, económicas y demográficas, este proyecto se diferencia al proponer la caracterización detallada de la curva de deserción a lo largo del tiempo. Además, el software desarrollado, con su interfaz gráfica intuitiva, no solo facilita la caracterización de la deserción de la UNAD, sino que también permite identificar el modelo estadístico más adecuado para los datos disponibles y generar predicciones as sobre el comportamiento de la deserción en los próximos semestres.

Planteamiento del Problema

La deserción estudiantil, es un fenómeno de gran importancia en las instituciones de educativas, dado que las decisiones estratégicas enfocadas en su caracterización y disminución, contribuyen con el incremento de la cobertura, así como al mejoramiento de la calidad, pertinencia y eficiencia en la educación (Ministerio de Educación Nacional, 2009). La deserción estudiantil puede definirse como el estado de un estudiante que de manera voluntaria o involuntaria no aparece matriculado por 2 o más periodos académicos de manera consecutiva en el programa en el que se matriculó, sin aparecer en los registros como graduado o retirado por motivos disciplinarios (MEN (Ministerio de Educación Nacional), n.d.; Romero-Contreras et al., 2022). Así mismo, de acuerdo con lo presentado en (Sáez et al., 2020; Vega-García et al., 2014; Velez, A.; López, 2004), la deserción puede ser concebida como el abandono de las actividades escolares antes de terminar algún grado o nivel educativo. Aunque la deserción académica se relaciona directamente con otros factores como la repitencia y el rezago académico, este fenómeno puede tener explicaciones más allá de lo académico, de tal modo que puede estar relacionada también con los aspectos demográficos, económicos y sociales (Aldeman & Szekely, 2017; Ó. Espinoza et al., 2021; Noltemeyer et al., 2015).

Con la gran difusión de la inteligencia artificial (IA) y las diferentes herramientas que posibilitan su implementación, son diferentes los trabajos que se han realizado para la caracterización y predicción de la deserción en diferentes instituciones educativas. Así en (Cruz et al., 2022) los autores realizan una revisión por la literatura sobre la aplicación de técnicas de machine Learning aplicadas a la caracterización de la deserción, obteniendo que se han aplicado diferentes modelos de aprendizaje supervisado y no supervisado y determinando que los factores de tipo demográfico tienen una gran influencia sobre la deserción. En (Valero Cajahuanca et al.,

2022) se evalúa y determina el mejor modelo de machine Learning para la predicción de la deserción en la UNITELS de Perú, partiendo de un dataset con variables socioeconómicas y académicas y obteniendo una precisión del 91% con el modelo KNN. En (Gutierrez-Villareal et al., 2021) los autores hacen de técnicas de aprendizaje no supervisado y análisis de componentes principales para la caracterización de la deserción en una Institución Universitaria de Bogotá, obteniendo que las dificultades económicas son la principal causa de deserción en hombres con un 67.6%. En (Chinkes, 2018) los autores hacen uso de modelo predictivo basado en redes neuronales para la caracterización y predicción de la deserción en la Facultad de Ciencias Económicas de la Universidad de Buenos Aires entre 2011 y 2013, obteniendo una precisión entre el 90 y el 95%. En (Rivera Vergaray, 2021) los autores evalúan diferentes modelos de aprendizaje supervisado para predecir la deserción en la Universidad Nacional Intercultural de la Amazonia, haciendo uso de un dataset con variables asociadas al perfil académico, los aspectos económicos y el rendimiento académico, obteniendo como resultado que el modelo KNN es el de la precisión más alta con un 88.84%. En (Ayala-Yaguara et al., 2019) los autores aplican modelos de aprendizaje supervisado en la caracterización de la deserción en estudiantes de Ingeniería de Sistemas de la Universidad de Cundinamarca, a partir de un dataset de aspectos socioeconómicos y familiares, obteniendo que el modelo de regresión logística es el que presenta una mejor precisión con un 71%. En (Kuz & Morales, 2023) los autores evaluaron diferentes modelos de aprendizaje supervisado en un dataset que incluye factores de tipo social, demográfico, financiero, geográfico y familiar asociados a una Universidad privada de México, obteniendo que el modelo de árboles de decisión obtuvo la más alta precisión con un 93.33%. En (Smith Uldall & Gutiérrez Rojas, 2022) los autores compararon diferentes modelos de machine learning y redes neuronales para la predicción de la deserción en estudiantes de nivel escolar en

Chile, obteniendo que el modelo basado en redes neuronales tiene la mejor precisión con un 93.80%. En (Urteaga et al., 2020) los autores buscaron determinar el mejor modelo de machine learning y redes neuronales que caracteriza la deserción y las perdidas económicas generadas por esta en los cursos de extensión de la Universidad Tecnológica de Argentina, obteniendo que los modelos evaluados presentan una precisión superior al 90%. En (Caselli Gismondi & Urrelo Huiman, 2021), los autores determinaron los mejores atributos que permiten conformar un dataset para la aplicación de modelos predictivos que caracterizan la deserción en la Universidad Nacional de Santa, obteniendo un total de 18 variables de tipo demográfico y académico.

Una vez revisados los trabajos anteriores, es posible apreciar dos aspectos importantes: a) La mayoría de los trabajos presentados se han enfocado en caracterizar la deserción en educación presencial haciendo uso de modelos de aprendizaje supervisado y/o redes neuronales; b) Los modelos predictivos empleados, están basados en etiquetado, es decir a partir de un conjunto de variables de diferente tipo (académicas, demográficas, sociales, etc) se busca determinar si un estudiante podrá o no desertar. De este modo no se ha evidenciado la aplicación de modelos basados en series de tiempo que predigan el porcentaje de deserción a futuro en instituciones de educación superior a distancia. De acuerdo con lo anterior, en este proyecto se buscó responder la siguiente pregunta de investigación: ¿Cómo mejorar el análisis de la deserción estudiantil en la UNAD mediante el uso de modelos ARIMA para series de tiempo?.

La anterior pregunta de investigación, fue respondida mediante el desarrollo de un prototipo software soportado en un modelo tipo ARIMA para la caracterización y análisis de la predicción sobre las tendencias futuras de la deserción en la UNAD. El sistema software pretende servir de apoyo para la Vicerrectoría de Servicios a Aspirantes, Estudiantes y Egresados (VISAE) en cuanto a la definición de estrategias para fortalecer la retención en los diferentes

programas de la UNAD. Así mismo, se convierte en una herramienta de seguimiento para verificar y evaluar la efectividad de estas estrategias. En este sentido, mediante las predicciones obtenidas a partir de la herramienta software propuesta, se pretende mitigar posibles consecuencias para la UNAD, tales como la disminución en los índices de calidad académica, la pérdida de ingresos económicos derivados de las matrículas, la reducción en la credibilidad institucional frente a estudiantes y organismos acreditadores, así como el impacto negativo en el cumplimiento de su misión educativa y social, especialmente en comunidades vulnerables que dependen de la educación a distancia para su desarrollo profesional.

Justificación

El prototipo software propuesto pretende beneficiar a diferentes partes interesadas que permiten justificar su desarrollo . Así es importante mencionar que en primer lugar se busca beneficiar a la Vicerrectoría de Servicios a Aspirantes, Estudiantes y Egresados (VISAE), en cuanto a la toma de decisiones estratégicas enfocadas en la mitigación de la deserción en la Universidad Nacional Abierta y a Distancia en el mediano y corto plazo. De este modo, las predicciones obtenidas a partir del software y, por ende, a través del modelo en años futuros, permitirán anticipar el camino a seguir por parte de la VISAE en cuanto al manejo de la deserción. Además, el uso de series de tiempo en la predicción de la deserción brinda la posibilidad de identificar patrones cíclicos o tendencias ocultas a lo largo del tiempo, lo que permitirá ajustar políticas preventivas con mayor precisión. Al contar con un sistema automatizado, la VISAE podrá realizar análisis continuos y actualizados de la situación de los estudiantes, optimizando recursos y enfocando esfuerzos en intervenciones personalizadas y más eficaces, mejorando así la retención estudiantil y, en última instancia, contribuyendo al éxito académico de los estudiantes de la UNAD.

Una de las principales ventajas de anticipar la deserción académica y caracterizar los factores que la desencadenan es que se facilita el desarrollo de intervenciones proactivas que pueden abordar los problemas antes de que se manifiesten, lo cual se ha evidenciado en otras investigaciones del estado del arte (Martelo et al., 2017). Este enfoque preventivo no solo permite la creación de estrategias más efectivas para mejorar la retención, sino que también posibilita una asignación más eficiente de los recursos, ya que se podrá identificar con mayor precisión a los grupos de estudiantes en riesgo. Para la VISAE, contar con predicciones a través del sistema software significará tener una herramienta robusta para realizar análisis dinámicos y

basados en datos en tiempo real. Además, esto promoverá la colaboración entre las distintas áreas académicas y administrativas de la Universidad, generando una sinergia que fortalecerá tanto la calidad educativa como el acompañamiento estudiantil, contribuyendo a la mejora continua en la retención y el éxito académico.

Continuando con la justificación, el trabajo desarrollado permitirá a los futuros trabajos de la Especialización en Ciencia de Datos y Analítica, orientar sus temas de estudio o de investigación hacia la temática de la deserción, abordándola desde temas de investigación como pueden ser las series de tiempo o los enfoques de machine learning. De este modo, se pretende que este trabajo, abra un nuevo abanico de oportunidades de investigación para la especialización y sus grupos de investigación asociados. En este sentido, se pudo evidenciar a partir del estado del arte, que la mayoría de los estudios realizados a la deserción se centran en machine learning (Caselli Gismondi & Urrelo Huiman, 2021; Cruz et al., 2022; Gutierrez-Villareal et al., 2021; Kuz & Morales, 2023; Miranda et al., 2022).

Finalmente, con la construcción del prototipo software a propuesto, también se pretende aportar al desarrollo de futuras investigaciones dentro de los grupos de investigación que conforman las diferentes cadenas de formación de la Escuela de Ciencias Básicas, Tecnología e Ingeniería (ECBTI), de tal modo que el proyecto desarrollado trascienda las fronteras de la Especialización y pueda ser beneficiosa para extrapolar el modelo o el enfoque utilizado en otras investigaciones pertenecientes a otros programas de la Escuela en mención. De otra parte, se pretende que tanto el modelo propuesto como el software desarrollado puedan ser extrapolados a otras instituciones con el fin de caracterizar y hacer seguimiento a la deserción en estas, contribuyendo así al Objetivo de Desarrollo Sostenible 4: Garantizar una educación inclusiva, equitativa y de calidad y promover oportunidades de aprendizaje durante toda la vida para todos.

De este modo, este aporte está alineado con la meta de reducir las desigualdades en el acceso a la educación y mejorar los indicadores de retención escolar en diversas instituciones.

Objetivos

Objetivo General

Construir un prototipo de software para la caracterización y predicción de la deserción en la UNAD a partir del uso de modelos basados en series de tiempo.

Objetivos Específicos

Analizar los datos históricos de deserción de la UNAD obtenidos a partir de la plataforma SPADIES del MEN, para la identificación de patrones y variables clave.

Identificar los parámetros óptimos del modelo ARIMA, mediante la evaluación de la estacionariedad, la autocorrelación y otras características de las series de tiempo.

Determinar el modelo de series de tiempo ARIMA óptimo para la predicción de la deserción en la UNAD en semestres futuros.

Desarrollar un prototipo funcional de software para la predicción de la deserción de la UNAD a partir del modelo ajustado.

Marco de Referencia

En esta sección se presenta el marco teórico, el marco conceptual y el estado del arte considerado en el presente trabajo. Dentro del marco teórico se tuvo en cuenta la temática de deserción estudiantil y lo referente a los sistemas de seguimiento a la deserción en Colombia. En el marco conceptual se consideraron los conceptos de series de tiempo y modelos de series de tiempo tipo ARIMA.

Marco Conceptual

A nivel del marco teórico, en este proyecto se tuvo en cuenta como contexto principal la deserción estudiantil, de tal modo que a continuación se describen a partir del estado del arte, sus diferentes definiciones y posibles causas. Del mismo modo se presentan las generalidades de la plataforma del Ministerio de Educación Nacional para el seguimiento de la deserción.

Deserción Estudiantil

Teniendo en cuenta que la educación corresponde a una necesidad primordial para toda sociedad que tenga como meta principal el crecimiento y el desarrollo, es fundamental prestar atención a fenómenos como la deserción en el contexto universitario, ya que en esta etapa los individuos tienen la capacidad de decidir y sentar las riendas de su vida profesional (Dávila-Morán et al., 2022). En este sentido, la deserción estudiantil representa un fenómeno de gran relevancia en las instituciones de educativas, dado que todo esfuerzo enfocado en su caracterización y disminución contribuye con el incremento de la cobertura, así como al mejoramiento de la calidad, pertinencia y eficiencia en la educación (Ministerio de Educación Nacional, 2009).

La deserción estudiantil puede ser entendida como el estado de un estudiante que de manera voluntaria o involuntaria no aparece matriculado por 2 o más periodos académicos

consecutivos dentro del programa en el que se matriculó, sin aparecer en los registros como graduado o retirado por motivos disciplinarios (MEN (Ministerio de Educación Nacional), n.d.; Romero-Contreras et al., 2022). Así mismo, de acuerdo con lo propuesto en (Sáez et al., 2020; Vega-García et al., 2014; Velez, A.; López, 2004) la deserción puede definirse como el abandono de las actividades escolares antes de terminar algún grado o nivel educativo. Por otra parte, según (Pachay-López & Rodríguez-Gámez, 2021), la deserción escolar corresponde al abandono de las actividades académicas de un estudiante debido a diversas situaciones de tipo económico, social, ambiental o asociados a la salud. En este mismo orden de ideas, en (Rochin Berumen, 2021) además de definir la deserción, la clasifican a nivel espacial y temporal, de tal modo que a nivel temporal puede considerarse como: precoz, temprana y tardía, mientras que a nivel espacial esta se refiere a si cambia de programa o de institución.

La deserción escolar está directamente relacionada con otros factores tales como la repitencia y el rezago, sin embargo este fenómeno tiene un trasfondo más allá de lo académico (Alban & Mauricio, 2019). De este modo, aunque la deserción universitaria representa mediante cifras, la situación de estudiantes de diferentes contextos, se caracteriza por ser multidimensional, lo que implica que interrumpir un proceso formativo dentro de una carrera universitaria, no solo depende de aspectos económicos, sino también de otros aspectos tales como son los psicológicos; entre ellos, la motivación, satisfacción con el curso, autorregulación y expectativas de autoeficacia (Díaz-Mujica et al., 2019; Sommer & Dumont, 2011) y otros como los culturales y físicos (Dávila-Morán et al., 2022; Romero-Contreras et al., 2022). Así, de acuerdo con (Ó. Espinoza et al., 2021) son dos las categorías en las que se pueden agrupar los factores que inciden en la deserción escolar: variables extraescolares y variables intraescolares. Dentro de las variables extraescolares se encuentran la pobreza y la vulnerabilidad, la situación

socioeconómica, el desempleo, el origen étnico, la desintegración familiar y las limitadas expectativas con respecto a la educación de la familia (De Witte & Rogge, 2013; O. Espinoza et al., 2021; Foley et al., 2014; Ingrum, 2007; Peña & Toledo, 2017). De manera similar a nivel de las variables intraescolares que detonan la deserción se destacan el autoritarismo docente, los problemas comportamentales, el adulto centrismo y el bajo rendimiento académico (Darling-Hammond & Cook-Harvey, 2018).

Sistema de Seguimiento de la Deserción en Colombia

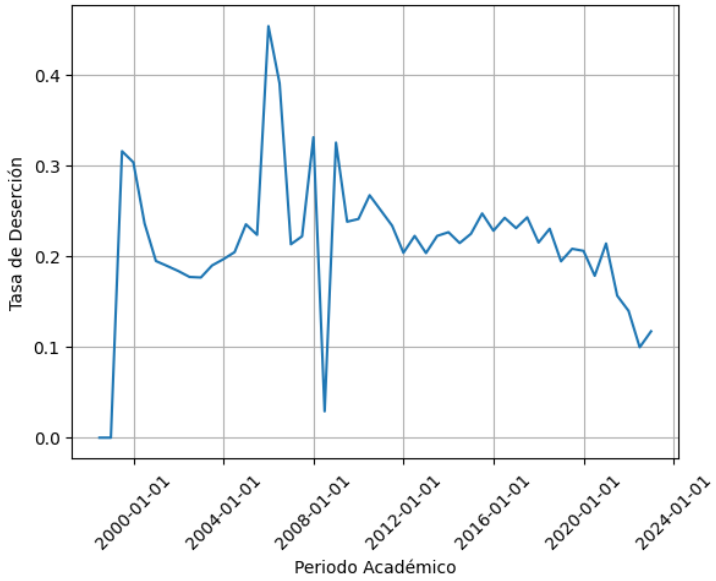
En Colombia, el Ministerio de Educación Nacional cuenta con la plataforma SPADIES (Sistema para la Prevención de la Deserción de la Educación Superior), la cual corresponde a un sistema especializado de información que permite el análisis de la permanencia en instituciones de educación superior a partir de los datos consolidados de la deserción en Universidades públicas y privadas de Colombia (MEN (Ministerio de Educación Nacional), 2023). Así, esta plataforma pone a disposición de los investigadores académicos estos datos de manera abierta, de cara a la conducción de investigaciones desde la perspectiva de la ciencia de datos.

Dentro de las variables que es posible obtener a partir del SPADIES se destaca la deserción anual y la tasa de deserción intersemestral, las cuales permiten realizar la evaluación de la deserción año tras año. La tasa de deserción anual tiene por objetivo medir el fenómeno de la deserción a corto plazo, permitiendo realizar un seguimiento sobre los esfuerzos de las instituciones de Educación Superior por mejorar los índices de deserción año a año. En este sentido, por ejemplo, para el año 2021, la tasa de deserción nacional se ubicó en 10.08%, lo que corresponde a 1.22 puntos porcentuales por arriba de la tasa obtenida para el año 2020, en donde se obtuvo un valor de 8.85% (Ministerio de Educación Nacional, 2023).

Cabe mencionar que teniendo en cuenta que en este proyecto se tendrán en cuenta los datos históricos de deserción correspondientes a la Universidad Nacional Abierta y a Distancia para la implementación y ajuste de un modelo predictivo basado en series de tiempo, a partir de la plataforma en línea del SPADIES; se descargaron los datos de deserción de la UNAD entre 1998 y 2022, los cuales incluyen la traza de deserción y retención durante el periodo en mención. De estas dos variables, en el presente proyecto se tomará en consideración la de deserción, la cual se cuenta con un total de 50 muestras correspondientes a los diferentes semestres académicos, tal como se muestra en la Figura 1. Así en el eje y del gráfico presentado en la Figura 1 se muestra el porcentaje de deserción asociado a cada uno de los 50 semestres académicos mencionados.

Figura 1

Datos Históricos de la Deserción en la UNAD



Marco Teórico

A nivel del marco conceptual en este proyecto se tuvo en cuenta el concepto de series de tiempo y de manera específica el referente a series de tiempo bajo el modelo ARIMA.

Series de Tiempo

Una serie de tiempo puede ser entendida como un conjunto de observaciones de una variable determinada, la cual es medida en puntos sucesivos a lo largo del tiempo o en periodos de tiempo sucesivos, de tal modo que dichas observaciones son obtenidas con el fin de caracterizar el comportamiento de la serie de valores y obtener pronósticos a futuro (Anderson et al., 2008). Del mismo modo, según (J. D. Melgarejo Garcilazo, 2020), un serie de tiempo puede ser definida como una secuencia de n observaciones o datos que son ordenados y equidistantes de manera cronológica, con respecto a una o varias características que tienen asociadas una unidad y cuyas valores son obtenidos en diferentes momentos (horas, días, meses, años). A partir del análisis de una serie de tiempo, es posible realizar pronósticos y conocer el patrón de comportamiento de la serie, de tal modo que sea posible prever la evolución a futuro de los datos, partiendo del supuesto de que las condiciones de la serie de tiempo no presentarán cambios significativos con respecto a los valores actuales y pasados. En este sentido, de acuerdo con (Canchano et al., 2019) uno de los aspectos más importantes en el estudio de series de tiempo, es la posibilidad de predecir los valores futuros de la serie, de tal modo que a partir de esas predicciones es posible proyectar los valores que tomará la variable objeto de estudio, de cara a la toma de decisiones estratégicas en un campo de estudio determinado.

En los recientes años, el modelado o caracterización de series de tiempo ha recibido mucha atención por parte de investigadores y académicos, dada la creciente necesidad de contar con herramientas que puedan apoyar la toma de decisiones y posibiliten superar las limitaciones

de tipo teórico, conceptual y practico, que presentan algunas de las herramientas convencionales para el análisis de los datos (Sánchez & Velásquez, 2010). En este sentido, para modelar una serie de tiempo es necesario construir de manera sistemática, una representación matemática que permita plasmar de manera total o parcial, el proceso generador de los datos, de tal modo que una vez construido dicho modelo, es posible realizar el pronóstico de valores futuros de la serie para un horizonte determinado (Sánchez-Sánchez & García-González, 2017). El estudio de las series de tiempo, se basa principalmente en la idea de descomponer la variación de una serie en diferentes componente básicos a saber (González León, 2018):

Tendencia. La tendencia de una serie temporal se refiere a la dirección en la que tiende a moverse a lo largo de un periodo específico. Esta tendencia revela el patrón general de crecimiento sostenido o declive a largo plazo dentro de la serie de datos a medida que transcurre el tiempo. En síntesis, la tendencia es la trayectoria principal, ya sea ascendente, descendente o estable, que la serie de tiempo sigue durante un intervalo prolongado.

Ciclo. El ciclo de una serie de tiempo representa las fluctuaciones que se producen alrededor de la tendencia principal. Describe las variaciones periódicas, ya sean al alza o a la baja, que ocurren mientras la serie se desplaza en torno a su nivel de tendencia a lo largo del tiempo. En resumen, el ciclo muestra las oscilaciones o movimientos recurrentes que se apartan de la tendencia a largo plazo de la serie temporal.

Variación Estacional. Las variaciones estacionales dentro de una serie de tiempo representan los patrones repetitivos que se manifiestan en intervalos regulares, generalmente en los mismos meses o trimestres de cada año, con una intensidad similar. Estos cambios estacionales reflejan las fluctuaciones predecibles que se repiten anualmente y que muestran una consistencia en términos de su ocurrencia y magnitud en momentos específicos del año. En

resumen, las variaciones estacionales son las fluctuaciones regulares que se observan de manera constante en ciertos períodos del año y que siguen un patrón predecible en términos de su fuerza y duración.

Fluctuaciones Irregulares. Las fluctuaciones irregulares dentro de una serie de tiempo son cambios impredecibles y no sistemáticos que muestran un comportamiento errático o sin un patrón claramente definido. Estos movimientos no siguen una secuencia o ciclo discernible, manifestándose de manera imprevisible, lo que dificulta la identificación de una tendencia clara o un período definido en el que ocurren. En resumen, las fluctuaciones irregulares son variaciones impredecibles que no obedecen a un patrón evidente y que se presentan de forma caótica o desordenada en la serie temporal.

A partir de las componentes descritas, se puede representar la señal Yt como se aprecia en la ecuación (1), donde Tt es la tendencia, Et corresponde a la componente estacional e It es el ruido o parte aleatoria que conforman la señal.

$$Yt = Tt + Et + It \quad (1)$$

Dadas las necesidades en el contexto académico de contar con herramientas libres y open source que permitan caracterizar y apoyar la toma de decisiones con respecto a las series de tiempo, en el contexto del software de código abierto se destaca la librería stats model de Python. La librería stats models es una herramienta poderosa en el análisis de series de tiempo debido a su amplia gama de funcionalidades. Cuenta con la licencia de código abierto BSD y ofrece métodos estadísticos y herramientas para explorar, visualizar y modelar datos temporales de manera efectiva. Entre sus ventajas, se incluyen la capacidad de realizar análisis de descomposición para identificar tendencias, estacionalidad y otros patrones temporales, ajustar modelos ARIMA (Autoregressive Integrated Moving Average) y SARIMA (Seasonal ARIMA)

para predecir datos futuros, así como realizar pruebas de raíz unitaria y pruebas de causalidad. Además, la flexibilidad de la librería permite a los analistas personalizar y ajustar modelos según las particularidades de los datos de series temporales, lo que resulta fundamental en la toma de decisiones informadas en campos como la economía, finanzas, ciencias ambientales y otros dominios donde el tiempo desempeña un papel crucial (Statsmodels, 2023).

Modelos de Series de Tiempo ARIMA

Los modelos ARIMA son una poderosa herramienta en el análisis de series temporales, al combinar modelos autorregresivos (AR) y modelos de promedio móvil (MA). Esta combinación, conocida como modelos ARIMA, ofrece la capacidad de capturar patrones complejos a corto y largo plazo presentes en los datos temporales. Al integrar información sobre tendencias pasadas, efectos de media móvil y autocorrelación, los modelos ARIMA facilitan la generación de predicciones precisas y proporcionan una comprensión profunda de la dinámica temporal dentro de un conjunto de datos (Arciniegas Paspuel et al., 2021). Los modelos ARIMA(p, d, q) constituyen una formulación matemática para comprender y predecir patrones en series temporales. Los parámetros p, d y q representan las dimensiones clave del modelo: p se refiere al componente autorregresivo que considera las observaciones pasadas, q corresponde al componente de promedio móvil que considera los errores pasados, mientras que d representa el grado de diferenciación para hacer estacionaria la serie temporal. Esta configuración permite a los modelos ARIMA capturar la estructura de las observaciones previas y generar pronósticos utilizando únicamente la información histórica disponible en los datos (Ayala Castrejon & Bucio Pacheco, 2020).

De acuerdo con lo anterior, la ecuación que define un modelo ARIMA esta definida por tres componentes clave: p, que representa los elementos autorregresivos, d, que denota el grado

de diferenciación para estabilizar la serie temporal, y q, que muestra la cantidad de elementos dados por errores aleatorios más uno. La estructura de estos modelos se visualiza a través de una ecuación que refleja cómo las observaciones pasadas, los errores residuales y las diferencias entre los datos influyen en las predicciones futuras. En la ecuación, p representa la dependencia de los valores pasados, d la cantidad de diferencias realizadas para lograr estacionariedad y q el impacto de los errores residuales pasados en las predicciones futuras. Esta configuración ayuda a modelar la serie temporal considerando múltiples aspectos que influyen en su evolución y en la proyección de sus valores venideros (ver Ecuación 2)(Ayala Castrejon & Bucio Pacheco, 2020; Pérez-Ramírez, 2010).

$$\nabla^d Y_t = \phi_1 \nabla^d Y_{t-1} + \phi_2 \nabla^d Y_{t-2} + \cdots + \phi_p \nabla^d Y_{t-p} + \theta_1 \varepsilon_{t-1} + \theta_2 \varepsilon_{t-2} + \cdots + \theta_q \varepsilon_{t-q} + \varepsilon_t \quad (2)$$

A nivel de las librerías de código abierto que son útiles para la implementación de modelos de series de tiempo se destaca la librería stats models. La librería stats models ofrece varias ventajas significativas para el análisis de series temporales mediante modelos ARIMA (Autoregressive Integrated Moving Average). Entre estas ventajas se encuentra su robusto conjunto de herramientas estadísticas que permiten a los usuarios implementar y ajustar fácilmente modelos ARIMA para pronósticos precisos. Statsmodels ofrece un entorno flexible que posibilita la personalización de los parámetros del modelo, lo que es crucial para adaptarse a las particularidades de diversos conjuntos de datos. Además, proporciona una amplia gama de métodos para diagnosticar la bondad del ajuste del modelo, identificar residuos y evaluar la precisión de las predicciones. Esta capacidad de diagnóstico facilita la mejora y validación de los modelos ARIMA, permitiendo a los analistas tomar decisiones más informadas en términos de predicciones y proyecciones temporales (Statsmodels, 2023).

Estado del Arte

En cuanto a la deserción estudiantil y su caracterización, son diferentes los trabajos en el contexto de América Latina y en el mundo, en los cuales se ha realizado la caracterización de la deserción estudiantil mediante el uso de modelos predictivos.

Así en el contexto de América Latina, en (Dávila et al., 2023) los autores caracterizar la deserción estudiantil de los estudiantes de la licenciatura en ciencias de la computación de una Unersidad de Ecuador a partir del uso de técnicas de aprendizaje automático y aprendizaje profundo, con el fin de identificar factores claves de tipo personal, académico y/o económico que puedan contribuir con la formulación de estrategias para la mitigación de la deserción en dicha Universidad. Del mismo modo en (Cardoso Melo & Sumika Hojo de Souza, 2023) se hizo uso de modelos predictivos basados en aprendizaje semisupervisado para la predicción del riesgo de abandono en los cursos de pregrado de una institución de educación superior brasileña, obteniendo como resultado una precisión en la clasificación del 90%. En este mismo sentido, en (Krüger et al., 2023) se hizo uso de técnicas de aprendizaje supervisado y de manera específica clasificadores para la predicción del abandono escolar partiendo de un dataset con datos pertenecientes a 19 esculas de Brasil, con lo cual se obtuvo una tasa de precisión de la clasificación del 95% y un conjunto de factores claves para caracterizar y mitigar la deserción en estas instituciones. En (Gutierrez-Pachas et al., 2023) se hizo uso de un dataset con datos correspondientes a tres años de seis programas académicos de una universidad latinoamericana, de tal modo que sobre dicho dataset se aplicaron tanto técnicas de aprendizaje automático para determinar si un estudiante abandonará o no abandonará la carrera, como técnicas de supervivencia para determinar cuando ocurrirá una deserción. Así mismo, en (Jiménez et al., 2023) se realizó un estudio basado en modelos de machine learning y redes neuronales de cara a

la predicción del riesgo de abandono y a la identificación de variables que inciden en la deserción, obteniendo que el modelo de Random Forest es el que tiene la mayor precisión y determinando que los factores claves que inciden en la deserción universitaria son la edad, la duración y el método de financiación. En (Gonzalez-Nucamendi et al., 2023) se hizo uso de técnicas de aprendizaje supervisado con el fin de predecir el riesgo de deserción en estudiantes de pregrado de una institución universitaria privada de México, obteniendo el mejor ajuste con el modelo de Random Forest y determinando que dentro de las variables claves que inciden en la deserción se encuentran: el rendimiento académico los primeros semestres, su promedio de calificación, la puntuación en el examen de ingreso, el número de horas de clase tomadas, la edad del estudiante y el estado de financiamiento. De otra parte, en (Alvarado-Uribe et al., 2022) se aplican métodos de aprendizaje de máquina con el fin de caracterizar y predecir la deserción estudiantil en el Tecnológico de Monterrey de México, haciendo uso de datos recopilados entre 2014 y 2020, buscando identificar las variables académicas, sociales y económicas que inciden en la deserción y que pueden considerarse en el ajuste los modelos. De manera similar, en (Karabacak & Yaslan, 2023) se compararan 10 modelos de aprendizaje supervisado con el fin de caracterizar y predecir la deserción en el Tecnológico de Monterrey de México, obteniendo que el modelo de XGBoost es el que obtiene el mejor rendimiento con una precisión del 92%. Por otra parte, en (Martelo et al., 2017) fueron implementadas técnicas de seires de tiempo y MULTIPOL con el fin de evaluar diferentes estrategias para la mitigación de la deserción académica dentro del programa de la Administración de Empresas de la Universidad de Cartagena en Colombia, obteniendo que la creación de modelos educativos flexibles, las inducciones pertinentes y las alianzas con instituciones de educación media son las estrategias más adecuadas para disminuir la deserción. En este mismo sentido, en (Prieto-Romero et al.,

2024), los autores proponen como aporte un modelo para la caracterización y predicción de la tasa de graduación en la Universidad de Cartagena a partir del uso de modelos de series de tiempo tipo ARIMA, obteniendo que el modelo con el mejor ajuste es el ARIMA (1,0,0) y la tasa de graduación para los próximos 10 años en la Universidad de Cartagena será de 0.5.

De otra parte, en otras latitudes del mundo también se han realizado estudios enfocados en la caracterización de la deserción académica. Así en (D. Patel et al., 2024) proponen un modelo de regresión linel múltiple y polinomial para la caracterización de las tasas de abandono escolar en instituciones educativas de la India, logrando métricas de rendimiento a nivel del coeficiente R^2 de 0.9976. De otra parte en (Arthana, 2024) se derrolló un estudio basado en aprendizaje de máquina para la caracterización predicción de la probabilidad de abandono escolar en Indonesia a partir de un dataset desequilibrado con datos entre 2013 y 2023, para lo cual hicieron uso de la técnicas de sobremuestreo por minoría sintéritca (SMOTE), obteniendo que el modelo KNN tuvo una precisión del 94%. En (Ortiz-Lozano et al., 2024) fueron usados modelos de aprendizaje automático y redes neuronales para la caracterización y predicción del abandono temprano de los estudiantes de primer año de la carrera de Adeministración de Empresas de la Universidad de Barcelona, obteniendo como precisión más alta un porcentaje de acierto en la clasificación del 77%. En (Sabbir et al., 2024) los autores desarrollan un estudio comparativo de modelos de aprendizaje automático para caracterizar y predecir tasas de abandono escolar en escuelas de Bangladesh, obteniendo que el rendimiento más alto se obtuvo en el modelo Random Forest con una precisión del 99%. En (Won et al., 2023) los autores proponen un nuevo enfoque para la predicción de la deserción basado en inferencia de lenguaje natural a partir del dataset de estudiantes del primer año de la Universidad Católica de Corea, obteniendo que el modelo mejora en un 9% la métrica F1 con respecto a los modelos

convencionales. Por otra parte, en (K. K. Patel & Amin, 2024) autores de la india proponen el uso de modelos predictivos basados en aprendizaje de máquina y redes neuronales para la caracterización y predicción de la deserción en cursos masivos y abiertos de la Universidad Abierta del Reino Unido, obteniendo que el modelo XGBoost fue el más eficaz con una precisión del 87%. En (Krüger et al., 2023) los autores comparan diferentes modelos de aprendizaje automático para caracterizar y predecir la deserción en estudiantes de la Facultad de Ciencias Sociales y Políticas de la Universidad Cristiana de Indonesia entre 2016 y 2018, obteniendo como resultado que el modelo Gradient Boosting se destacó por su capacidad de predecir con precisión los estudiantes en riesgo de abandonar, ayudando a identificar patrones clave y proponer intervenciones personalizadas. En (Song et al., 2023) los autores evalúan diferentes modelos de aprendizaje automático para predecir la deserción académica en la Universidad de Corea, obteniendo como resultado relevante que el modelo LightGBM obtuvo el mejor rendimiento con una precisión del 94%. En (Segura et al., 2022) aplican modelos de aprendizaje automático (máquinas de soporte vectorial y regresión logística) y redes neuronales para predecir la posible deserción de estudiantes matriculados en la Universidad Computense de Madrid, obteniendo que el modelo basado en redes neuronales obtuvo un mejor ajuste sobre el dataset estudiado. En (Martins et al., 2023) los autores se centraron en la aplicación de técnicas de aprendizaje automático para el desarrollo de modelos de predicción destinados a la detección temprana de los estudiantes en riesgo de abandonar los estudios en una universidad privada de Portugal, obteniendo que el modelo con el mejor ajuste fue el de Random Forest.

Los anteriores trabajos muestran como en el contexto de América Latina y en otras latitudes del mundo se han venido desarrollando diferentes modelos predictivos enfocados en la caracterización y predicción de la deserción académica, los cuales en su mayoría han hecho uso

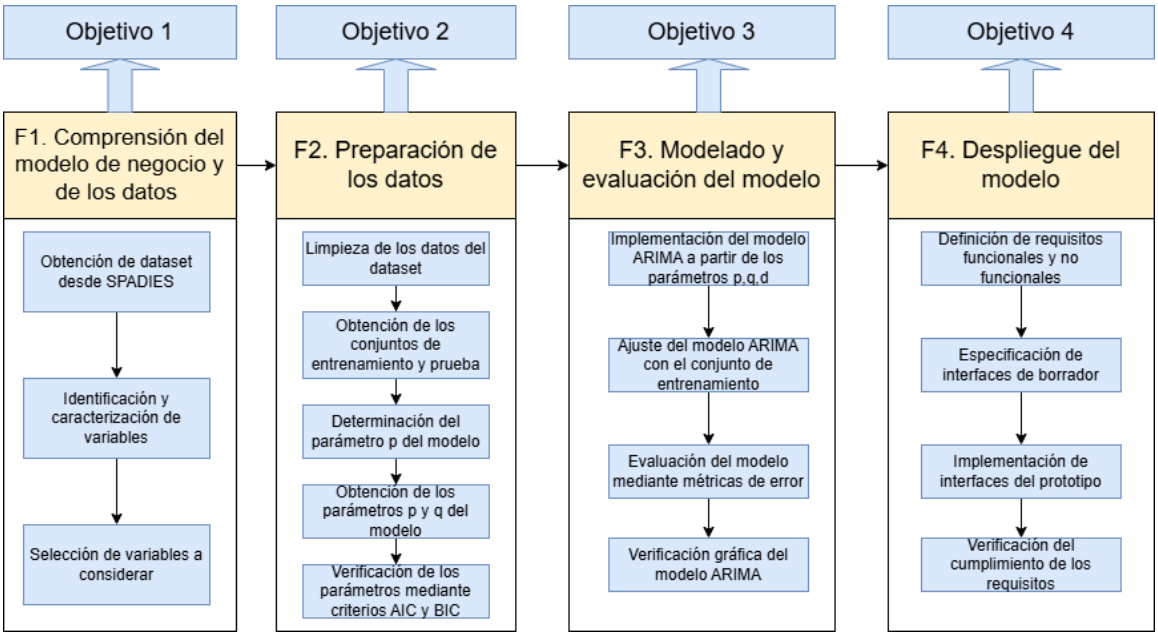
de algoritmos de aprendizaje automático y redes neuronales, los cuales fueron aplicados sobre datasets en los que de manera previa o a través de los mismos modelos se han determinado las variables relevantes. Así mismo cabe resaltar que en su mayoría estos trabajos se han centrado tanto en clasificar a estudiantes de diferentes niveles como desertores o no desertores, como en la clasificación del riesgo de deserción, de tal modo que estos modelos de aprendizaje automático en si no se han enfocado en el estudio de la tasa porcentual de deserción. Por otra parte, los modelos de series de tiempo identificados en el estado del arte se han centrado en caracterizar tasas de graduación y en analizar rendimiento académico, por lo que este trabajo presenta como principal aporte el uso de modelos de series de tiempo tipo ARIMA para la caracterización de la tasa de deserción en la UNAD, con el fin de obtener pronósticos que apoyen la toma de decisiones estratégicas encaminadas a mitigar la deserción. Así, este enfoque proporciona una visión más integral al ofrecer predicciones cuantitativas que permiten anticipar tendencias y facilitar una mejor planificación institucional.

Metodología

Para el desarrollo del presente proyecto se realizó una adaptación de la metodología para desarrollo de proyectos de ciencia de datos CRISP-DM (Dorado Bastidas et al., 2023; Espinosa Zúñiga, 2020; Shafique & Qaiser, 2014), de tal modo que se definieron cuatro fases a saber: F1. Comprensión del modelo de negocio y de los datos, F2. Preparación de los datos, F3. Modelado y evaluación del modelo, F4. Despliegue del modelo (ver Figura 2). En este proyecto fue escogida la metodología CRISP-DM dada que su estructura proporciona un enfoque sistemático que es esencial para la creación de modelos predictivos efectivos, ya que permite a los equipos de proyecto abordar de manera organizada y metódica los desafíos inherentes a la minería de datos y el aprendizaje automático(Ayele, 2020; Jaggia et al., 2020; Mariscal et al., 2010).

Figura 2

Metodología Considerada



En la fase 1 de la metodología, se realizó la obtención y reconocimiento de los datos de deserción a partir de la plataforma SPADIES del Ministerio de Educación Nacional, de tal modo que en esta fase se identificó el atributo o los atributos a considerar dentro del modelo de series de tiempo a utilizar, de tal modo que en este caso se seleccionó la tasa porcentual de deserción como variable a caracterizar y predecir. Esta fase de la metodología está relacionada directamente con primer objetivo del proyecto y tuvo asociadas las siguientes actividades:

A1.1 Obtención del dataset de deserción de la UNAD a partir del portal del SPADIES.

A1.2 Identificación y caracterización de las variables asociadas al dataset de deserción de la UNAD.

A1.3 Selección de las variables a considerar dentro del modelo de series de tiempo.

Dentro de la fase 2 de la metodología se realizaron las etapas de limpieza, selección de conjuntos de entrenamiento y pruebas del dataset (80% y 20%). Cabe mencionar que a nivel de la etapa de limpieza se realizó la imputación de periodos en cero a partir del valor de la media. Del mismo modo los conjuntos de entrenamiento y prueba fueron escogidos visualmente de tal modo que pudieran capturar la tendencia principal de la serie. Así mismo, se determinaron los parámetros p, d y q adecuados para la implementación del modelo de series de tiempo. Para la determinación del parámetro d, se tuvo en cuenta la prueba de estacionariedad de Dickey Fuller, mientras que para la obtención de los parámetros p y q se tendrá en cuenta la correlación y autocorrelación parcial. Esta fase está conectada directamente con el objetivo 2 y comprendió las siguientes actividades:

A2.1 Limpieza de los datos del dataset de deserción de la UNAD.

A2.2 Obtención de los conjuntos de entrenamiento y prueba a partir del dataset de deserción de la UNAD.

A2.3. Determinación del parámetro d del modelo a partir de la prueba de Dickey-Fuller.

A2.4. Obtención de los parámetros p y q del modelo mediante el uso de las gráficas de correlación y autocorrelación parcial de la serie.

A2.5. Verificación de los parámetros obtenidos a través del uso de los criterios AIC (Criterio de Información de Akaike) y BIC (Criterio de Información Bayesiano).

En la fase 3 de la metodología se realizará la implementación y ajuste del modelo de series de tiempo ARIMA a partir del conjunto de entrenamiento y teniendo en cuenta los parámetros d, p y q determinados (Ayala Castrejon & Bucio Pacheco, 2020). Así mismo, dentro de esta fase se realizará la evaluación del modelo, comparando las predicciones obtenidas con respecto al conjunto de prueba, para lo cual se hará uso de métricas de error (MAE, MSE y RMSE). Esta fase está directamente relacionada con el objetivo 3 del presente proyecto y comprende el desarrollo de las siguientes actividades:

A3.1 Implementación del modelo ARIMA a partir de los parámetros d, p y q.

A3.2 Ajuste del modelo ARIMA con el conjunto de entrenamiento.

A3.3 Evaluación de la efectividad del modelo ARIMA haciendo uso del conjunto de pruebas y las métricas de error.

A3.4 Verificación y comparación gráfica del modelo ARIMA con respecto a los datos reales de deserción.

Finalmente, en la fase 4 de la metodología se realizará el despliegue del modelo, a través de la implementación de una herramienta software para la predicción de la deserción de la UNAD en periodos futuros. De este modo, esta fase comprendió la especificación de requisitos, el diseño de las interfaces de alto nivel, la implementación del software y la verificación del

funcionamiento de este, de acuerdo a los requisitos definidos. Esta fase se relaciona directamente con el objetivo específico 4 y estuvo constituida por las siguientes actividades:

A4.1 Definición de los requisitos funcionales y no funcionales del sistema software.

A4.2 Especificación de la funcionalidad del prototipo software mediante interfaces de borrador.

A4.3 Implementación de las interfaces del prototipo de software e integración con el modelo ARIMA implementado.

A4.4 Verificación del cumplimiento de los requisitos funcionales del sistema software.

Resultados

Para asegurar el cumplimiento tanto de los objetivos, en esta sección se presenta el modo en el que fueron desarrolladas las diferentes actividades correspondientes a las fases de la metodología: F1. Comprensión del modelo de negocio y de los datos, F2. Preparación de los datos, F3. Modelado y evaluación del modelo, F4. Despliegue del modelo. Cabe mencionar que la fase 1 comprende las actividades relacionadas con el cumplimiento del objetivo 1, la fase 2 incluye las actividades asociadas al cumplimiento del objetivo 2, la fase 3 involucra las actividades correspondientes al cumplimiento del objetivo 3 y finalmente la fase 4 comprende las actividades asociadas al cumplimiento del objetivo 4.

Desarrollo de Actividades

A continuación se presenta el desarrollo en detalle de cada una de las actividades de la metodología del presente proyecto y que dan cuenta del proceso de todo el proceso que conlleva la construcción del modelo ARIMA y el prototipo de herramienta desarrollada.

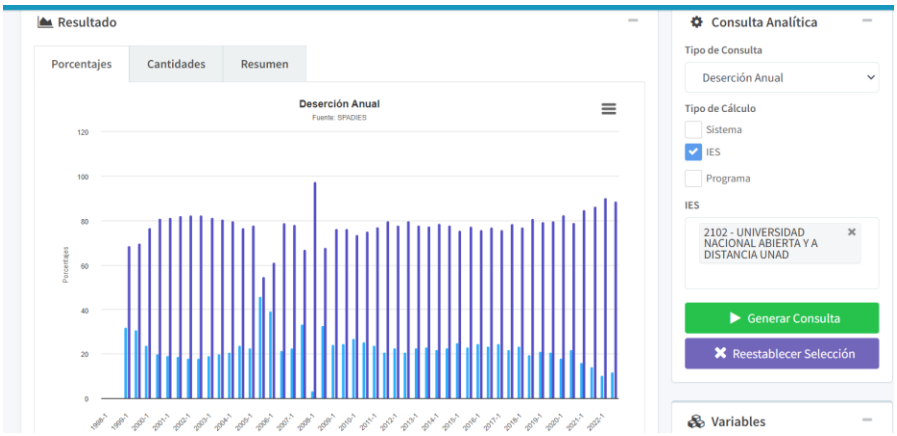
Obtención del Dataset de Deserción de la UNAD

En primer lugar a partir del portal de la plataforma SPADIES se realizó la consulta de los datos de deserción correspondientes a la UNAD, lo cual permitió obtener como resultado la tasa porcentual de deserción y redención desde 1998 hasta 2022, tal como se presenta en la Figura 3. La plataforma del SPADIES tiene por función monitorizar la deserción estudiantil en instituciones de educación superior, proporcionando un análisis detallado y en tiempo real de las tendencias de abandono académico. Entre sus ventajas, destaca la capacidad de identificar los factores de riesgo asociados a la deserción, lo que permite a las instituciones implementar políticas preventivas y mejorar la retención estudiantil. Además, SPADIES ofrece herramientas

para la segmentación de datos por género, región, y programa académico, facilitando la toma de decisiones basadas en evidencia.

Figura 3

Datos de Deserción Obtenidos del SPADIES

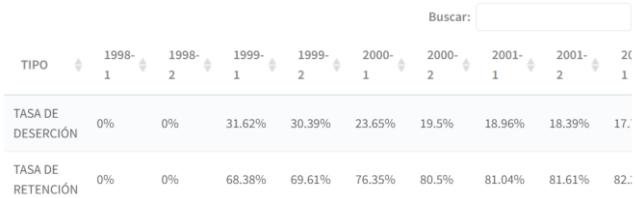


Identificación y Caracterización de las Variables del Dataset de Deserción de la UNAD

Tal como se mencionó de manera previa, a partir de la plataforma del SPADIES, es posible descargar un dataset que incluye 3 variables: el semestre académico, la tasa porcentual de deserción y la tasa porcentual de deserción. Lo anterior puede observarse en la Figura 4, donde se muestra el reporte que presenta la plataforma SPADIES del dataset, en el cual se aprecian los 3 atributos mencionados.

Figura 4

Dataset de Deserción Obtenidos del SPADIES



Así mismo, una vez identificados los 3 atributos del dataset obtenido a partir del SPADIES, en la Tabla 1 se presenta la descripción detallada de cada uno de estos atributos.

Tabla 1

Descripción de los Atributos del Dataset

Atributo	Descripción
Semestre	Corresponde al semestre académico en el cual fue realizada la medición de la tasa porcentual de deserción y la tasa porcentual de retención. Este atributo varía desde
Tasa de deserción	La tasa de deserción universitaria es el porcentaje de estudiantes que abandonan sus estudios antes de completar el programa académico en el que están matriculados. Este indicador refleja el nivel de abandono en una institución educativa y es clave para identificar problemas en el proceso formativo.
Tasa de retención	La tasa de retención universitaria mide el porcentaje de estudiantes que continúan con sus estudios de un semestre o año académico al siguiente, lo que indica el éxito de la institución en mantener a los estudiantes inscritos y comprometidos con su formación.

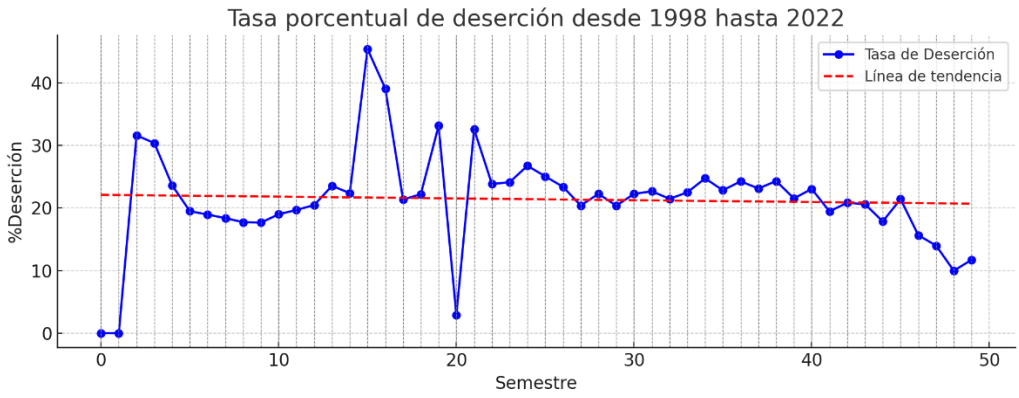
Nota. Datos recopilados por los autores de la investigación.

Selección de Variables a Considerar en el Modelo

Dado que los modelos de series de tiempo toman en consideración una sola variable con índice temporal y teniendo en cuenta el foco del trabajo, dentro del presente proyecto fue escogida la variable o atributo tasa porcentual de deserción. De este modo, en la Figura 5 se presenta una gráfica con la serie de tiempo dela tasa porcentual de deserción en la UNAD entre los años 1998 hasta 2022.

Figura 5

Tasa Porcentual de Deserción de la UNAD entre 1998 hasta 2022



A partir de la Figura 5, se observan fluctuaciones significativas a lo largo del tiempo, con picos notables alrededor del semestre 10 (2003-2), semestre 20 (2008-2) y semestre 25 (2011-1), donde la deserción supera el 40%. Después de este periodo, la tasa de deserción parece estabilizarse, aunque sigue mostrando variaciones entre semestres, con un descenso más notable hacia el semestre 50 (2022-2), donde la deserción se sitúa por debajo del 10%. Así mismo, la línea de tendencia, representada en rojo, indica que la tasa de deserción promedio se sitúa en torno al 20%, aunque algunos periodos, como en los semestres mencionados, están claramente por encima de este valor. Las caídas recientes sugieren una posible mejora en la retención de

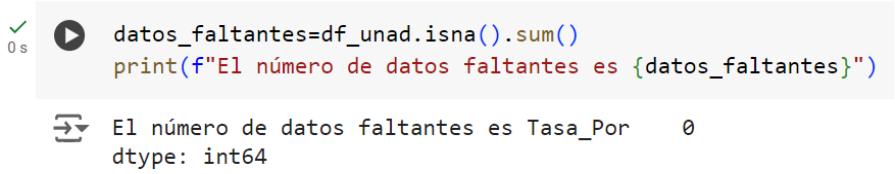
estudiantes, pero las oscilaciones pasadas señalan la necesidad de un análisis más profundo sobre las causas de estos picos. Para abordar esta complejidad, aplicar modelos de series de tiempo, como el modelo ARIMA (AutoRegressive Integrated Moving Average), puede ser una estrategia clave. Este tipo de modelo no solo permite capturar los patrones de tendencia y estacionalidad que han influido en la deserción en el pasado, sino que también ofrece una herramienta robusta para predecir futuras fluctuaciones en las tasas de deserción. Con este enfoque, sería posible ajustar de manera precisa los datos históricos, identificar factores subyacentes que expliquen los picos y caídas, y generar pronósticos que puedan ayudar a las instituciones a anticiparse a posibles aumentos en la deserción, permitiendo la implementación de estrategias proactivas para mejorar la retención estudiantil.

Limpieza de los Datos del Dataset de Deserción de la UNAD

A nivel de la limpieza de los datos, en primera instancia se procede con la identificación de valores faltantes o nulos de cara a realización de procesos de imputación, los cuales permiten mantener la integridad del conjunto de datos y evitar la pérdida de información importante. La imputación es clave para asegurar que los análisis y modelos no se vean sesgados por la ausencia de datos, permitiendo obtener resultados más precisos y confiables. De este modo en la Figura 6 se presenta la revisión realizada mediante la librería pandas al dataset con la tasa porcentual de deserción de la UNAD, de tal modo que se obtiene que el dataset no cuenta con valores NA o faltantes.

Figura 6

Revisión de Datos Faltantes en el Dataset



A pesar de que no se evidencian datos faltantes, en la Figura 5 se aprecia que los dos primeros valores se encuentran en cero. Estos valores atípicos pueden distorsionar la distribución de los datos y, en consecuencia, afectar negativamente la capacidad del modelo predictivo para capturar patrones subyacentes de la serie temporal. La presencia de ceros al inicio de la serie puede sesgar los resultados y reducir la precisión del modelo, especialmente si este depende de tendencias o comportamientos históricos para realizar predicciones. Por esta razón, se optó por realizar un proceso de imputación de estos valores, reemplazándolos por la media de la serie. De este modo en la Figura 7 se muestra como a partir de la librería pandas, los dos valores iniciales que están en cero son reemplazados por valores nulos o NA para posteriormente realizar la imputación de los datos a partir de la media. Así, para los dos semestres de 1998 la imputación realizada fue de 22.303.

Figura 7

Imputación de la Media en Valores Cero de la Serie

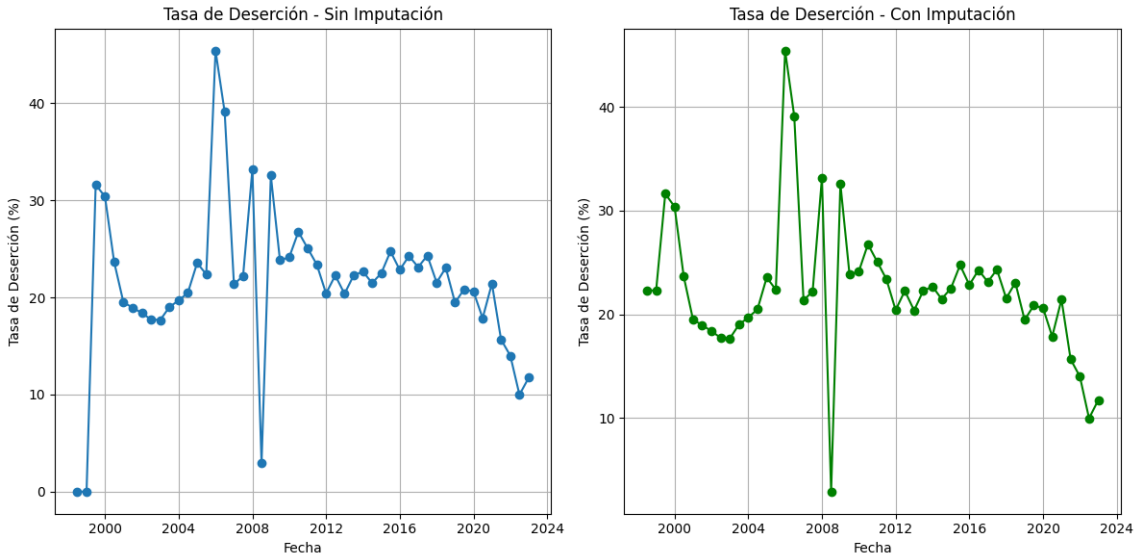
```
import numpy as np
import warnings
warnings.simplefilter(action='ignore', category=FutureWarning)

# Reemplazar los valores 0 por NaN
df_unad_def=df_unad.copy()
df_unad_def['Tasa_Por'].replace(0, np.nan, inplace=True)

# Calcular la media de los primeros valores conocidos
media = df_unad_def['Tasa_Por'].mean()

# Rellenar los NaN restantes con la media estimada o el valor que desees
df_unad_def['Tasa_Por'].fillna(media, inplace=True)

df_unad_def
```

	Tasa_Por
1998-06-30	22.303125
1998-12-31	22.303125
1999-06-30	31.620000

Finalmente, una vez fue realizada la imputación con la media se genera el gráfico de la Figura 8 en la cual se comparan la serie sin imputar con la serie imputada.

Figura 8

Comparación de las Series sin Imputar e Imputadas con la Media



Obtención de los Conjuntos de Entrenamiento y Prueba

A continuación, se procede a seleccionar el conjunto de entrenamiento y el conjunto de pruebas, el cual a diferencia del machine learning es un proceso que no se realiza de manera aleatoria sino secuencial. Esto se debe a que las series de tiempo tienen una dependencia temporal, donde los valores futuros están influenciados por los valores pasados. Una división aleatoria rompería esta estructura temporal, impidiendo que el modelo capture las tendencias y patrones inherentes en los datos. Por lo tanto, es fundamental preservar el orden cronológico al dividir los datos para garantizar la coherencia y validez en el proceso predictivo.

Figura 9

Separación en Conjunto de Entrenamiento y Pruebas

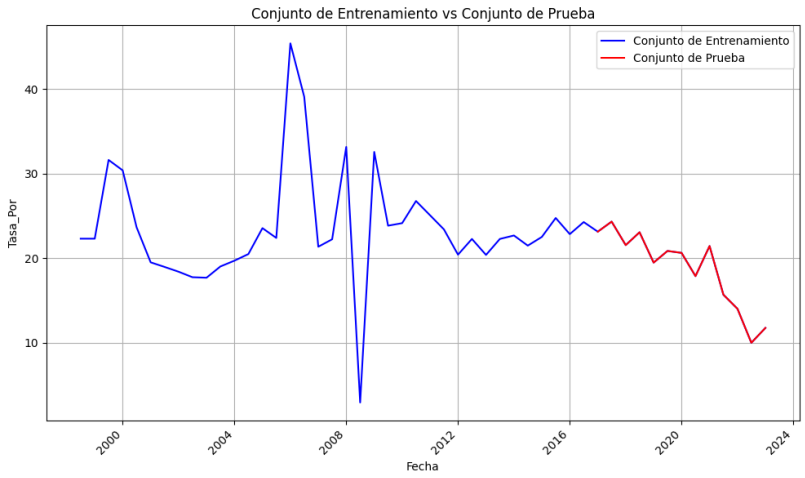
```
X=df_unad_def['Tasa_Por']
tam=len(df_unad_def['Tasa_Por'])
val=tam-int(tam*0.75)
train,test=X[0:-val], X[-val:]
train
```

1999-12-31	30.390000
2000-06-30	23.650000
2000-12-31	19.500000
2001-06-30	18.960000
2001-12-31	18.390000
2002-06-30	17.740000

Del mismo modo, en la Figura 10 se presenta la gráfica que incluye el conjunto de entrenamiento (color azul) y el conjunto de prueba (color rojo).

Figura 10

Conjunto de Entrenamiento y de Pruebas



Determinación del Parámetro d del Modelo

De cara a la estimación del parámetro d del modelo ARIMA, el cual hace referencia al número de veces en que debe ser diferenciada la serie para que cumpla con las condiciones de estacionariedad, se hizo uso de la prueba de Dickey-Fuller, la cual evalúa la presencia de una raíz unitaria en la serie temporal. Si el valor de la prueba es significativo, se rechaza la hipótesis nula de no estacionariedad, lo que indica que la serie es estacionaria en su forma actual. En caso contrario, se requiere realizar diferenciaciones adicionales hasta que la prueba de Dickey-Fuller muestre evidencia de estacionariedad, determinando así el valor adecuado de d para el modelo ARIMA. Para lo anterior se hizo uso de las ventajas provistas por la librería statsmodels de Python a través de su función addfuller la cual permite obtener el estadístico ADF y el p-value mediante el cual se concluye si se rechaza o no la hipótesis nula de no estacionariedad (ver Figura 11). Estos resultados son clave para tomar decisiones informadas sobre la necesidad de realizar diferenciaciones adicionales en el modelo ARIMA.

Figura 11

Prueba de Estacionariedad de Dickey Fuller

```
from statsmodels.tsa.stattools import adfuller
from numpy import log

result=adfuller(train.dropna())
print(f"ADF statistic:{result[0]}")
print(f"p-value: {result[1]}")
```

```
ADF statistic:-5.426772140566247
p-value: 2.9751524525302738e-06
```

Así, se aplicó la prueba de Dickey-Fuller sobre la serie original y sobre la serie diferenciada, obteniendo los resultados presentados en la Tabla 2.

Tabla 2

Resultados Prueba de Estacionariedad de Dickey-Fuller

Serie	Resultados
Original	ADF statistic:-5.426772140566247 p-value: 2.9751524525302738e-06
Primera diferenciación	ADF statistic:-7.060796471797739 p-value: 5.230107554146483e-10

Nota. Datos recopilados por los autores de la investigación.

Los resultados obtenidos de la prueba de Dickey-Fuller (ADF) para la serie original muestran un estadístico ADF de -5.4267 y un p-value de 2.97e-06. Dado que el p-value es significativamente menor al nivel de significancia típico de 0.05, se rechaza la hipótesis nula de no estacionariedad, lo que indica que la serie original es estacionaria. Así mismo, para la primera diferenciación, el estadístico ADF es de -7.0608 con un p-value de 5.23e-10, lo que también es mucho menor que 0.05, permitiendo rechazar la hipótesis nula de no estacionariedad. Sin embargo, el valor del estadístico ADF más negativo en la serie diferenciada sugiere que esta pasa la prueba de estacionariedad de forma más contundente, ya que un valor ADF más bajo indica una mayor evidencia de estacionariedad. De este modo, es posible concluir que ambas series pasan la prueba de estacionariedad, pero la serie diferenciada lo hace de manera más robusta.

Con el fin de reforzar los resultados obtenidos en la prueba de Dickey-Fuller con respecto a la estacionariedad de la serie, se evaluó de manera estadística si la serie original y la primera diferenciación tienen tendencia cero, varianza constante y autocorrelación constante, haciendo uso de las ventajas provistas por las librerías statsmodels y scipy. Para identificar la tendencia cero se realizó el cálculo de la tendencia de la serie mediante regresión lineal y se obtuvo la pendiente de dicha tendencia. En lo que respecta a la identificación de la varianza constante, se hizo uso de la prueba de heterocedasticidad de Breusch-Pagan. Finalmente, para la determinación de la autocorrelación constante, se obtuvieron los valores de correlación para los diferentes rezagos de la serie. El código realizado en Python en el cual se verifica el cumplimiento de las 3 condiciones es presentado en la Figura 12.

Figura 12

Evaluación de las Condiciones de Estacionariedad

```
import statsmodels.api as sm
from statsmodels.stats.diagnostic import het_breuschpagan
from statsmodels.graphics.tsaplots import acf
from scipy.stats import linregress

# 1. Cálculo de la tendencia usando regresión lineal
x = np.arange(len(train))  # Variables independientes (índices de tiempo)
slope, intercept, r_value, p_value, std_err = linregress(x, train)

# Imprimir el resultado de la pendiente (slope) para evaluar la tendencia
print(f"Pendiente (Tendencia): {slope}")

# 2. Prueba de Heterocedasticidad de Breusch-Pagan para la varianza constante
# Se necesita ajustar un modelo OLS (mínimos cuadrados ordinarios)
X = sm.add_constant(x)  # Añadir una constante a los datos independientes
ols_model = sm.OLS(train, X).fit()
bp_test = het_breuschpagan(ols_model.resid, X)

# Imprimir los resultados de la prueba de Breusch-Pagan
labels = ['Lagrange multiplier statistic', 'p-value', 'f-value', 'f p-value']
bp_results = dict(zip(labels, bp_test))
print("\nPrueba de Heterocedasticidad de Breusch-Pagan (Varianza Constante):")
for key, value in bp_results.items():
    print(f"{key}: {value}")

# 3. Cálculo de la autocorrelación usando ACF (autocorrelación)
acf_values = acf(train, fft=False)

# Imprimir los valores de autocorrelación
print("\nValores de Autocorrelación (ACF):")
for i, acf_val in enumerate(acf_values):
    print(f"Lag {i}: {acf_val}")
```

Como resultado de la aplicación del script presentado en la Figura 12, en la Tabla 3 se presentan los resultados obtenidos en las tres condiciones tanto para la serie original, como para la primera diferenciación.

Tabla 3

Resultados de la Verificación de las 3 Condiciones de Estacionariedad

Serie	Resultados
Original	Pendiente (Tendencia): -0.005 Prueba de Heterocedasticidad de Breusch-Pagan (Varianza Constante): Lagrange multiplier statistic: 0.309507328944044 p-value: 0.5779826654192144 Valores de Autocorrelación (ACF): Lag 0: 1.0 Lag 1: 0.07199890149027201 Lag 2: 0.07064484819698891 Lag 3: 0.09375296943008787 Lag 4: -0.17600964901585942 Lag 5: -0.30020608966387685 Lag 6: -0.01206318049293602 Lag 7: -0.14673384864130684 Lag 8: -0.10952595038132354 Lag 9: -0.031428980107899754 Lag 10: -0.07713922229202037 Lag 11: -0.049560520921074804 Lag 12: 0.0672150806649294 Lag 13: 0.1326495163744018 Lag 14: -0.03334839274944094 Lag 15: -0.03008579117531139

Serie	Resultados
Primera diferenciación	Pendiente (Tendencia): -0.0025 Prueba de Heterocedasticidad de Breusch-Pagan (Varianza Constante): Lagrange multiplier statistic: 0.0012692702361190022 p-value: 0.9715799233772848 Valores de Autocorrelación (ACF): Lag 0: 1.0 Lag 1: -0.49930221524728685 Lag 2: -0.009740324908728613 Lag 3: 0.15806650311301268 Lag 4: -0.08094109227373743 Lag 5: -0.22434255009370221 Lag 6: 0.22778204001440408 Lag 7: -0.09250654258042187 Lag 8: -0.022836036415646903 Lag 9: 0.0671039476329562 Lag 10: -0.03975817628063566 Lag 11: -0.04830644245856845 Lag 12: 0.02748270579114963 Lag 13: 0.12661038869793792 Lag 14: -0.09170008518836292 Lag 15: 0.02094267165022904

Nota. Datos recopilados por los autores de la investigación.

De acuerdo con los resultados obtenidos previamente, es posible observar con respecto a la serie original que la pendiente calculada a partir de la regresión lineal es de -0.0050, lo que indica una ligera tendencia descendente en la serie a lo largo del tiempo. Sin embargo, el valor de la pendiente es cercano a cero, lo que sugiere que la tendencia general es mínima. En consecuencia, aunque no hay una tendencia completamente nula, la inclinación es lo suficientemente baja como para considerar que la serie no presenta una tendencia significativa. Ahora bien, con respecto a la revisión de la varianza constante, los resultados de la prueba de heterocedasticidad de Breusch-Pagan muestran un p-value de 0.5779, lo que significa que no hay suficiente evidencia para rechazar la hipótesis nula de homocedasticidad. Es decir que no se detecta una variación significativa en la varianza de la serie a lo largo del tiempo, lo que sugiere que la varianza es aproximadamente constante. Finalmente, en lo referente a la revisión de la autocorrelación, los valores de autocorrelación (ACF) muestran que, aunque el rezago 1 tiene una autocorrelación positiva baja (0.072), los rezagos posteriores muestran valores cercanos a cero o incluso negativos, lo que implica que no hay un patrón de autocorrelación significativo más allá del primer rezago. Esto indica que la serie no presenta autocorrelación significativa en los rezagos más lejanos, por lo que se puede considerar que la autocorrelación es constante y no afecta sustancialmente a la serie. De este modo, los resultados anteriores indican que la serie tiene una ligera tendencia descendente, aunque casi insignificante. La varianza es constante a lo largo del tiempo y no se observan patrones de autocorrelación más allá del primer rezago. Por lo tanto, las tres propiedades (tendencia cero, varianza constante y autocorrelación constante) se cumplen de manera razonable para esta serie temporal.

Ahora bien, con base en los resultados obtenidos para la serie diferenciada, es posible observar que la pendiente de la regresión lineal en la serie diferenciada es de -0.0025, lo que

indica una tendencia negativa, pero es aún más cercana a cero que en la serie original. Esto significa que, tras la diferenciación, la tendencia ha disminuido aún más, y la serie se aproxima mucho más a tener una tendencia nula. Lo anterior permite concluir que la diferenciación ha ayudado a reducir la tendencia, acercándola más a cero. Ahora bien, con respecto a la varianza constante para la serie diferenciada, los resultados de la prueba de heterocedasticidad de Breusch-Pagan muestran un p-value de 0.9716, lo que indica una alta probabilidad de que la hipótesis nula de homocedasticidad sea cierta. En este caso, el p-value es aún más alto que en la serie original, lo que sugiere que la varianza es aún más constante en la serie diferenciada. Por lo tanto, es posible concluir que la varianza se ha estabilizado y es constante a lo largo de la serie. Finalmente, en lo que respecta a la autocorrelación de la serie diferenciada, los valores de autocorrelación (ACF) muestran una autocorrelación negativa significativa en el rezago 1 (-0.499), lo que indica un comportamiento inverso en la serie con respecto al rezago anterior. Sin embargo, en los rezagos posteriores, los valores de autocorrelación se acercan mucho más a cero, con pequeñas fluctuaciones tanto positivas como negativas, lo que implica que no hay un patrón claro de autocorrelación más allá del primer rezago. De este modo, la serie diferenciada muestra una disminución de la autocorrelación en los rezagos posteriores al primero, lo que sugiere una autocorrelación constante y sin dependencia significativa en el tiempo. A partir de lo anterior, es posible concluir que la serie diferenciada ha logrado reducir la tendencia, acercándola más a cero, lo que implica que la tendencia ya no es un factor significativo. La varianza es aún más constante después de la diferenciación, como lo indica el alto p-value en la prueba de Breusch-Pagan. Además, aunque hay una autocorrelación negativa en el primer rezago, los rezagos posteriores muestran poca o ninguna autocorrelación, lo que sugiere que la autocorrelación es constante después de la diferenciación. Así, en general, la diferenciación ha mejorado las tres

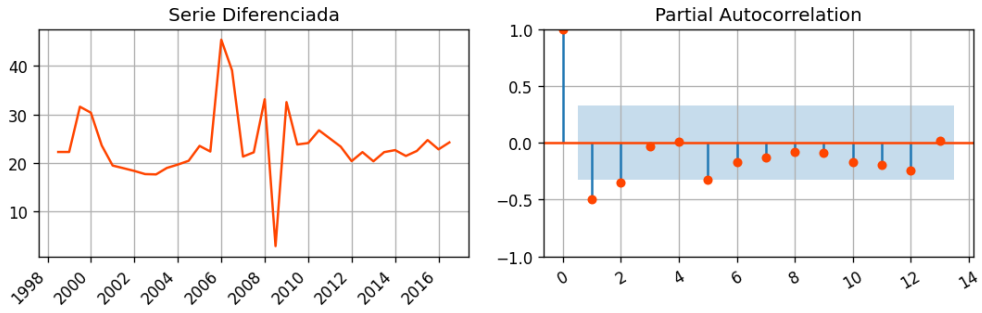
propiedades analizadas en la serie temporal, por lo que el valor de d puede ser 0 o 1, siendo mejor el comportamiento estacionario para d=1.

Obtención de los Parámetros p y q del Modelo

Una vez fueron identificados los posibles valores de d, a continuación se procedió con la estimación de los posibles parámetros p y q, mediante el uso respectivo de las gráficas de autocorrelación parcial y simple, de tal modo que en cada caso se realiza el análisis del comportamiento de los rezagos de la serie diferenciada. Cabe mencionar que esta estimación no siempre es definitiva y es necesario aplicar pruebas de ajuste y bondad para corroborar los modelos ARIMA que mejor se adaptan a las particularidades de la serie. Así, en la Figura 13 se presenta la gráfica de autocorrelación parcial generada para la serie diferenciada una vez.

Figura 13

Autocorrelación Parcial de la Serie Diferenciada



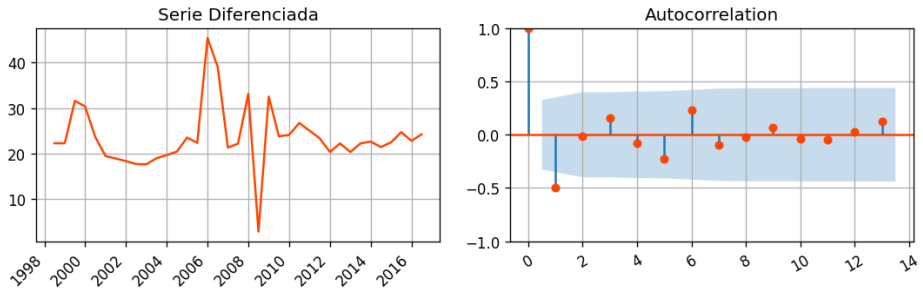
A partir de la gráfica de autocorrelación parcial (PACF) presentada en la Figura 13, es posible observar que el primer lag (lag 1) tiene una correlación significativa que sobrepasa claramente la banda de confianza. Así mismo, el segundo lag (lag 2) también sobrepasa la banda de confianza, aunque de manera más leve. Esto indica que el modelo autoregresivo (AR) podría ser de orden 1 o, en algunos casos, de orden 2. Sin embargo, debido a que el lag 1 muestra una mayor significancia y el lag 2 es menos claro, un modelo AR(1) parece ser una buena opción

inicial. Lo anterior implica que la dependencia en los valores anteriores de la serie de tiempo está mayormente influenciada por el valor inmediatamente anterior. Por tanto, p = 1 es el valor sugerido para este modelo autoregresivo, aunque también es posible experimentar las métricas de ajuste y bondad con p = 2.

Del mismo modo en lo que respecta al valor de q, se hizo uso de la gráfica de autocorrelación simple presentada en la Figura 14, la cual fue generada mediante el uso de la librería stastsmodels de Python.

Figura 14

Autocorrelación Simple de la Serie Diferenciada



Al observar la gráfica de autocorrelación (ACF) presentada en la Figura 14, es posible apreciar que el único rezago que sobrepasa significativamente la zona de confianza es el rezago 1. Esto indica que la autocorrelación más significativa se encuentra en el primer rezago y que, para los rezagos posteriores, los valores están dentro del intervalo de confianza, lo que sugiere que no hay una autocorrelación significativa más allá de este punto. Dado este patrón, el posible valor de q en un modelo ARIMA, que representa el número de rezagos de la parte MA (media móvil), sería 1. Esto se debe a que solo el primer rezago muestra una autocorrelación significativa, mientras que los rezagos posteriores no muestran un comportamiento autocorrelacionado considerable. Por lo tanto, el modelo podría beneficiarse al incluir un

componente de media móvil de orden 1 (MA(1)). Si bien el valor q del modelo ARIMA puede ser 1, de acuerdo con el análisis de la autocorrelación simple, también se tomará en consideración el valor de 0 basado en la experiencia. Esto se debe a que, en algunos casos, modelos más simples pueden ofrecer un ajuste adecuado y evitar el riesgo de sobreajuste, manteniendo la interpretabilidad y eficiencia del modelo.

Verificación de los Parámetros Obtenidos Usando los Criterios AIC y BIC

Como resumen de los resultados obtenidos en la sección anterior, se logró concluir que el posible valor de d puede ser 1. Así mismo para parámetro p se tendrán en cuenta los valores de 1 y 2, mientras que para el parámetro q se hará uso de los valores 0, 1. De acuerdo con lo anterior, se desarrolló el script presentado en la Figura 15, en donde se comparan las diferentes combinaciones de modelos ARIMA, con el fin tanto de de evaluar si los diferentes componentes de las ecuaciones son significativos (p-value <=0.05), como de comparar las métricas de ajuste y bondad AIC y BIC.

Figura 15

Comparación de los Posibles Modelos ARIMA en Python

```
from statsmodels.tsa.arima.model import ARIMA
lista = [(1,1,1), (1,1,0), (1,2,1), (1,2,0), (2,2,1), (2,2,0)]
for modo in lista:
  model_x=ARIMA(train, order=modo)
  model_x_fit=model_x.fit()
  #print(model_x_fit.summary())
  print("========================")
  print("Modo", modo)
  print("AIC",round(model_x_fit.aic,3))
  print("BIC",round(model_x_fit.bic,3))
  print("Log Likelihood", round(model_x_fit.llf))
  print("P_values",model_x_fit.pvalues)
```

Como resultado de la aplicación del script de la Figura 15, son presentadas en la Tabla 4 las significancias de las ecuaciones de los diferentes modelos (ARIMA(1,1,1), ARIMA(1,1,0), ARIMA(1,2,1), ARIMA(1,2,0),ARIMA(2,2,1), ARIMA(2,2,0)) y las métricas de ajuste y bondad (AIC y BIC) asociados a cada uno de estos. Es importante mencionar que a medida que

el modelo se hace más complejo es mayor la cantidad de componentes y también mayor el número de significancias correspondientes a estos.

Tabla 4

Comparación de Modelos ARIMA con los Parámetros Estimados

Modelo	Significancias y métricas de ajuste y bondad
ARIMA(1,1,1)	AIC 249.442 BIC 254.193 Log Likelihood -122 P_values componentes: ar.L1 0.462973 ma.L1 0.970167 sigma2 0.970157
ARIMA(1,1,0)	AIC 256.373 BIC 259.54 Log Likelihood -126 P_values componentes: ar.L1 2.691367e-05 sigma2 3.698170e-10
ARIMA(1,2,1)	AIC 256.702 BIC 261.368 Log Likelihood -125 P_values componentes: ar.L1 0.000041 ma.L1 0.983386

Modelo	Significancias y métricas de ajuste y bondad
	sigma2 0.983371
ARIMA(1,2,0)	AIC 278.572
	BIC 281.683
	Log Likelihood -137
	P_values componentes:
	ar.L1 7.685935e-16
	sigma2 1.453131e-09
ARIMA(2,2,1)	AIC 254.908
	BIC 261.13
	Log Likelihood -123
	P_values componentes:
	ar.L1 8.999163e-08
	ar.L2 4.570799e-02
	ma.L1 9.730873e-01
	sigma2 9.729974e-01
ARIMA(2,2,0)	AIC 266.001
	BIC 270.667
	Log Likelihood -130
	P_values componentes:
	ar.L1 9.842724e-31
	ar.L2 5.581950e-05

Modelo	Significancias y métricas de ajuste y bondad
	sigma2 1.030917e-06

Nota. Datos recopilados por los autores de la investigación.

A partir de los resultados presentados, es posible observar que los modelos con componentes estadísticamente significativos (p-value ≤ 0.05), cumpliendo con la hipótesis nula de que los coeficientes no son iguales a cero, son los modelos ARIMA(1,1,0), ARIMA(1,2,0), y ARIMA(2,2,0). En particular, el modelo ARIMA(1,1,0) destaca por tener p-values extremadamente bajos en sus componentes, lo que refuerza la conclusión de que estos coeficientes son significativamente distintos de cero. Además, este modelo presenta los menores valores de AIC (256.373) y BIC (259.54), lo que sugiere que es el más adecuado entre los evaluados, tanto por su simplicidad como por su capacidad predictiva. Las métricas AIC (Criterio de Información de Akaike) y BIC (Criterio de Información Bayesiano) son fundamentales en la evaluación de modelos, ya que penalizan la complejidad del modelo, favoreciendo aquellos que logran un equilibrio entre un buen ajuste a los datos y la parquedad en el número de parámetros. En este contexto, los valores más bajos de AIC y BIC del ARIMA(1,1,0) indican que este modelo no solo proporciona un buen ajuste, sino que lo hace de manera eficiente, evitando el sobreajuste al limitar el número de parámetros innecesarios, lo que fortalece su capacidad para generalizar y predecir fuera de la muestra.

De manera adicional, también se realizó la evaluación de los modelos ARIMA que tienen a d=0 como parámetro (ARIMA(1,0,0), ARIMA (1,0,1), ARIMA (2,0,0) y ARIMA

(2,0,1)), de tal modo que en la Tabla 5 son presentadas las significancias de los componentes de estas ecuaciones y las métricas de ajuste AIC y BIC.

Tabla 5

Comparación de Modelos ARIMA con el Parámetro d=0

Modelo	Significancias y métricas de ajuste y bondad
ARIMA(1,0,0)	AIC 251.656 BIC 256.489 Log Likelihood -123 P_values componentes: const 4.331669e-73 ar.L1 5.338241e-01 sigma2 8.785357e-12
ARIMA(1,0,1)	AIC 253.567 BIC 260.011 Log Likelihood -123 P_values componentes: const 5.024649e-59 ar.L1 8.837041e-01 ma.L1 9.065283e-01 sigma2 1.960628e-09
ARIMA(2,0,0)	AIC 253.503 BIC 259.947 Log Likelihood -123 P_values components:

Modelo	Significancias y métricas de ajuste y bondad
ARIMA(2,0,1)	const 1.874164e-62 ar.L1 5.792153e-01 ar.L2 8.296064e-01 sigma2 9.398107e-12 AIC 255.471 BIC 263.526 Log Likelihood -123 P_values componentes: const 1.751973e-57 ar.L1 9.418869e-01 ar.L2 8.734247e-01 ma.L1 9.597777e-01 sigma2 7.161252e-10

Nota. Datos recopilados por los autores de la investigación.

De acuerdo con los resultados obtenidos y presentados en la Tabla 5, es posible apreciar que ninguno de los modelos ARIMA evaluados y que tienen como parámetro d=0 tienen a todos sus componentes significativos, por lo cual se concluye que estos modelos no son adecuados para capturar de manera precisa la dinámica subyacente de la serie temporal. A pesar de que algunos componentes individuales presentan p-values bajos, lo que indica cierta relevancia estadística, el hecho de que otros componentes no sean significativos sugiere que los modelos con d=0 no logran un ajuste robusto. Esto refuerza la necesidad de trabajar con modelos donde la serie es diferenciada (es decir, d=1), como en el caso del ARIMA(1,1,0), el cual, en las evaluaciones presentadas de manera previa, demostró ser el modelo con el mejor ajuste y mayor

capacidad predictiva, tanto por la significancia de sus parámetros como por los menores valores de AIC y BIC obtenidos.

Implementación del Modelo ARIMA Mediante los Parámetros d, p y q

Una vez determinado que el modelo cuyos componentes son significativos y presenta mejores métricas de ajuste y bondad es el ARIMA(1,1,0) se procedió con la implementación y ajuste del modelo haciendo uso de las ventajas provistas por la librería statsmodels, tal como muestra en la Figura 16.

Figura 16

Script con la Implementación del Modelo ARIMA (1,1,0)

```
#Se pasan como paràmetro al modelo los valores p(1), d(1), q(0)
from statsmodels.tsa.arima.model import ARIMA
model_mejor=ARIMA(train, order=(1,1,0))
mejor_fit=model_mejor.fit()
print(mejor_fit.summary())
```

En el fragmendo de código presentado en la Figura 16, se implementa y ajusta un modelo ARIMA con parámetros (1,1,0), que se corresponde con el proceso autorregresivo integrado de orden 1 y sin media móvil. La primera línea importa la clase ARIMA del módulo statsmodels.tsa.arima.model, la cual permite construir modelos ARIMA. En la segunda línea, se crea un objeto del modelo denominado model_mejor, al cual se le pasa como parámetro el conjunto de entrenamiento train y el orden del modelo especificado como (1,1,0). Esto indica que se está considerando un componente autorregresivo (AR) de orden 1, una diferenciación (I) de primer orden para hacer estacionaria la serie, y sin componente de media móvil (MA). En la tercera línea, se ajusta el modelo a los datos con el método .fit(), obteniendo así los coeficientes estimados y los ajustes necesarios. Finalmente, en la cuarta línea, se imprime un resumen

completo del modelo ajustado utilizando .summary(), el cual permite mostrar detalles sobre los coeficientes, errores estándar, intervalos de confianza y valores de las métricas de bondad de ajuste, como AIC y BIC. Esta información se analizará en detalle en la siguiente actividad.

Evaluación de la Efectividad del Modelo ARIMA Usando el Conjunto de Pruebas y las Métricas de Error

Ahora bien, una vez realizada la implementación y ajuste del modelo ARIMA (1,1,0), en la Figura 17 se presenta el resumen de los resultados proporcionados por la librería statsmodels, lo cual incluye los detalles asociados a los coeficientes, errores estándar, intervalos de confianza y valores de las métricas de bondad de ajuste, como AIC y BIC.

Figura 17

Resumen de los Resultados Obtenidos para el Modelo ARIMA(1,1,0)

SARIMAX Results

Dep. Variable:	Tasa_Por	No. Observations:	37
Model:	ARIMA(1, 1, 0)	Log Likelihood	-126.186
Date:	Thu, 10 Oct 2024	AIC	256.373
Time:	03:48:02	BIC	259.540
Sample:	06-30-1998	HQIC	257.478
	- 06-30-2016		
Covariance Type:	opg		

	coef	std err	z	P>|z|	[0.025	0.975]
ar.L1	-0.4863	0.116	-4.198	0.000	-0.713	-0.259
sigma2	64.3913	10.276	6.266	0.000	44.251	84.532

Ljung-Box (L1) (Q):	1.28	Jarque-Bera (JB):	16.28
Prob(Q):	0.26	Prob(JB):	0.00
Heteroskedasticity (H):	0.16	Skew:	-0.48
Prob(H) (two-sided):	0.00	Kurtosis:	6.15

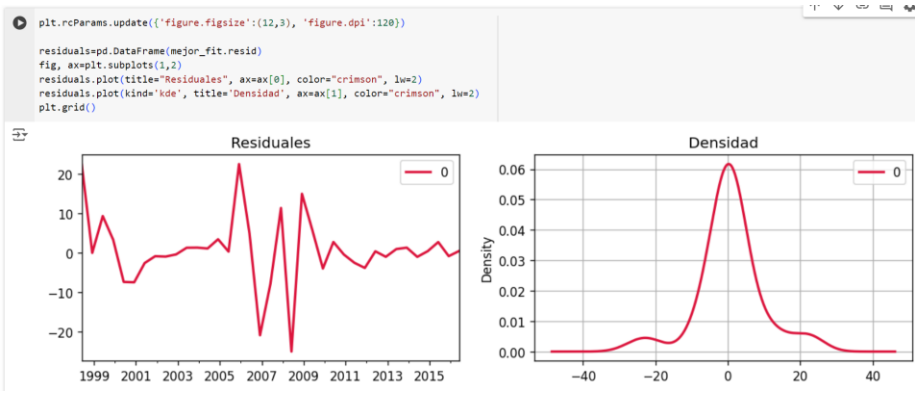
Los resultados presentados en la Figura 17 con respecto al modelo ARIMA(1,1,0) indican que el componente autorregresivo (AR) es estadísticamente significativo, con un coeficiente estimado de -0.4863 y un p-valor de 0.000, lo que refuerza la idea de que este parámetro tiene un impacto considerable en la predicción de la serie temporal. Además, el término sigma2, que representa la varianza del error, también es altamente significativo (p-valor

de 0.000), lo que sugiere que la volatilidad del error se modela adecuadamente. En términos de las métricas de ajuste, el modelo presenta un valor de AIC de 256.373, BIC de 259.540 y HQIC de 257.478, lo que indica que es un modelo relativamente parsimonioso en comparación con alternativas más complejas. Así mismo, el valor de Log Likelihood es -126.186, lo que, junto con los valores bajos de AIC y BIC, sugiere un buen ajuste del modelo sin sobreajustar los datos. Estas métricas de información penalizan la complejidad del modelo y, en este caso, apoyan que el ARIMA(1,1,0) es una opción adecuada, equilibrando simplicidad y capacidad predictiva.

Así mismo, a partir del modelo ARIMA con el mejor ajuste, se hizo el análisis de los residuales del modelo, obteniendo su función de densidad de probabilidad, tal como se muestra en la Figura 18. En este sentido, la distribución de los residuales muestra una forma aproximadamente simétrica, con una alta concentración alrededor de cero, lo que indica una aproximación a la distribución normal, aunque con colas más pronunciadas. Esto sugiere que, si bien los residuales están cerca de seguir una distribución normal, existen algunos valores extremos que no son perfectamente capturados por el modelo que presenta el mejor ajuste de los datos de deserción de la UNAD.

Figura 18

Residuales del Modelo ARIMA(1,1,0)



Evaluación de la Efectividad del Modelo ARIMA

De cara a la evaluación de la efectividad del modelo ARIMA(1,1,0) se evaluaron las métricas de MSE, MAE y RMSE tanto para el conjunto de entrenamiento y prueba, de tal modo que fueron definidas las funciones presentadas en la Figura 19 para la obtención de dichas métricas. Cabe mencionar que dado que el conjunto de pruebas tiene un total de 13 valores, antes de evaluar las métricas se realizó con el modelo la predicción de 13 valores posteriores a la ultima fecha considerada por el conjunto de entrenamiento (2016-06-30).

Figura 19

Métodos Implementados para la Evaluación del Modelo ARIMA

```python
import numpy as np
from sklearn import metrics

def mean_absolute_percentage_error(y_true, y_pred):
    y_true, y_pred=np.array(y_true), np.array(y_pred)
    return np.mean( np.abs((y_true-y_pred)/y_true)) * 100

def metricas(y_true, y_pred):
  print("Resultados: ")
  print(f"MSE es {metrics.mean_squared_error(y_true, y_pred)}")
  print(f"MAE es {metrics.mean_absolute_error(y_true, y_pred)}")
  print(f"RMSE es {np.sqrt(metrics.mean_squared_error(y_true, y_pred))}")
  print(f"MAPE es {mean_absolute_percentage_error(y_true, y_pred)}")
```

Ahora bien una vez usados los métodos definidos previamente, se obtiene como resultado los reportes en las métricas presentados en la Tabla 6, donde dichos resultados han sido discriminados para los conjuntos de entrenamiento y de prueba.

Tabla 6

Resultados de las Métricas de Error para el Modelo

Conjunto	Métricas de error
Entrenamiento	MSE es 76.09549038166149
37 instancias	MAE es 5.335038580986912
	RMSE es 8.723272916839269
Pruebas	MSE es 44.84141252985531
13 instancias	MAE es 5.108372458841453
	RMSE es 6.69637308771362

Nota. Datos recopilados por los autores de la investigación.

Los resultados mostrados en la Tabla 6 reflejan las métricas de error obtenidas al evaluar el modelo ARIMA(1,1,0) sobre el conjunto de entrenamiento y pruebas, específicamente para predecir la tasa porcentual de deserción en la UNAD. En el conjunto de entrenamiento, compuesto por 37 instancias, el error cuadrático medio (MSE) es de 76.09, el error absoluto medio (MAE) es de 5.33 y la raíz del error cuadrático medio (RMSE) es de 8.72. Estos valores indican que, durante la fase de entrenamiento, el modelo tiene un error relativamente bajo al capturar la tendencia de la serie, con una desviación promedio de alrededor de 5.33 puntos porcentuales entre las predicciones y los valores reales. Así mismo, en lo que respecta al conjunto de pruebas, que consta de 13 instancias, el modelo presenta un mejor rendimiento con un MSE de 44.84, un MAE de 5.10 y un RMSE de 6.69. El hecho de que las métricas de error en el conjunto de pruebas sean menores que en el conjunto de entrenamiento sugiere que el modelo

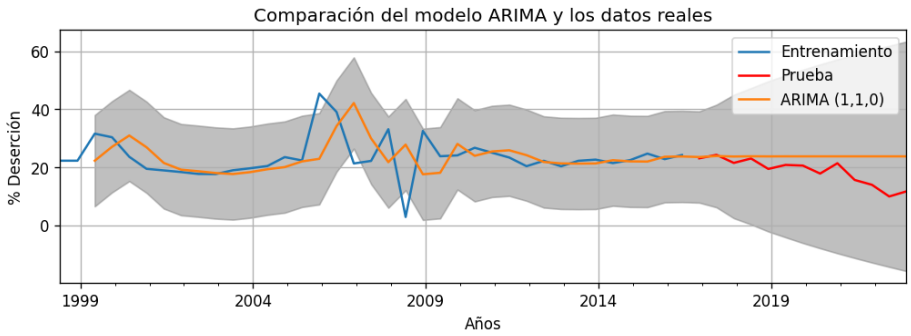
no está sobreajustado, lo que es un buen indicativo de su capacidad de generalización. La disminución del MSE y el RMSE refleja que el modelo es capaz de realizar predicciones más precisas en nuevos datos, manteniendo una desviación promedio cercana a los 5 puntos porcentuales en ambas fases. Este desempeño es aceptable dado el contexto de la tasa de deserción, donde errores en el rango de 5 puntos porcentuales pueden considerarse razonables para tomar decisiones predictivas. Así los resultados obtenidos, permiten considerar al modelo ARIMA considerado como consistente y con un ajuste adecuado para la caractertización y predicción de la deserción a semestres futuros.

Verificación y Comparación Gráfica del Modelo ARIMA con Respecto a los Datos Reales de Deserción

Una vez evaluadas las métricas de error para el conjunto de entrenamiento y de pruebas, se procedió con la generación de una gráfica comparativa del modelo ARIMA (1,1,0) con respecto a los conjuntos de entrenamiento y pruebas, haciendo uso de las ventajas gráficas provistas por la librería statsmodels de Python (ver Figura 20). Así, en color azul se muestra el conjunto de entrenamiento de los datos, en color rojo se presenta el conjunto de entrenamiento de la serie, mientras que en color naranja se muestra el modelo ARIMA (1,1,0).

Figura 20

Comparación del Modelo ARIMA con los Conjuntos



En la gráfica del la Figura 20, es posible observar como el modelo ARIMA(1,1,0) (línea naranja) sigue de manera efectiva la serie temporal de las tasas de deserción porcentual tanto en el conjunto de entrenamiento (línea azul) como en el de prueba (línea roja). A lo largo del tiempo, el modelo logra capturar las fluctuaciones de la serie sin mostrar signos de sobreajuste, manteniendo una trayectoria suave y razonable. En las fases más recientes de la serie, correspondientes al conjunto de prueba, el modelo converge a un valor constante ligeramente superior al 20%, lo que refleja su capacidad para capturar la tendencia general de la deserción sin reaccionar de manera excesiva a las variaciones menores. Este comportamiento sugiere que el modelo puede generalizar adecuadamente, proporcionando predicciones estables y consistentes.

Ahora bien, a partir del modelo ajustado se realizaron predicciones más allá del conjunto de pruebas, de tal modo que se obtuvieron un total de 8 predicciones, lo cual comprende desde los semestres de 2025 hasta los semestres de 2028 (ver Tabla 7).

Tabla 7

Predicciones Obtenidas por el Modelo ARIMA(1,1,0)

Índice	Predicción
2025-06-30	23.802115
2025-12-31	23.802113
2026-06-30	23.802114
2026-12-31	23.802114
2027-06-30	23.802114
2027-12-31	23.802114
2028-06-30	23.802114
2028-12-31	23.802114

Nota. Datos recopilados por los autores de la investigación.

Las predicciones obtenidas en la Tabla 7 para los años 2025 hasta 2028 utilizando el modelo ARIMA(1,1,0) muestran una tendencia constante con valores que oscilan alrededor de 23.80%. Al analizar estas predicciones, se puede calcular que el valor promedio es aproximadamente 23.8021, mientras que la desviación estándar es prácticamente nula, lo que indica que el modelo no predice fluctuaciones significativas en la tasa de deserción para este periodo. Esta estabilidad en las predicciones sugiere que la tasa de deserción se mantendrá constante, sin variaciones importantes a lo largo de los próximos años. A partir de lo anterior, dado que el valor medio predicho es del 23.80%, esto implica que, por cada 100 estudiantes que ingresan a la universidad, alrededor de 24 podrían desertar antes de finalizar sus estudios, según las proyecciones del modelo. Este valor proporciona una perspectiva importante para las políticas educativas, ya que predice una tasa de deserción considerablemente alta que deberá abordarse para mejorar la retención de estudiantes en la institución.

Definición de los Requisitos Funcionales y no Funcionales del Software

Para la especificación de los requisitos funcionales del prototipo software desarrollado, se realizó una adaptación del formato propuesto por el estándar IEEE 830, de tal modo que a continuación se presentan la descripción de los requisitos: “Cargar dataset”, “Separar conjuntos”, “Identificar parámetro d”, “Identificar parámetros p y q”, “Evaluar modelos”, “Ajustar y evaluar modelo” y “Realizar predicciones”. De este modo, en la Tabla 8 se presenta la descripción del requisito “Cargar dataset”, el cual hace referencia a la funcionalidad del prototipo software que permite al usuario final cargar dataset de deserción de la UNAD y visualizar en primera instancia la serie de tiempo cargada.

Tabla 8

Requisito Funcional "Cargar Dataset"

Requisito Funcional	REQ1
Nombre del requisito	Cargar dataset
Tipo	Requisito funcional
Descripción	El prototipo software permite al usuario cargar a partir de un archivo excel el dataset con la fecha y la tasa de deserción porcentual asociada a dicha fecha, de tal modo que el sistema una vez cargado el archivo permite visualizar un gráfico de líneas con la deserción semestre a semestre.
Prioridad	Media

Nota. Datos recopilados por los autores de la investigación.

En ese mismo sentido, en la Tabla 9 es presentada la descripción del requisto funcional "Separar conjuntos", en el cual se permite al usuario indicar y visualizar el porcentaje de separación del dataset de deserción de la UNAD en los conjuntos de entrenamiento y de prueba. El conjunto de entrenamiento permite al modelo ARIMA realizar el ajuste, mientras que el conjunto de prueba permite verificar la capacidad de ajuste con un conjunto con el cual no fue entrenado el modelo.

Tabla 9

Requisito Funcional "Separar Conjuntos"

Requisito Funcional	REQ2
Nombre del requisito	Separar conjuntos
Tipo	Requisito funcional
Descripción	El prototipo software permite al usuario especificar el porcentaje en el cual será separado el dataset de derserción de la UNAD en conjuntos de entrenamiento y prueba. Así mismo una vez escogido dicho porcentaje, el prototipo software permitirá la visualización en una gráfica de línea de los dos conjuntos.
Prioridad	Media

Nota. Datos recopilados por los autores de la investigación.

Así mismo, en la Tabla 10, se presenta la descripción del requisito funcional denominado "Identificar parámetro d", en el cual se permite al usuario aplicar la prueba de estacionariedad de Dickey Fuller sobre las diferentes versiones de la serie (serie original y sus diferenciaciones), lo cual es útil para identificar los valores factibles que puede tomar el parámetro d del modelo ARIMA.

Tabla 10

Requisito Funcional "Identificar Parámetro p"

Requisito Funcional	REQ3
Nombre del requisito	Identificar parámetros p
Tipo	Requisito funcional
Descripción	El prototipo software permite al usuario especificar la serie a considerar (original o diferenciaciones) para a partir de esta aplicar y obtener los resultados de la prueba de Dickey Fuller, los cuales permiten identificar si la serie analizada es estacionaria o no.
Prioridad	Alta

Nota. Datos recopilados por los autores de la investigación.

De otra parte, en la Tabla 11, es presentada la descripción del requisito funcional "Identificar parámetros p y q", el cual incluye la funcionalidad que permite al usuario visualizar las gráficas de autocorrelación parcial y autocorrelación simple de la serie original o de las diferenciaciones de la serie, a partir de las cuales es posible estimar de manera respectiva los valores factibles de p y q.

Tabla 11

Requisito Funcional "Identificar Parámetros p y q"

Requisito Funcional	REQ4
Nombre del requisito	Identificar parámetros p y q
Tipo	Requisito funcional
Descripción	El software permite al usuario especificar la serie a considerar (original o diferenciaciones) para generar las gráficas de autocorrelación parcial y simple de la serie escogida, las cuales permiten identificar los posibles valores de los parámetros p y q del modelo ARIMA.
Prioridad	Alta

Nota. Datos recopilados por los autores de la investigación.

Así mismo, en la Tabla 12, se presenta la descripción del requisito funcional denominado "Evaluar modelos", el cual incluye la funcionalidad que permite al usuario especificar los valores máximos que pueden tomar los parámetros p, q y d, de tal modo que el prototipo software permitirá implementar los modelos ARIMA que combinan esos parámetros obteniendo los p-values de cada componente de la ecuación de estos modelos, así como las métricas de bondad y ajuste: AIC y BIC. A partir de los p-values, el usuario puede identificar el modelo ARIMA más

significativo según las características de los datos. Además, mediante las métricas AIC y BIC, es posible determinar el modelo con el mejor ajuste.

Tabla 12

Requisito Funcional "Evaluar Modelos"

Requisito Funcional	REQ5
Nombre del requisito	Evaluar modelos
Tipo	Requisito funcional
Descripción	El software permite al usuario especificar los parámetros p,q y d máximos que pueden tomar los modelos, de modo que una vez especificados dichos parámetros el sistema ajusta y obtiene los p-values de las ecuaciones de dichos modelos, así como las métricas de ajuste y bondad AIC y BIC. Estas métricas permiten determinar el modelo de series de tiempo que presenta un mejor ajuste a los datos.
Prioridad	Alta

Nota. Datos recopilados por los autores de la investigación.

En este mismo sentido, en la Tabla 13, se presenta la descripción del requisito funcional denominado "Ajustar y evaluar modelo", en el cual se incluye la funcionalidad que permite al usuario especificar los parámetros (p,q y d) de alguno de los modelos ARIMA con mejor ajuste

para que el sistema obtenga las métricas de error (MAE, MSE, RMSE) para el conjunto de entrenamiento y prueba. Lo anterior, permite identificar la capacidad de generalización del modelo ARIMA con un conjunto con el que no fue entrenado.

Tabla 13

Requisito Funcional "Ajustar y Evaluar Modelo"

Requisito Funcional	REQ6
Nombre del requisito	Ajustar y evaluar modelo
Tipo	Requisito funcional
Descripción	El prototipo software permite al usuario especificar los parámetros p,q y d de un modelo ARIMA determinado, de tal manera que el sistema obtiene las métricas de error (MAE, MSE y RMSE) del modelo escogido en cuanto al conjunto de entrenamiento y de prueba. Así es posible identificar si el modelo es consistente y tiene buena capacidad de generalización o predicción con valores que desconoce.
Prioridad	Alta

Nota. Datos recopilados por los autores de la investigación.

Finalmente, en la Tabla 14 se presenta la descripción del requisito funcional denominado "Realizar predicciones", el cual incluye la funcionalidad en la cual luego de que el usuario

especifica los parámetros p, q y de de un modelo ARIMA determinado, el sistema obtiene las predicciones de tasa de deserción de dicho modelo para un número de semestres especificado también por el usuario. Estas predicciones son de utilidad para la toma de decisiones por parte de las autoridades educativas de la UNAD con respecto a la mitigación de la deserción estudiantil.

Tabla 14

Requisito Funcional "Realizar Predicciones"

Requisito Funcional	REQ7
Nombre del requisito	Realizar predicciones
Tipo	Requisito funcional
Descripción	El prototipo software permite al usuario especificar los parámetros p,q y d de un modelo ARIMA determinado, así como el número de semestres a predecir por parte del modelo, de tal manera que el sistema obtiene las predicciones para el número de semestres indicado por el usuario.
Prioridad	Media

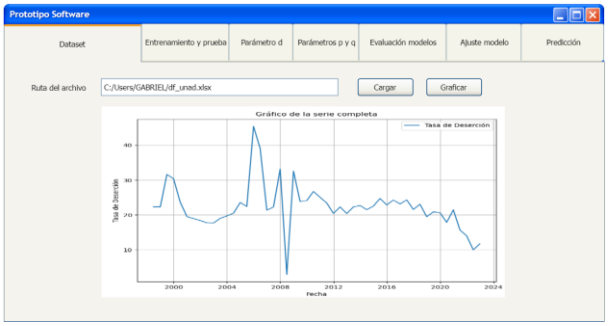
Nota. Datos recopilados por los autores de la investigación.

Especificación de la Funcionalidad del Software Mediante Interfaces de Borrador

Partiendo de los requisitos funcionales definidos previamente, en esta sección se presentan interfaces de borrador que permitan la especificación de los requisitos y puedan servir de guía para la implementación del prototipo software a implementar. Así, para la definición de estas interfaces de alto nivel se hizo uso de las ventajas de prototipado de la herramienta de código abierto Pencil. De este modo en la Figura 21 se aprecia la interfaz de alto nivel del prototipo software, el cual fue diseñado para contar con un total de 7 pestañas a saber: “Dataset”, “Entrenamiento y prueba”, “Parámetro d”, “Parámetros p y q”, “Evaluación modelos”, “Ajuste modelo”, “Predicción”. En la primera pestaña “Dataset” el prototipo software permite cargar el dataset con la tasa de deserción de la UNAD y visualizar mediante un diagrama de líneas la fluctuación de la deserción a lo largo de los semestres al presionar el botón “Graficar”.

Figura 21

Interfaz Principal de Borrador de la Herramienta

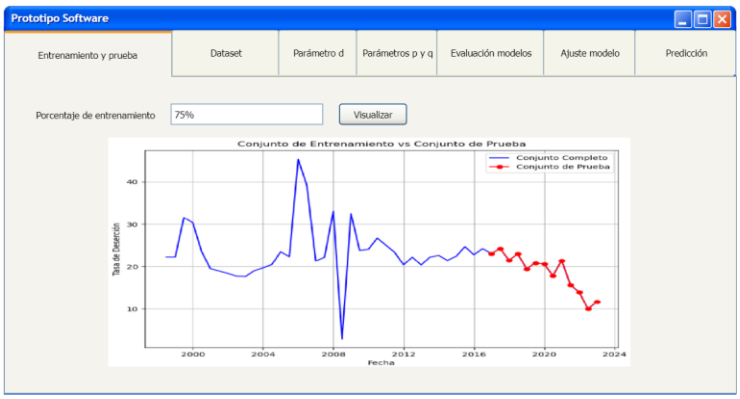


De otra parte, en la Figura 22 es posible apareciar la interfaz de alto nivel correspondiente a la pestaña “Entrenamiento y prueba” en la cual el usuario puede indicar el porcentaje en el que se realizará la separación de los conjuntos de entrenamiento y prueba, de tal modo que al dar clic en el botón “Visaulizar” el prototipo software despliega la gráfica de la serie de tiempo con la

división de los conjuntos (color azul para el conjunto de entrenamiento y color rojo para el conjunto de prueba).

Figura 22

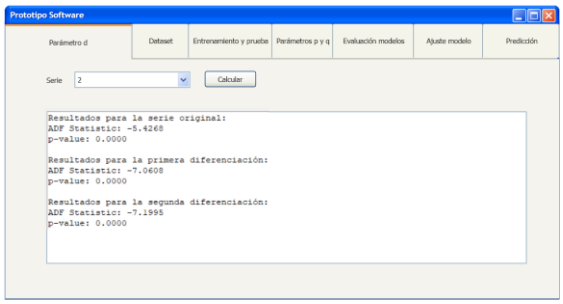
Interfaz de Borrador Pestaña "Entrenamiento y Prueba"



Así mismo, en la Figura 23 es posible apreciar la interfaz de alto nivel correspondiente a la pestaña "Parámetro d", en la cual una vez el usuario indica el orden de la serie de tiempo y presiona el botón "Calcular", el prototipo software se encarga de aplicar la prueba de Dickey Fuller sobre el orden de la serie especificado, de tal manera que es obtenido el p-value y el estadístico ADF para la o las series especificadas.

Figura 23

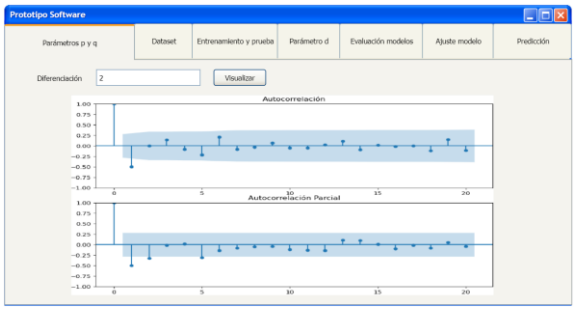
Interfaz de Borrador Pestaña "Parámetro d"



De otra parte, en la Figura 24 es posible apreciar la interfaz de alto nivel correspondiente a la pestaña “Parámetros p y q”, en la cual una vez el usuario especifica el orden de la serie y se presiona el botón “Visualizar”, el prototipo software se encarga de generar la autocorrelación parcial y la autocorrelación simple de la serie especificada.

Figura 24

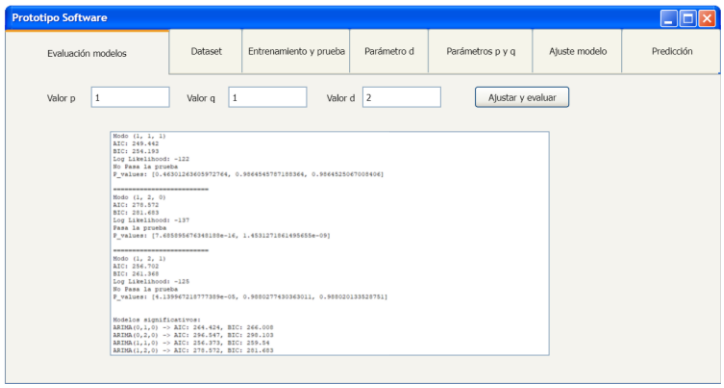
Interfaz de Borrador Pestaña “Parámetros p y q”



En este mismo sentido en la Figura 25 se presenta la interfaz de alto nivel correspondiente a la pestaña “Evaluación modelos”, en la cual una vez especificados los valores máximos de los parámetros de ARIMA (p, q y d) y presionado el botón “Ajustar y evaluar”, el prototipo software obtiene los p-value de cada componente del modelo y las métricas de ajuste y bondad (AIC y BIC) asociados a cada modelo ARIMA.

Figura 25

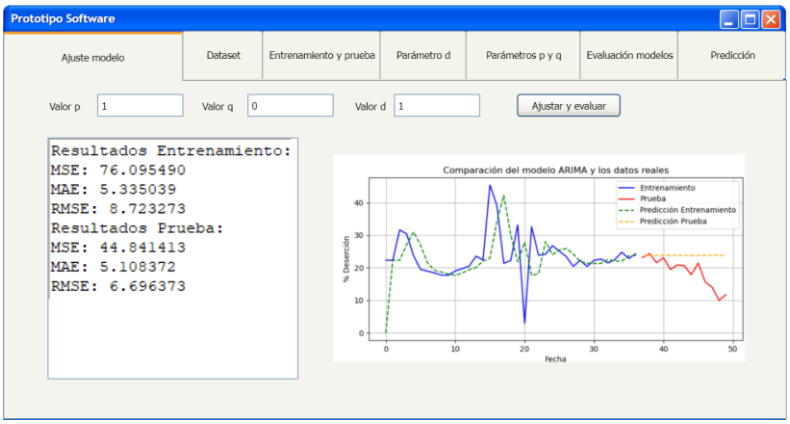
Interfaz de Borrador Pestaña "Evaluación Modelos"



En este mismo sentido, en la Figura 26 se presenta la interfaz de alto nivel correspondiente a la pestaña "Ajuste modelo", en la cual una vez especificados los valores de los parámetros de un modelo ARIMA (p, q y d) particular y presionado el botón "Ajustar y evaluar", el prototipo software obtiene las métricas de error MSE, RMSE y MAE tanto para el conjunto de entrenamiento, como para el conjunto de prueba. Estas métricas son de gran relevancia puesto que permiten identificar que tan bueno es el modelo generalizando a partir de datos que no conoce. Así mismo, la magnitud en ambos conjuntos permite evaluar la consistencia del modelo en los conjuntos de entrenamiento y prueba.

Figura 26

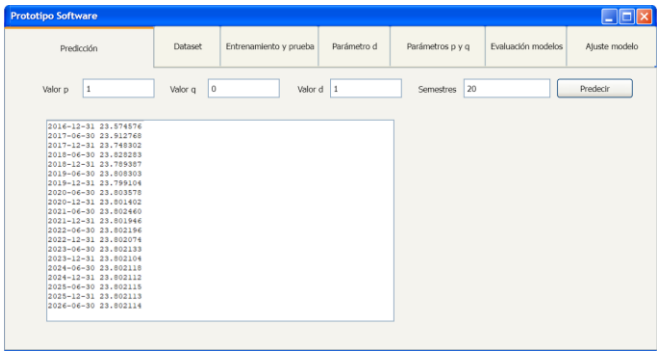
Interfaz de Borrador Pestaña "Ajuste Modelo"



Finalmente, en la Figura 27 se presenta la interfaz de alto nivel correspondiente a la pestaña "Predicción", en la cual una vez especificados tantos los valores de los parámetros de un modelo ARIMA (p, q y d) particular, como el número de semestres a predecir, el prototipo software obtiene las predicciones del modelo ARIMA para los semestres especificados, una vez se ha presionado el botón "Predecir".

Figura 27

Interfaz de Borrador Pestaña "Predicción"



Implementación de las Interfaces del Prototipo de Software

Teniendo en cuenta las interfaces de alto nivel definidas en la sección anterior, en esta sección se presenta la implementación del prototipo software haciendo uso de librerías del código abierto tales como: statsmodels, matplotlib, pandas, numpy, tkinter entre otras. La librería statsmodels fue utilizada para la implementación de los modelos ARIMA, la librería matplotlib fue empleada para la generación de las gráficas asociadas a los modelos ARIMA, la librería pandas fue utilizada para cargar el archivo excel con las tasas porcentuales de deserción y para la creación del dataframe a partir del dataset, la librería numpy fue usada para la realización de operaciones vectoriales relacionadas con el dataset y finalmente la librería tkinter fue empleada para la generación de la interfaz gráfica del prototipo software.

Cabe mencionar que el prototipo software implementado está conformado por un total de siete pestañas a saber: “Dataset”, “Entrenamiento y Prueba”, “Parámetro d”, “Parámetros p y q”, “Evaluación Modelos”, “Ajuste Modelo” y “Predicción”, tal como se presenta en la Figura X. Así, en la pestaña “Dataset”, el usuario puede cargar el archivo excel con las tasas porcentuales de deserción de la UNAD haciendo uso del botón “Cargar”, de tal modo que la ruta del archivo es presentada en la caja de texto de esta pestaña. Así mismo, una vez el usuario carga el archivo, puede presionar el botón “Graficar”, el cual permite la generación y el despliegue de una gráfica de línea con los datos completos de deserción presentes en el archivo excel (ver Figura 28). La generación de esta gráfica es realizada a partir de las ventajas de la librería matplotlib, la cual permite generar diferentes tipos de gráficas y ser integrada fácilmente como panel gráfico dentro de la interfaz.

Figura 28

Interfaz del Prototipo Software Implementado

De otra parte, en la pestaña “Entrenamiento y Prueba”, el usuario puede configurar el porcentaje de entrenamiento en la caja de texto disponible en esta pestaña, de tal modo que al presionar el botón “Visualizar” se genera y despliega una gráfica que permite diferenciar el conjunto de entrenamiento (color azul) y el conjunto de prueba (color rojo), según el porcentaje escogido (ver Figura 29). Cabe mencionar que la segmentación del porcentaje de entrenamiento y prueba es realizado a partir de las ventajas provistas por la librería pandas, mientras que la generación de la gráfica se realiza a través de la librería matplotlib. Así mismo se aprecia que para el dataset de deserción de la UNAD, el conjunto de entrenamiento va hasta el primer semestre de 2016. Lo anterior es clave de claro a identificar los rangos de fechas que corresponden a cada conjunto de cara a la obtención de las predicciones generadas por la serie de tiempo.

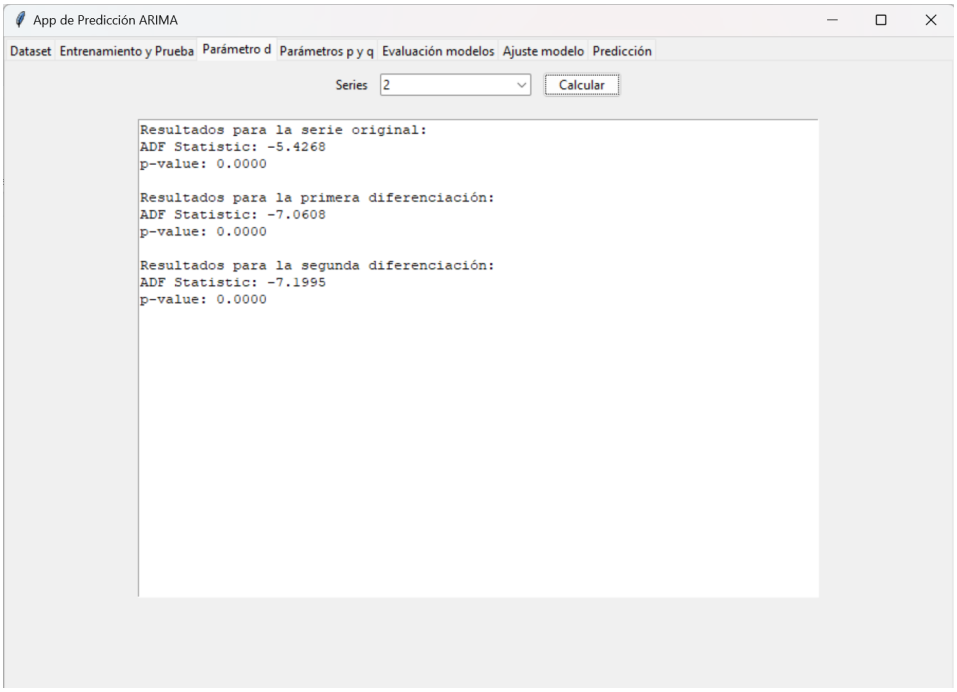
Figura 29

Interfaz de la Pestaña "Entrenamiento y Prueba"

De otra parte, en la pestaña "Parámetro d", el prototipo software permite la aplicación de la prueba de Dickey Fuller sobre la serie original o sus diferenciaciones, de tal modo que el usuario puede escoger en un combobox si la prueba se aplicará solo en la serie original (opción 0), sobre la serie original y la primera diferenciación (opción 1) o sobre la serie original, la primera diferenciación y la segunda diferenciación (opción 2). Una vez escogida la opción a partir del combobox y presionado el botón "Calcular", los resultados de las pruebas dependiendo de la opción escogida son presentados en el área de texto (ver Figura 30). Se puede apreciar a partir de la Figura 30 como para el dataset de la UNAD, tanto la serie original, la primera diferenciación y la segunda diferenciación paran la prueba de Dickey Fuller, lo que quiere decir que en los 3 casos se alcanza la estacionariedad.

Figura 30

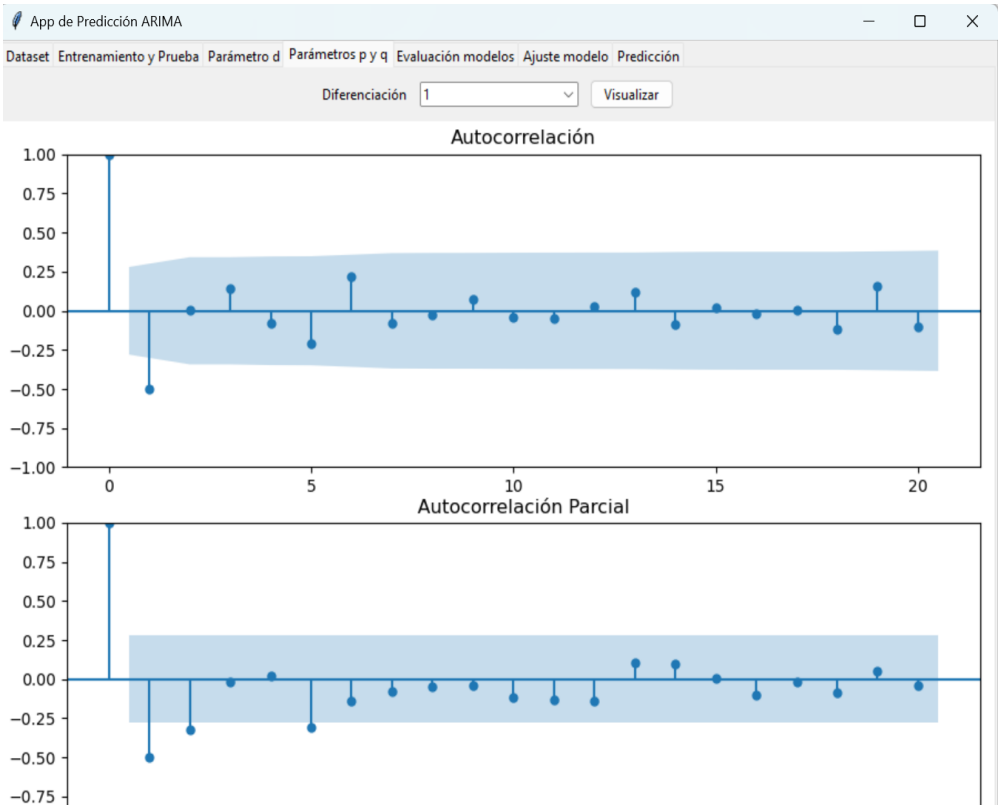
Interfaz de la Pestaña "Parámetro d"



Por otro lado, en la pestaña "Parámetros p y q", el usuario puede escoger a partir de un combobox, el nivel de diferenciación de la serie sobre el que se obtendrá la autocorrelación parcial y la autocorrelación simple (ver Figura 31). Así al escoger el nivel de diferenciación y presionar el botón "Visualizar", mediante la librería statsmodels y matplotlib se generan las gráficas de la autocorreolacón parcial y la autocorrelación simple, las cuales son usadas respectivamente para determinar de manera empírica los valores de p y q. Se puede apreciar que para la primera diferenciación de la serie tanto en la autocorrelación, como en la autocorrelación parcial, el rezago significativo es el primero.

Figura 31

Interfaz de la Pestaña "Parámetros p y q"

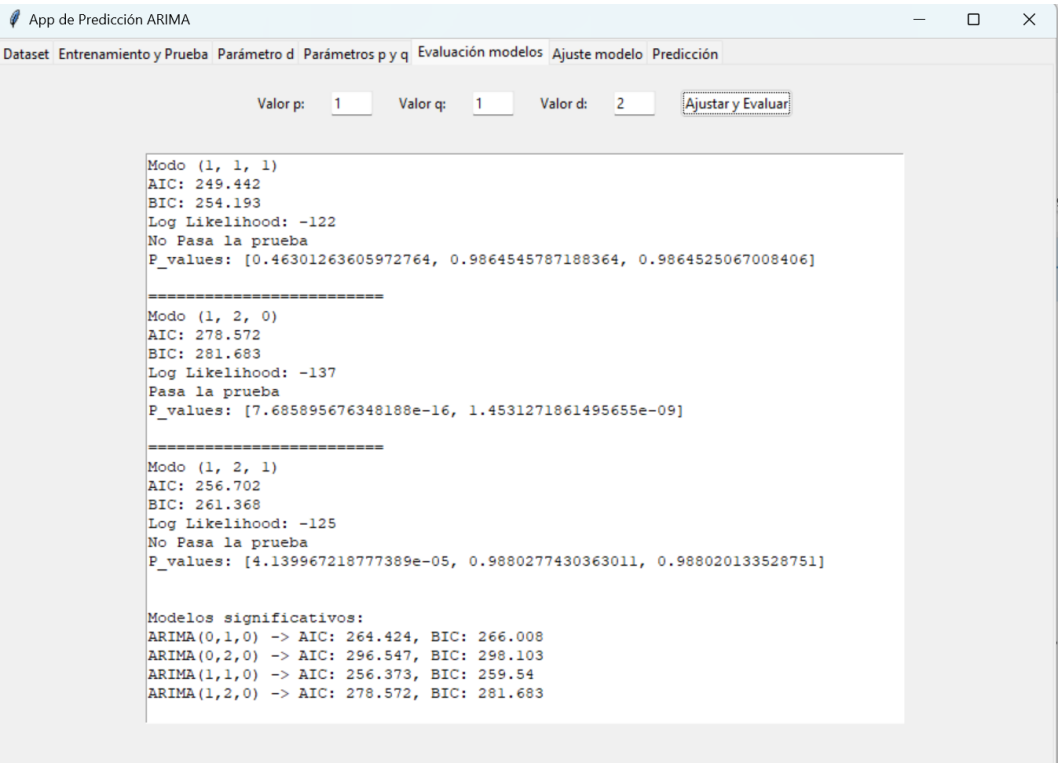


Continuando con la descripción de la interfaz, en la pestaña "Evaluación modelos" el usuario puede ingresar los posibles valores máximos de los parámetros p, q y d en los campos de texto respectivos, de acuerdo a los análisis empíricos de las pestañas anteriores, de tal modo que al presionar el botón "Ajustar y Evaluar", el prototipo software se encarga de implementar y ajustar diferentes combinaciones de modelos ARIMA conformados por las posibles combinaciones de los rangos de parámetros ingresados, obteniendo los p-values de sus componentes y las métricas AIC y BIC (ver Figura 32). Así, tal como se observa en la Figura X, para el caso del dataset con la tasa porcentual de deserción de la UNAD, la herramienta

determina que los mejores modelos ARIMA son los que tienen modo (0,1,0), (0,2,0), (1,1,0), (1,2,0), siendo el modelo ARIMA(1,1,0) el que obtiene las mejores métricas de bondad y ajuste.

Figura 32

Interfaz de la Pestaña "Evaluación Modelos"



Así mismo, en la pestaña "Ajuste modelo" el usuario puede incluir los parámetros p,q y d de alguno de los modelos que pasan las pruebas de significancia en la pestaña anterior a nivel de sus componentes y métricas AIC y BIC, de tal forma que una vez incluidos dichos parámetros y presionado el botón "Ajustar y Evaluar" es posible apreciar en la caja de texto de la pestaña los resultados de las métricas de error (MSE, MAE y RMSE) tanto para el conjunto de entrenamiento, como para el conjunto de prueba del modelo ARIMA configurado. De este modo, en la Figura 33, se mustra el ajuste que presenta el modelo ARIMA (1,1,0) el cual presenta las mejores métricas de ajuste y bondad para el dataset de la deserción de la UNAD.

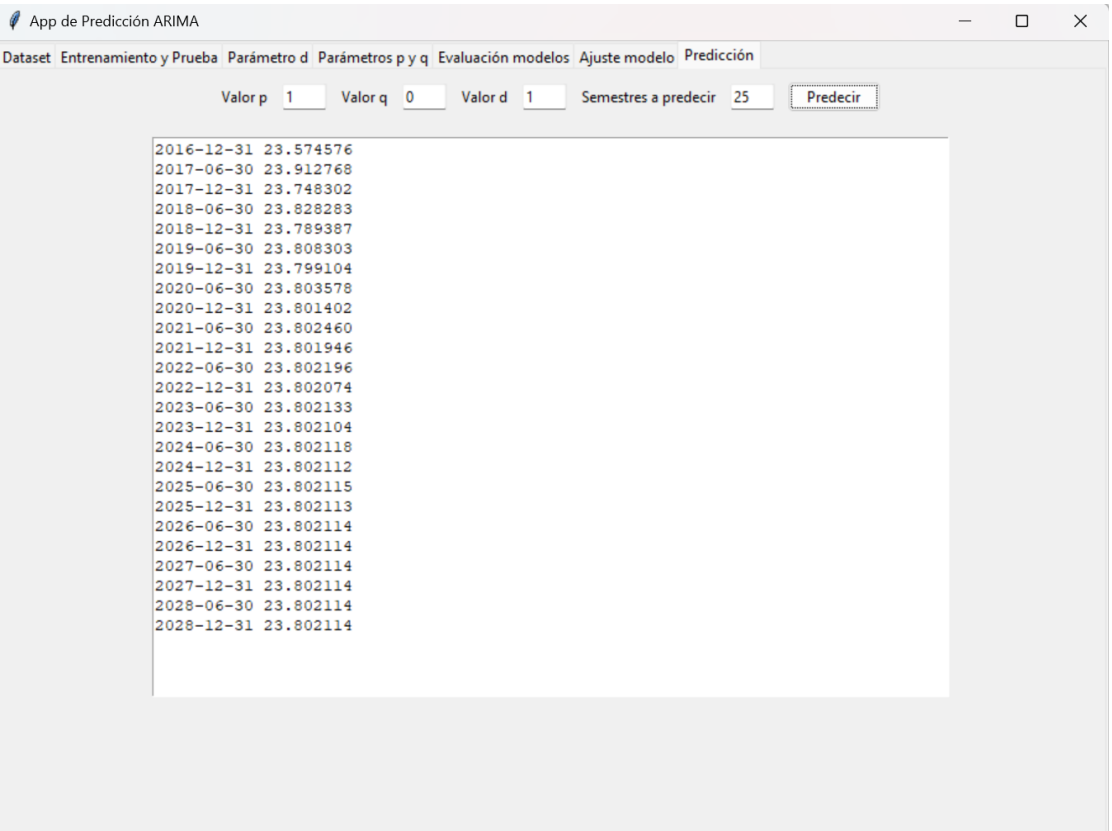
Figura 33

Interfaz de la Pestaña "Ajuste Modelo"

Finalmente, en la pestaña "Predicción", el usuario puede incluir los parámetros del modelo con mejor ajuste y el número de semestres a predecir luego de la última fecha considerada por el conjunto de entrenamiento, de tal modo que una vez incluidos estos datos y presionado el botón "Predecir", el prototipo software se encarga de ajustar el modelo con los parámetros incluidos y realizar la predicción para el número de semestres indicados (ver Figura 34). Así, para el caso del dataset de deserción de la UNAD, se aprecia que para los semestre futuros el modelo ARIMA (1,1,0) predice que la deserción de encontrará alrededor del 23%:

Figura 34

Interfaz de la Pestaña "Predicción"



Verificación del Cumplimiento de los Requisitos

Con el fin de verificar el cumplimiento de los requisitos inicialmente planteados, se realizó una evaluación de las salidas obtenidas por el prototipo sofware a partir de las entradas proporcionadas al sistema en cada uno de los requisitos. De este modo, en la Tabla 15 se presenta la descripción del desarrollo de las pruebas de verificación del cumplimiento de cada uno de los requisitos especificados previamente: "Cargar dataset", "Separar conjuntos", "Identificar parámetro d", "Identificar parámetros p y q", "Evaluar modelos", "Ajustar y evaluar modelo" y "Realizar predicciones".

Tabla 15

Verificación de los Requisitos Funcionales del Sistema

Requisito Funcional	Variable de entrada	Variable de salida	Descripción
Cargar dataset	Dataset con los datos de deserción históricos semestre a semestre para la UNAD.	El sistema carga el archivo excel con el dataset, muestra la ruta del archivo y despliega de manera correcta una gráfica de líneas con la deserción histórica de la UNAD semestre a semestre.	El requisito “Cargar dataset” se cumple de manera satisfactoria dentro del prototipo software.
Separar conjuntos	Porcentaje de división del dataset en conjuntos de entrenamiento y prueba.	El prototipo software permite dividir y presentar adecuadamente de manera gráfica los conjuntos de entrenamiento y prueba a partir del dataset original.	El requisito “Separar conuntos” se cumple de manera satisfactoria dentro del prototipo software.
Identificar parámetro d	Se ingresa el orden de la serie a la que se le aplicará la prueba de Dickey Fuller (serie original, primera	Una vez se ha indicado el orden de la serie a la cual se le verificará la estacionariedad, el prototipo software	El requisito “Identificar parámetro d” se cumple de manera satisfactoria dentro del prototipo software.

Requisito Funcional	Variable de entrada	Variable de salida	Descripción
	diferenciación, segunda diferenciación).	permite aplicar de manera adecuada la prueba de Dickey Fuller tanto a la serie original, como a la primera y segunda diferenciación.	
Identificar parámetro p y q	Se ingresa el orden de la serie (original, primera diferenciación, segunda diferenciación) de la cual el prototipo software obtendrá la autocorrelación parcial y la autocorrelación simple.	Una vez se ha indicado el orden de la serie, el prototipo software se encarga de calcular y desplegar correctamente en la interfaz el gráfico de la autocorrelación parcial y la autocorrelación simple, cada una con sus bandas de confianza.	El requisito “identificar parámetros p y q” se cumple de manera satisfactoria dentro del prototipo software.
Evaluar modelos	Se ingresan los valores máximos de los parámetros p,q y d a partir de los cuales son ajustados diferentes modelos ARIMA para determinar si son o no significativos.	Una vez indicados los valores máximos de los parámetros p,q y d, el prototipo software ajusta e imprime de manera correcta los p-values de los diferentes modelos	El requisito “Evaluar modelos” se cumple de

Requisito Funcional	Variable de entrada	Variable de salida	Descripción
		ARIMA, así como las métricas AIC y BIC.	manera satisfactoria dentro del prototipo software.
Ajustar y evaluar modelo	Se ingresan los valores de los parámetros p,q y d del modelo específico ARIMA a ajustar y evaluar mediante métricas de error.	Una vez ingresados los parámetros del modelo ARIMA a evaluar el prototipo software muestra de manera correcta las métricas de error (MSE, RMSE y MAE) para los conjuntos de entrenamiento y prueba. Así mismo, el prototipo software despliega de manera correcta una gráfica que compara el modelo ARIMA con respecto a los conjuntos de entrenamiento y prueba.	El requisito “Ajustar y evaluar modelo” se cumple de manera satisfactoria dentro del prototipo software.
Realizar predicciones	Son ingresados los parámetros p, q y d del modelo ARIMA a evaluar, así como el número de semestres a	Una vez son ingresados los parámetros (p, q y d) del modelo ARIMA a emplear y el número de semestres a predecir, el	

Requisito Funcional	Variable de entrada	Variable de salida	Descripción
	predecir mediante el modelo ARIMA.	modelo ARIMA se encarga de predecir de manera correcta la tasa de deserción porcentual de la UNAD para los semestres indicados.	El requisito “Realizar predicciones” se cumple de manera satisfactoria dentro del prototipo software.

Nota. Datos recopilados por los autores de la investigación.

Discusión

Teniendo en cuenta que la mayoría de los trabajos encontrados en el estado del arte se han centrado en el uso de técnicas de machine learning para identificar factores sociales, económicos y demográficos que inciden en la deserción académica, con el objetivo de clasificar a los estudiantes como posibles desertores o no desertores, este trabajo introduce un enfoque innovador basado en series de tiempo tipo ARIMA. Este enfoque no solo permite la caracterización de la tasa porcentual de deserción, sino que también facilita la predicción de su evolución en el tiempo. De este modo, el modelo y el prototipo propuestos en este trabajo ofrecen una perspectiva a largo plazo, brindando a las autoridades académicas de la UNAD una herramienta robusta para anticipar cambios en la tasa de deserción y, en consecuencia, diseñar y ajustar estrategias más efectivas y proactivas para mitigar este fenómeno. Al proporcionar una visión predictiva más allá del análisis de factores inmediatos, este enfoque refuerza la toma de decisiones informada y el desarrollo de políticas académicas sostenibles.

A la hora de trabajar con series de tiempo, es fundamental, según lo propuesto por la metodología CRISP-DM, realizar un proceso exhaustivo de limpieza de datos. Esto incluye la identificación y tratamiento de datos faltantes mediante técnicas de imputación, como la interpolación o el uso de la media. En el caso del dataset de la tasa de deserción porcentual de la UNAD, se aplicó la imputación para los dos primeros periodos con valores cero, permitiendo que el modelo tenga un mejor ajuste. Eliminar o corregir valores nulos o anómalos evita que el modelo sea influenciado negativamente, lo que mejora la precisión de las predicciones y garantiza resultados más confiables. Este proceso de limpieza no solo asegura la integridad de los datos, sino que también optimiza el rendimiento del modelo al eliminar posibles sesgos o distorsiones que podrían afectar su capacidad predictiva. De este modo, se promueve la creación

de modelos más robustos y efectivos, especialmente en entornos donde la calidad de los datos es crucial para la toma de decisiones.

Una tarea clave en los modelos predictivos es la separación de los datos en conjuntos de entrenamiento y prueba para evaluar el rendimiento del modelo con datos no vistos. Sin embargo, en el caso de las series de tiempo, este proceso no es aleatorio como en el aprendizaje automático tradicional, sino secuencial, ya que se asume la dependencia de los valores pasados de la serie. Separar adecuadamente los datos es fundamental para garantizar la validez del modelo, ya que permite simular su comportamiento en situaciones reales y evitar problemas como el sobreajuste. De esta manera, se puede evaluar de forma más precisa la capacidad del modelo para generalizar y hacer predicciones confiables en el futuro.

Uno de los parámetros clave en los modelos ARIMA es el parámetro d, que define el número de diferenciaciones necesarias para hacer estacionaria una serie temporal. La prueba de Dickey-Fuller es útil para evaluar la hipótesis de no estacionariedad, pero es fundamental complementarla verificando condiciones adicionales, como tendencia cero, varianza constante y autocorrelación parcial constante. En este sentido, el cumplimiento tanto de la prueba de Dickey-Fuller como de estas condiciones garantiza la adecuación de los datos para los modelos ARIMA. En el presente trabajo, se determinó que la serie alcanzaba un mejor comportamiento estacionario tras la primera diferenciación, aunque la prueba de Dickey-Fuller indicaba estacionariedad tanto en la serie original como en la diferenciada.

Derivado de lo mencionado anteriormente, es importante resaltar que el hecho de que tanto la serie original como sus diferenciaciones superen la prueba de estacionariedad de Dickey-Fuller refuerza la hipótesis de que los modelos de series de tiempo, como ARIMA, son altamente pertinentes para caracterizar la deserción porcentual. Esto sugiere que un modelo de series de

tiempo puede captar mejor la dinámica temporal subyacente en comparación con un modelo de aprendizaje supervisado basado en regresión lineal, que asume independencia entre observaciones. La prueba de Dickey-Fuller confirma que la serie tiene las características necesarias para ser modelada de manera efectiva con ARIMA, lo que permite una predicción más precisa y confiable a largo plazo, ajustándose mejor a la naturaleza temporal de los datos.

En cuanto a los parámetros p y q del modelo ARIMA, en este trabajo se utilizó el método empírico basado en la revisión de la autocorrelación parcial (PACF) y la autocorrelación simple (ACF) para identificar los rezagos significativos. En el caso de la tasa de deserción porcentual de la UNAD, se observó que el primer rezago en ambas funciones era significativo, sugiriendo que los valores más adecuados para p y q son 1. No obstante, fue crucial validar que los modelos ARIMA con esta combinación de parámetros, junto con el parámetro d previamente identificado, fueran significativos mediante la revisión de sus p-values. Esta verificación asegura la solidez estadística del modelo y su capacidad predictiva.

Al comparar los diferentes modelos ARIMA con los valores de los parámetros p, q y d estimados empíricamente, se determinó que los modelos significativos en sus componentes son: ARIMA(0,1,0), ARIMA(0,2,0), ARIMA(1,1,0) y ARIMA(1,2,0), todos con p-values menores a 0.05. Sin embargo, para seleccionar el mejor modelo entre los significativos, las métricas de ajuste y bondad como el AIC (Criterio de Información de Akaike) y el BIC (Criterio de Información Bayesiano) juegan un papel fundamental. Estas métricas penalizan la complejidad del modelo y favorecen aquellos que logran un equilibrio entre ajuste y simplicidad. En este caso, el modelo ARIMA(1,1,0) mostró los valores más bajos de AIC y BIC, lo que indica que es el mejor ajuste, proporcionando una predicción más precisa y eficiente sin sobreajustar los datos.

El uso de AIC y BIC asegura que el modelo no solo se ajuste bien a los datos de entrenamiento, sino que también tenga un mejor desempeño en datos futuros.

Al ser el modelo ARIMA(1,1,0) el que presentó el mejor ajuste para el conjunto de entrenamiento con los datos correspondientes a la tasa porcentual de deserción de la UNAD, se evaluó su rendimiento mediante el uso de métricas de error, obteniendo que para el conjunto de entrenamiento el MSE fue de 76.095490, el MAE de 5.335039 y el RMSE de 8.723273. En el conjunto de prueba, las métricas reflejan un mejor desempeño, con un MSE de 44.841413, un MAE de 5.108372 y un RMSE de 6.696373. Estos resultados indican que el modelo tiene un buen ajuste y mejora su capacidad predictiva en el conjunto de prueba, mostrando una disminución en el error cuadrático medio (MSE) y el error cuadrático medio de la raíz (RMSE), lo que sugiere que es capaz de capturar correctamente la estructura de los datos, reduciendo el margen de error en las predicciones para datos no vistos. Aunque las diferencias entre los errores en ambos conjuntos no son muy significativas en el MAE, la mejora en las métricas de MSE y RMSE refuerza la validez del modelo para predecir la tasa de deserción con mayor precisión en el conjunto de prueba.

Las predicciones realizadas a partir del modelo ARIMA (1,1,0) permitieron determinar que, para los semestres venideros desde 2025 en adelante, la tasa de deserción porcentual de la UNAD se mantendrá alrededor del 23%, lo que indica que se estima que por cada 100 estudiantes que ingresan a la universidad, aproximadamente 23 no podrán finalizar sus estudios. Además, el análisis de la desviación estándar en estas predicciones muestra una baja variabilidad, lo que refuerza la estabilidad de las proyecciones realizadas. Esta consistencia en las cifras brinda un panorama confiable que puede servir de base para la formulación de políticas educativas universitarias dirigidas a mitigar la deserción. El conocimiento preciso de la tasa y su

estabilidad permitirá a las autoridades académicas anticipar los recursos y acciones necesarios para mejorar la retención estudiantil en los próximos años.

A nivel académico, este trabajo presenta una contribución significativa al utilizar tecnologías y librerías de código abierto en todas las fases del proceso de ajuste del modelo y desarrollo del prototipo de software. La librería *pandas* fue fundamental para la carga y manipulación de los datos, permitiendo realizar operaciones clave como indexación y eliminación de valores nulos sobre los *dataframes*. Asimismo, *statsmodels* facilitó la aplicación de pruebas de estacionariedad, el ajuste del modelo ARIMA y la evaluación de su desempeño. Por otro lado, *matplotlib* permitió la generación de visualizaciones de las series de tiempo, mientras que *numpy* fue utilizada para manejar operaciones con arreglos y la partición de los datos en conjuntos de entrenamiento y prueba. Finalmente, *tkinter* fue empleada para desarrollar la interfaz gráfica, facilitando la interacción del usuario y la gestión de eventos en los controles visuales del software. El uso de estas herramientas refuerza la replicabilidad y accesibilidad del trabajo, promoviendo el desarrollo de soluciones abiertas y colaborativas en el ámbito académico.

A partir del modelo ARIMA propuesto, que caracteriza y predice la tasa porcentual de deserción en la UNAD, este trabajo presenta como contribución un prototipo de software con una interfaz gráfica intuitiva. Esta herramienta permite cargar datasets, identificar los parámetros *p*, *q* y *d* del modelo, evaluar el ajuste y las métricas de error de diferentes modelos, y generar predicciones para semestres futuros. Aunque la herramienta fue evaluada con el dataset de la tasa de deserción de la UNAD, su flexibilidad permite adaptarla a diferentes datasets de series de tiempo en variados dominios de aplicación, como la salud, la economía, la educación y la industria. Esto la convierte en una herramienta versátil con un gran potencial para apoyar la toma

de decisiones basada en datos. En el contexto educativo, podría ser utilizada para predecir tendencias relacionadas con la deserción, el rendimiento estudiantil u otros indicadores clave, permitiendo a las instituciones anticipar desafíos y formular estrategias proactivas. De igual manera, su aplicación en otros sectores puede optimizar procesos de planificación y gestión, haciéndola valiosa en múltiples escenarios.

En el desarrollo del prototipo de software utilizando la librería *tkinter* de Python, la definición clara de los requisitos funcionales y, en particular, la especificación de interfaces de alto nivel, fue crucial para garantizar un proceso de desarrollo eficiente. Estas interfaces actuaron como una guía fundamental durante la implementación, asegurando que cada componente del software estuviera alineado con los objetivos funcionales establecidos. Definir mockups o interfaces desde el inicio permitió reducir los errores y tiempos de ajuste, mejorando la eficiencia del desarrollo. En este contexto, la herramienta *Pencil* fue particularmente valiosa, ya que permite crear prototipos de interfaces de forma rápida y sencilla, proporcionando componentes gráficos predefinidos que hacen posible la elaboración de borradores visuales antes de la implementación. De este modo, este enfoque de trabajo permite anticipar problemas de diseño y funcionalidad, optimizando el flujo de trabajo y asegurando que el producto final cumpla con las expectativas de los usuarios de manera más efectiva.

Finalmente, como trabajo futuro derivado de la presente investigación, se plantea en primera instancia extrapolar el modelo a diferentes programas académicos de la UNAD, de tal modo que sea posible establecer una comparativa con el modelo general propuesto. Así mismo, se plantea la posibilidad de adaptar el sistema propuesto al contexto de la web, lo anterior con el fin de propiciar su difusión y facilidad de acceso y ejecución por parte de las autoridades educativas de la Universidad, de cara al seguimiento continuo de la deserción estudiantil.

Conclusiones

Dentro del presente proyecto, se caracterizaron en primera instancia las variables obtenidas a partir de la plataforma de seguimiento de la deserción SPADIES del Ministerio de Educación Nacional, de tal modo que se escogió como variable relevante para ser utilizada en la serie de tiempo la tasa de deserción porcentual, la cual corresponde al porcentaje de estudiantes que desertan semestre a semestre dentro de las instituciones educativas. De este modo, esta plataforma aporta variables claves que pueden ser usadas para caracterizar y predecir la deserción en instituciones educativas haciendo uso de modelos de series de tiempo como ARIMA. Del mismo modo, otra variable relevante que puede ser obtenida a partir de la plataforma es el porcentaje de retención, el cual corresponde a la proporción de estudiantes que permanecen inscritos y continúan sus estudios semestre a semestre dentro de una misma institución educativa. Este indicador es fundamental para evaluar la efectividad de las estrategias institucionales y garantizar la continuidad educativa. Además, puede ser usado de igual modo para ser caracterizado y para la obtención de pronósticos, lo que permite a las instituciones diseñar estrategias proactivas y personalizadas que mejoren la retención estudiantil y reduzcan la deserción a futuro, fortaleciendo así la planificación académica y administrativa.

Como paso previo al ajuste del modelo de series de tiempo ARIMA, se imputaron los datos correspondientes a los 2 primeros periodos del dataset, haciendo uso de la media de la serie de tiempo. Una vez ajustada la serie, se procedió con la determinación del parámetro d de la serie, a través de la prueba de estacionariedad de Dickey-Fuller obteniendo que la serie original y la primera diferenciación son estacionarias. Así mismo se usaron las gráficas de la autocorrelación parcial y la autocorrelación simple para la estimación de los posibles valores de los parámetros p y q, los cuales posteriormente se verifican a partir de los p-values de los

componentes de la ecuación. La determinación de los posibles valores de los parámetros es crucial para posteriormente evaluar y ajustar los modelos que satisfacen dichos parámetros con el fin de identificar los modelos significativos y con mejor ajuste. Como aspecto clave, es importante señalar que la determinación de los posibles parámetros fue realizada gracias a las ventajas provistas por la librería statsmodels.

Una vez identificados los posibles parámetros del modelo ARIMA, se realizó el ajuste y la comparación de diferentes modelos, teniendo en cuenta para lo anterior que los componentes de la ecuación de los diferentes modelos pasaran la prueba de significancia y tuvieron los valores más bajos posibles en las métricas AIC y BIC. De este modo, se determinó que el modelo que tiene mejor ajuste a la serie de tiempo es el ARIMA(1,1,0), el cual demostró tener métricas de error adecuadas y consistencia entre el conjunto de entrenamiento y prueba. Este modelo obtuvo como predicción un porcentaje de deserción alrededor de 23% para años posteriores a 2025, lo que indica que por cada 100 estudiantes que ingresan a la UNAD, 23 desertan.

Finalmente, este trabajo presentó como aporte relevante, el diseño y desarrollo de un prototipo software en el lenguaje Python, el cual a partir de una interfaz gráfica implementada con la librería Tkinter, permite cargar el dataset, determinar los posibles valores de los parámetros p,q y d, determinar el mejor modelo, evaluar dicho modelo mediante métricas de error y realizar predicciones. Esta herramienta pretende beneficiar a la UNAD en cuanto a la caracterización y la deserción para años venideros, de cara a la toma de decisiones estratégicas enfocadas a mejorar la retención estudiantil. Finalmente, se realizó un proceso de verificación de los requisitos funcionales del software, obteniendo como resultado que la herramienta cumple adecuadamente las funcionalidades definidas.

Recomendaciones

A la hora de trabajar con series de tiempo, es crucial revisar los datos faltantes o nulos, para luego aplicar la estrategia adecuada según las características y el contexto de la serie. En este sentido, las técnicas de interpolación resultan especialmente útiles, ya que permiten imputar los datos faltantes siguiendo la tendencia de la serie. Esto asegura que la continuidad y la estructura temporal de los datos se mantengan, minimizando el impacto de los valores faltantes y mejorando la precisión del modelo al preservar la coherencia de la serie.

Si bien los métodos empíricos para la determinación de los parámetros *p* y *q*, basados en el análisis gráfico de los rezagos en las gráficas de autocorrelación parcial y autocorrelación, son fundamentales, no constituyen una condición suficiente. Estos métodos brindan una estimación inicial que debe ser verificada al ajustar los modelos, comprobando la significancia estadística de los componentes de la ecuación. Es crucial realizar un análisis estadístico riguroso para determinar si los parámetros y el modelo en su conjunto son significativos, garantizando así la validez de las predicciones. De este modo, el análisis tanto visual como estadístico es esencial para asegurar la confiabilidad y precisión del modelo final.

En este mismo sentido, respecto al parámetro *d* en los modelos ARIMA, aunque la prueba de estacionariedad de Dickey-Fuller es útil para determinar si se rechaza o acepta la hipótesis nula de no estacionariedad, es fundamental complementarla con la verificación de las tres condiciones clave de estacionariedad: tendencia cero, varianza constante y autocorrelación constante. Estas condiciones aseguran que la serie cumpla con los requisitos necesarios para el modelado adecuado, y su verificación permite evaluar si es más apropiado trabajar con la serie original o su diferenciación. Este enfoque asegura que el modelo ARIMA se ajuste correctamente y produzca predicciones más precisas.

A la hora de trabajar con modelos de series de tiempo basados en ARIMA utilizando la librería *statsmodels*, es fundamental desarrollar un script que permita iterar entre diferentes modelos ARIMA ajustados con los posibles valores de los parámetros *p, q* y *d*, los cuales se estiman a partir de la prueba de estacionariedad de Dickey-Fuller y el análisis de las gráficas de autocorrelación y autocorrelación parcial. Este script debe imprimir los *p-values* de los componentes de las ecuaciones, así como las métricas de ajuste y bondad, como el AIC y BIC. Al hacerlo, se obtiene una visión clara de la significancia estadística de cada modelo y su capacidad de ajuste, lo que facilita la selección del modelo más adecuado según las características específicas de los datos. Este enfoque asegura que se elija un modelo robusto y bien ajustado, optimizando la capacidad predictiva del ARIMA.

A la hora de implementar un prototipo de software basado en un modelo predictivo, es esencial definir los requisitos funcionales del software a partir de las fases desarrolladas para el modelo. Con estos requisitos como base, es crucial especificar las interfaces de alto nivel o mockups, utilizando herramientas de prototipado como *Pencil*. Estas interfaces proporcionan una representación visual clara de la estructura y funcionalidad del software antes de su implementación, lo que contribuye con la mejora de la eficiencia del proceso al identificar posibles problemas de diseño temprano. El uso de interfaces de alto nivel permite también una mejor planificación del flujo de trabajo, optimiza los tiempos de desarrollo y asegura que el prototipo se alinee correctamente con los requisitos funcionales, garantizando un desarrollo más ágil y orientado a los objetivos del proyecto.

Referencias Bibliográficas

Alban, M., & Mauricio, D. (2019). Neural networks to predict dropout at the universities. *International Journal of Machine Learning and Computing*, *9*(2), 149–153. https://doi.org/10.18178/ijmlc.2019.9.2.779

Alcauter, I., Martinez-Villaseñor, L., & Ponce, H. (2023). Explaining Factors of Student Attrition at Higher Education. *Computación y Sistemas*, *27*(4). https://doi.org/10.13053/cys-27-4-4776

Aldeman, M., & Szekely, M. (2017). An Overview of School Dropout in Central America: Unresolved Issues and New Challenges for Education Progress. *European Journal of Educational Research*, *6*(3), 235–259. https://doi.org/10.12973/eu-jer.6.3.235

Alvarado-Uribe, J., Mejía-Almada, P., Masetto Herrera, A. L., Molontay, R., Hilliger, I., Hegde, V., Montemayor Gallegos, J. E., Ramírez Díaz, R. A., & Ceballos, H. G. (2022). Student Dataset from Tecnologico de Monterrey in Mexico to Predict Dropout in Higher Education. *Data*, *7*(9), 119. https://doi.org/10.3390/data7090119

Anderson, D., Sweeney, D., & Williams, T. (2008). *Estadística para administración y economía*. Thomson/Southwestern. https://fad.unsa.edu.pe/bancayseguros/wp-content/uploads/sites/4/2019/03/LIBRO-13-Estadistica-para-administracion-y-economia.pdf

Arciniegas Paspuel, O. G., Castro Morales, L. G., & Arias Collaguazo, W. M. (2021). Análisis y predicción de la recaudación tributaria en el Ecuador ante la COVID-19, aplicando el modelo ARIMA. *Dilemas Contemporáneos: Educación, Política y Valores*, *8*(3), 1–18. https://doi.org/10.46377/dilemas.v8i.2708

Arthana, I. K. R. (2024). Optimizing Dropout Prediction in University Using Oversampling

Techniques for Imbalanced Datasets. *International Journal of Information and Education Technology*, *14*(8), 1052–1060. https://doi.org/10.18178/ijiet.2024.14.8.2133

Ayala-Yaguara, H. Y., Valenzuela-Sabogal, G. M., & Espinosa-García, A. (2019). Obtención de un modelo de minería de datos aplicado a la deserción universitaria del programa de Ingeniería de Sistemas de la Universidad de Cundinamarca. *Revista ONTARE*, *7*, 134–150. https://dialnet.unirioja.es/servlet/articulo?codigo=8705565

Ayala Castrejon, R. F., & Bucio Pacheco, C. (2020). Modelo ARIMA aplicado al tipo de cambio peso-dólar en el periodo 2016-2017 mediante ventanas temporales deslizantes. *Revista Mexicana de Economía y Finanzas*, *15*(3), 331–354. https://doi.org/10.21919/remef.v15i3.466

Ayele, W. Y. (2020). Adapting CRISP-DM for Idea Mining. *International Journal of Advanced Computer Science and Applications*, *11*(6). https://doi.org/10.14569/IJACSA.2020.0110603

Canchano, O., Coronado, L., Sanchez, P. S., & others. (2019). Redes neuronales para pronóstico de series de tiempo hidrológicas del Caribe colombiano. *Investigación y Desarrollo En TIC*, *10*(2), 18–31.

Cardoso Melo, E., & Sumika Hojo de Souza, F. (2023). Improving the prediction of school dropout with the support of the semi-supervised learning approach. *ISys - Brazilian Journal of Information Systems*, *16*(1). https://doi.org/10.5753/isys.2023.2852

Caselli Gismondi, H. E., & Urrelo Huiman, L. V. (2021). Características para un modelo de predicción de la deserción académica universitaria. Caso Universidad Nacional de Santa. *LLamkasun: Revista de Investigación Científica y Tecnológica*, *2*(4), 2–22. https://dialnet.unirioja.es/servlet/articulo?codigo=8152475

Challa, M. L., Malepati, V., & Kolusu, S. N. R. (2020). S&P BSE Sensex and S&P BSE IT return forecasting using ARIMA. *Financial Innovation*, *6*(1), 47. https://doi.org/10.1186/s40854-020-00201-5

Chinkes, E. (2018). Pronósticos y data mining para la toma de decisiones. Pronóstico sobre la deserción de alumnos de una Facultad. *Cuadernos Del Cimbage*, *1*(20), 107–132. https://ojs.econ.uba.ar/index.php/CIMBAGE/article/view/1184/1793

Cruz, E., González, M., & Rangel, J. C. (2022). Técnicas de machine learning aplicadas a la evaluación del rendimiento y a la predicción de la deserción de estudiantes universitarios, una revisión. *Prisma Tecnológico*, *13*(1), 77–87. https://doi.org/10.33412/pri.v13.1.3039

Darling-Hammond, L., & Cook-Harvey, C. (2018). *Educating the whole child: Improving school climate to support student success*. https://learningpolicyinstitute.org/product/educating-whole-child-report

Dávila-Morán, R.-C., Aguero-Corzo, E. del C., Portillo-Ríos, H., & Quimbita-Chiluisa, O.-R. (2022). Deserción universitaria de los estudiantes de una universidad peruana. *Revista Universidad y Sociedad*, *14*(2), 421–427. http://scielo.sld.cu/pdf/rus/v14n2/2218-3620-rus-14-02-421.pdf

Dávila, G., Haro, J., González-Eras, A., Vivanco, O. R., & Coronel, D. G. (2023). Student Dropout Prediction in High Education, Using Machine Learning and Deep Learning Models: Case of Ecuadorian University. *2023 International Conference on Computational Science and Computational Intelligence (CSCI)*, 1677–1684. https://doi.org/10.1109/CSCI62032.2023.00277

De Witte, K., & Rogge, N. (2013). Dropout from Secondary Education: all's well that begins well. *European Journal of Education*, *48*(1), 131–149. https://doi.org/10.1111/ejed.12001

Díaz-Mujica, A., Pérez-Villalobos, M. V., Bernardo-Gutiérrez, A., Fernández-Castañón, A., & González-Pienda, J. (2019). Affective and cognitive variables involved in structural prediction of university dropout. *Psicothema*, *31*(4), 429–436. https://doi.org/10.7334/psicothema2019.124

Dorado Bastidas, C. A., Córdoba Campos, E. Y., & Chanchí Golondrino, G. E. (2023). Dashboard apoyado en inteligencia de negocios para toma de decisiones en el sector salud. *Revista Gestión y Desarrollo Libre*, *8*(16), 1–13. https://doi.org/https://doi.org/10.18041/2539-3669/gestionlibre.16.2023.10226

Espinosa Zúñiga, J. J. (2020). Aplicación de metodología CRISP-DM para segmentación geográfica de una base de datos pública. *Ingeniería Investigación y Tecnología*, *21*(1), 1–13. https://doi.org/10.22201/fi.25940732e.2020.21n1.008

Espinoza, Ó., González Fiegehen, L. E., & Loyola Campos, J. (2021). Factores determinantes de la deserción escolar y expectativas de estudiantes que asisten a escuelas alternativas. *Educación y Educadores*, *24*(1), 113–134. https://doi.org/10.5294/educ.2021.24.1.6

Espinoza, O., González, L. E., McGinn, N., & Castillo, D. (2021). Engaging dropouts with differentiated practices: some evidence from Chile. *Research Papers in Education*, *36*(6), 637–656. https://doi.org/10.1080/02671522.2020.1736615

Foley, K., Gallipoli, G., & Green, D. A. (2014). Ability, Parental Valuation of Education, and the High School Dropout Decision. *Journal of Human Resources*, *49*(4), 906–944. https://doi.org/10.1353/jhr.2014.0027

Gonzalez-Nucamendi, A., Noguez, J., Neri, L., Robledo-Rella, V., & García-Castelán, R. M. G. (2023). Predictive analytics study to determine undergraduate students at risk of dropout. *Frontiers in Education*, *8*. https://doi.org/10.3389/feduc.2023.1244686

González León, K. (2018). *Estudio multiparamétrico de series de tiempo biológicas*. Benemérita Universidad Autónoma de Puebla.

Gutierrez-Pachas, D. A., Garcia-Zanabria, G., Cuadros-Vargas, E., Camara-Chavez, G., & Gomez-Nieto, E. (2023). Supporting Decision-Making Process on Higher Education Dropout by Analyzing Academic, Socioeconomic, and Equity Factors through Machine Learning and Survival Analysis Methods in the Latin American Context. *Education Sciences*, *13*(2), 154. https://doi.org/10.3390/educsci13020154

Gutierrez-Villareal, J. O., Fonseca-Gómez, L. R., & Pineda-Ríos, W. (2021). Estimación de las principales causas de la deserción universitaria mediante el uso de técnicas de machine learning. *Aglala*, *12*(2), 293–311. https://revistas.curn.edu.co/index.php/aglala/article/view/2105

Ingrum, A. (2007). High School Dropout Determinants: The Effect of Poverty and Learning Disabilities. *The Park Place Economist*, *14*, 73–79. https://digitalcommons.iwu.edu/parkplace/vol14/iss1/16/

Jaggia, S., Kelly, A., Lertwachara, K., & Chen, L. (2020). Applying the CRISP-DM Framework for Teaching Business Analytics. *Decision Sciences Journal of Innovative Education*, *18*(4), 612–634. https://doi.org/10.1111/dsji.12222

Jiménez, O., Jesús, A., & Wong, L. (2023). Model for the Prediction of Dropout in Higher Education in Peru applying Machine Learning Algorithms: Random Forest, Decision Tree, Neural Network and Support Vector Machine. *2023 33rd Conference of Open Innovations Association (FRUCT)*, 116–124. https://doi.org/10.23919/FRUCT58615.2023.10143068

Karabacak, E. S., & Yaslan, Y. (2023). Comparison of Machine Learning Methods for Early

Detection of Student Dropouts. *2023 8th International Conference on Computer Science and Engineering (UBMK)*, 376–381. https://doi.org/10.1109/UBMK59864.2023.10286747

Kok, C. L., Ho, C. K., Chen, L., Koh, Y. Y., & Tian, B. (2024). *2. A Novel Predictive Modeling for Student Attrition by Utilizing Machine Learning and Sustainable Big Data Analytics*. https://doi.org/10.20944/preprints202408.1298.v1

Krüger, J. G. C., Britto, A. de S., & Barddal, J. P. (2023). An explainable machine learning approach for student dropout prediction. *Expert Systems with Applications*, *233*, 120933. https://doi.org/10.1016/j.eswa.2023.120933

Kuz, A., & Morales, R. (2023). Ciencia de Datos Educativos y aprendizaje automático: un caso de estudio sobre la deserción estudiantil universitaria en México. *Education in the Knowledge Society (EKS)*, *24*, e30080. https://doi.org/10.14201/eks.30080

Mariscal, G., Marbán, Ó., & Fernández, C. (2010). A survey of data mining and knowledge discovery process models and methodologies. *The Knowledge Engineering Review*, *25*(2), 137–166. https://doi.org/10.1017/S0269888910000032

Martelo, R., Herrera, K., & Villabona, N. (2017). Estrategias para disminuir la deserción universitaria mediante series de tiempo y multipol. *Revista Espacios*, *38*(45), 25. https://www.revistaespacios.com/a17v38n45/17384525.html

Martins, M. V., Baptista, L., Machado, J., & Realinho, V. (2023). Multi-Class Phased Prediction of Academic Performance and Dropout in Higher Education. *Applied Sciences*, *13*(8), 4702. https://doi.org/10.3390/app13084702

Melgarejo Garcilazo, J. (2019). *MODELOS POLINOMIALES Y MODELOS RADICALES DE SERIES DE TIEMPO APLICANDO TRANSFORMACIONES LINEALES*

[UNIVERSIDAD NACIONAL SANTIAGO ANTUNEZ DE MAYOLO]. https://repositorio.unasam.edu.pe/bitstream/handle/UNASAM/4193/T033_43453269_T.pdf?sequence=1&isAllowed=y

Melgarejo Garcilazo, J. D. (2020). *Modelos polinomiales y modelos radicales de series de tiempo aplicando transformaciones lineales*. Universidad Nacional Santiago Antúnez de Mayolo.

MEN (Ministerio de Educación Nacional). (n.d.). *Educación Superior*.

MEN (Ministerio de Educación Nacional). (2023). *SPADIES - Sistema para la Prevención de la Deserción en la Educación Superior*.

Ministerio de Educación Nacional. (2009). *Deserción estudiantil en la educación superior Colombiana*. Ministerio de Educación Nacional. https://www.mineducacion.gov.co/sistemasdeinformacion/1735/articles-254702_libro_desercion.pdf

Ministerio de Educación Nacional. (2023). *ESTADÍSTICAS DE DESERCIÓN Y PERMANENCIA EN EDUCACIÓN SUPERIOR SPADIES 3.0 - Indicadores 2021*. https://www.mineducacion.gov.co/sistemasinfo/spadies/secciones/Estadisticas-de-desercion/

Miranda, C. E., Salcedo-Morillo, D. D., & Sánchez-Trochez, G. (2022). El aprendizaje automático en entornos educativos universitarios. Caso deserción académica. *Prospectiva*, *20*(1). https://dialnet.unirioja.es/servlet/articulo?codigo=8297785

Nemtcan, E., Sæle, R. G., Gamst-Klaussen, T., & Svartdal, F. (2020). Drop-Out and Transfer-Out Intentions: The Role of Socio-Cognitive Factors. *Frontiers in Education*, *5*. https://doi.org/10.3389/feduc.2020.606291

Noltemeyer, A. L., Ward, R. M., & Mcloughlin, C. (2015). Relationship between school suspension and student outcomes: A meta-analysis. *School Psychology Review*, *44*(2), 224–240.

Ortiz-Lozano, J. M., Aparicio-Chueca, P., Triadó-Ivern, X. M., & Arroyo-Barrigüete, J. L. (2024). Early dropout predictors in social sciences and management degree students. *Studies in Higher Education*, *49*(8), 1303–1316. https://doi.org/10.1080/03075079.2023.2264343

Pachay-López, M., & Rodríguez-Gámez, M. (2021). La deserción escolar: Una perspectiva compleja en tiempos de pandemia. *Polo Del Conocimiento*, *6*(1), 130–155.

Patel, D., Savaj, K., Malani, P., Patel, J., & Trivedi, H. (2024). Unlocking Enigmatic Pathways: Empowering Student Dropout Analysis with Machine Learning and Energizing Holistic Investigation. *2024 IEEE 9th International Conference for Convergence in Technology (I2CT)*, 1–7. https://doi.org/10.1109/I2CT61223.2024.10543438

Patel, K. K., & Amin, K. (2024). Predictive modeling of dropout in MOOCs using machine learning techniques. *The Scientific Temper*, *15*(02), 2199–2206. https://doi.org/10.58414/SCIENTIFICTEMPER.2024.15.2.32

Peña, M., & Toledo, C. (2017). Ser pobre en el Chile Neoliberal: Estudio discursivo en una escuela vulnerable. *Revista Latinoamericana de Ciencias Sociales, Niñez y Juventud*, *25*(1), 207–218. https://doi.org/https://doi.org/10.11600/1692715x.1511225012016

Pérez-Ramírez, F. (2010). *Modelos ARIMA-ARCH. Algunas aplicaciones a las series de tiemp financieras*. Universidad de Medellin.

Prieto-Romero, A. M., Chanchí-Golondrino, G. E., & Ospina-Alarcón, M. A. (2024). Time series model for the characterization and prediction of the graduation rate at the University of

Cartagena. *Revista de Investigación, Desarrollo e Innovación*, *14*(2), 25–42. https://doi.org/10.19053/uptc.20278306.v14.n2.2024.17921

Rivera Vergaray, K. (2021). Modelo predictivo para la detección temprana de estudiantes con alto riesgo de deserción académica. *Innovación y Software*, *2*(2), 6–13. https://doi.org/10.48168/innosoft.s6.a40

Rochin Berumen, F. L. (2021). Deserción escolar en la educación superior en México: revisión de literatura. *RIDE Revista Iberoamericana Para La Investigación y El Desarrollo Educativo*, *11*(22). https://doi.org/10.23913/ride.v11i22.821

Romero-Contreras, K. Y., Castillo-Gil, D., Higuera-Hurtado, D. J., & Villalba-Gómez, C. E. (2022). Factores influyentes de la deserción estudiantil en la Universidad de La Salle (2018-2020). *Virtu@lmente*, *9*(2). https://doi.org/10.21158/2357514x.v9.n2.2021.3196

Sabbir, W., Abdullah-Al-Kafi, M., Afridi, A. S., Rahman, M. S., & Karmakar, M. (2024). Improving Predictive Analytics for Student Dropout: A Comprehensive Analysis and Model Evaluation. *2024 11th International Conference on Computing for Sustainable Global Development (INDIACom)*, 951–956. https://doi.org/10.23919/INDIACom61295.2024.10498520

Sáez, F., López, Y., Cobo, R., & Mella, J. (2020). Revisión sistemática sobre intención de abandono en educacion superior. *IX Conferencia Latinoamericana Sobre El Abandono En La Educación Superior*, *500*, 91–100.

Sánchez-Sánchez, P. A., & García-González, J. R. (2017). A New Methodology for Neural Network Training Ensures Error Reduction in Time Series Forecasting. *Journal of Computer Science*, *13*(7), 211–217. https://doi.org/10.3844/jcssp.2017.211.217

Sánchez, P., & Velásquez, J. (2010). Problemas de investigación en la predicción de series de

tiempo con redes neuronales artificiales. *Avances En Sistemas e Informática*, *7*(3), 67–73.

Sani, G., Oladipo, F., Ogbuju, E., & Agbo, F. J. (2022). Development of a Predictive Model of Student Attrition Rate. *Journal of Applied Artificial Intelligence*, *3*(2), 1–12. https://doi.org/10.48185/jaai.v3i2.601

Segura, M., Mello, J., & Hernández, A. (2022). Machine Learning Prediction of University Student Dropout: Does Preference Play a Key Role? *Mathematics*, *10*(18), 3359. https://doi.org/10.3390/math10183359

Shafique, U., & Qaiser, H. (2014). A comparative study of data mining process models (KDD, CRISP-DM and SEMMA). *International Journal of Innovation and Scientific Research*, *12*(1), 217–222.

Smith Uldall, J., & Gutiérrez Rojas, C. (2022). An Application of Machine Learning in Public Policy: Early Warning Prediction of School Dropout in the Chilean Public Education System. *Multidisciplinary Business Review*, *15*(1), 20–35. https://doi.org/10.35692/07183992.15.1.4

Sommer, M., & Dumont, K. (2011). Psychosocial factors predicting academic performance of students at a historically disadvantaged university. *South African Journal of Psychology*, *41*(3), 386–395.

Song, Z., Sung, S.-H., Park, D.-M., & Park, B.-K. (2023). All-Year Dropout Prediction Modeling and Analysis for University Students. *Applied Sciences*, *13*(2), 1143. https://doi.org/10.3390/app13021143

Statsmodels. (2023). *Statsmodels*. https://www.statsmodels.org/stable/index.html

Ugoh, C. I., Echebiri, U. V., Temisan, G. O., Iwuchukwu, J. K., & Guobadia, E. K. (2022). On Forecasting Nigeria's GDP: A Comparative Performance of Regression with ARIMA

Errors and ARIMA Method. *International Journal of Mathematics and Statistics Studies*, *10*(4), 48–64. https://doi.org/10.37745/ijmss.13/vol10n44864

Urteaga, I., Siri, L., & Garófalo, G. (2020). Predicción temprana de deserción mediante aprendizaje automático en cursos profesionales en línea. *Revista Iberoamericana de Educación a Distancia*, *23*(2), 169–182. https://redined.educacion.gob.es/xmlui/handle/11162/201572

Valero Cajahuanca, J. E., Navarro Raymundo, Á. F., Larios Franco, A. C., & Julca Flores, J. D. (2022). Deserción universitaria: evaluación de diferentes algoritmos de Machine Learning para su predicción. *Revista de Ciencias Sociales, ISSN-e 1315-9518, Vol. 28, Nº. 3, 2022, Págs. 362-375*, *28*(3), 362–375. https://dialnet.unirioja.es/servlet/articulo?codigo=8526463&info=resumen&idioma=ENG

Vega-García, R., Vázquez-Alamilla, M., Flores-Jiménez, R., Flores-Jiménez, I., & Rodríguez-Moreno, R. (2014). Propuesta para analizar la calidad educativa y deserción escolar a nivel superior en el estado de Hidalgo. Caso de un Instituto Tecnológico Superior en el occidente del estado de Hidalgo. *Boletín Científico de La Escuela Superior de Tlahuelilpan*, *2*(3). https://www.uaeh.edu.mx/scige/boletin/tlahuelilpan/n3/e6.html

Velez, A.; López, D. (2004). Estrategias para vencer la deserción universitaria. *Educación y Educadores, 7*.

Won, H.-S., Kim, M.-J., Kim, D., Kim, H.-S., & Kim, K.-M. (2023). University Student Dropout Prediction Using Pretrained Language Models. *Applied Sciences*, *13*(12), 7073. https://doi.org/10.3390/app13127073

Apéndices

Apendice A

Artículo Publicado en la Revista Ingenierías de la UDEM

HOME / ARCHIVES / VOL. 23 NO. 44 (2024): JANUARY-JUNE / ARTICLES

PROPOSAL OF A TIME SERIES-BASED MODEL FOR THE CHARACTERIZATION AND PREDICTION OF DROPOUT RATES AT THE NATIONAL OPEN AND DISTANCE UNIVERSITY

PDF

Submitted: Dec 1, 2023

Published: May 15, 2024

DOI: https://doi.org/10.22395/rium.v23n44a7

Gabriel Elías Chanchí G.
Universidad de Cartagena
http://orcid.org/0000-0002-0257-1988

Luis Fernando Monroy Gómez
Universidad Nacional Abierta y a Distancia - UNAD

Dayana Alejandra Barrera Buitrago
Universidad Loyola

*Nota***.** Artículo publicado en revista científica. .

Apendice B

Base de Datos de Deserción Obtenidos Del SPADIES

Fecha	Tasa Deserción %	Fecha	Tasa Deserción %
30/06/1998	22,303125	31/12/2010	25,07
31/12/1998	22,303125	30/06/2011	23,4
30/06/1999	31,62	31/12/2011	20,41
31/12/1999	30,39	30/06/2012	22,27
30/06/2000	23,65	31/12/2012	20,38
31/12/2000	19,5	30/06/2013	22,27
30/06/2001	18,96	31/12/2013	22,68
31/12/2001	18,39	30/06/2014	21,48
30/06/2002	17,74	31/12/2014	22,51
31/12/2002	17,68	30/06/2015	24,75
30/06/2003	19,01	31/12/2015	22,84
31/12/2003	19,7	30/06/2016	24,27
30/06/2004	20,48	31/12/2016	23,13
31/12/2004	23,55	30/06/2017	24,32
30/06/2005	22,39	31/12/2017	21,54
31/12/2005	45,42	30/06/2018	23,06
30/06/2006	39,1	31/12/2018	19,47
31/12/2006	21,35	30/06/2019	20,85
30/06/2007	22,23	31/12/2019	20,62
31/12/2007	33,16	30/06/2020	17,87

Fecha	Tasa Deserción %	Fecha	Tasa Deserción %
30/06/2008	2,91	31/12/2020	21,43
31/12/2008	32,57	30/06/2021	15,67
30/06/2009	23,84	31/12/2021	14
31/12/2009	24,14	30/06/2022	9,98
30/06/2010	26,76	31/12/2022	11,74

*Nota***.** Datos de deserción obtenidos del SPADIES. .

Apendice C

Código Fuente del Prototipo Software Implementado

```
import tkinter as tk
from tkinter import ttk, filedialog, messagebox
import pandas as pd
import matplotlib.pyplot as plt
from matplotlib.backends.backend_tkagg import FigureCanvasTkAgg
from statsmodels.tsa.stattools import adfuller
from statsmodels.graphics.tsaplots import plot_pacf, plot_acf
from statsmodels.tsa.arima.model import ARIMA
import numpy as np
from sklearn.metrics import mean_squared_error, mean_absolute_error
import warnings
warnings.filterwarnings("ignore")
# Función para mostrar la gráfica dentro de la interfaz
def mostrar_grafica(figura, frame):
    for widget in frame.winfo_children():
        widget.destroy()
    canvas = FigureCanvasTkAgg(figura, master=frame)
    canvas.draw()
    canvas.get_tk_widget().pack(side=tk.TOP, fill=tk.BOTH, expand=True)
# Funciones generales
def cargar_archivo():
```

```
    archivo = filedialog.askopenfilename(filetypes=[("Excel files", "*.xlsx")])
    ruta_archivo.set(archivo)
    if archivo:
        global df
        df = pd.read_excel(archivo)
        df['Fecha'] = pd.to_datetime(df['Fecha'])
        messagebox.showinfo("Archivo Cargado", "El archivo se cargó correctamente.")
    else:
        messagebox.showerror("Error", "No se ha cargado ningún archivo.")
def graficar_dataset():
    if df is not None:
        fig, ax = plt.subplots(figsize=(10, 6))
        ax.plot(df['Fecha'], df['Tasa_Por'], label='Tasa de Deserción')
        ax.set_xlabel('Fecha')
        ax.set_ylabel('Tasa de Deserción')
        ax.set_title('Gráfico de la serie completa')
        ax.legend()
        ax.grid(True)
        mostrar_grafica(fig, frame_grafico)
    else:
        messagebox.showerror("Error", "Cargue un archivo primero.")
# Función ajustada para visualizar el entrenamiento y prueba
def visualizar_entrenamiento_prueba():
```

```
            porcentaje = float(porcentaje_entrenamiento.get()) / 100
            fecha_corte = df['Fecha'].quantile(porcentaje)
            train = df[df['Fecha'] <= fecha_corte]
            test = df[df['Fecha'] > fecha_corte]
            fig, ax = plt.subplots(figsize=(10, 6))
            # Unir los conjuntos de entrenamiento y prueba en una sola línea continua
            ax.plot(df['Fecha'], df['Tasa_Por'], color='blue', linestyle='-', label='Conjunto
Completo')
            # Sobreponer la parte de prueba con diferente color y marcadores
            ax.plot(test['Fecha'], test['Tasa_Por'], color='red', linestyle='-', marker='o',
label='Conjunto de Prueba')
            ax.set_xlabel('Fecha')
            ax.set_ylabel('Tasa de Deserción')
            ax.set_title('Conjunto de Entrenamiento vs Conjunto de Prueba')
            ax.legend()
            ax.grid(True)

            # Mostrar la gráfica en el frame de la segunda pestaña
            mostrar_grafica(fig, frame_grafico_tab2)
        def calcular_dickey_fuller():
            train_size = int(len(df) * 0.75)
            train, test = df['Tasa_Por'][:train_size], df['Tasa_Por'][train_size:]
            serie = train
```

```
        d = int(combo_d.get())
        area_resultados_d.delete("1.0", tk.END)  # Limpiar el área de texto antes de mostrar nuevos resultados
        # Caso cuando d == 0 (prueba solo en la serie original)
        if d == 0:
            resultado_original = adfuller(serie.dropna())
            area_resultados_d.insert(tk.END, "Resultados para la serie original:\n")
            area_resultados_d.insert(tk.END, f"ADF Statistic: {resultado_original[0]:.4f}\n")
            area_resultados_d.insert(tk.END, f"p-value: {resultado_original[1]:.4f}\n\n")
        # Caso cuando d == 1 (prueba en serie original y serie diferenciada una vez)
        if d == 1:
            # Prueba en la serie original
            resultado_original = adfuller(serie.dropna())
            area_resultados_d.insert(tk.END, "Resultados para la serie original:\n")
            area_resultados_d.insert(tk.END, f"ADF Statistic: {resultado_original[0]:.4f}\n")
            area_resultados_d.insert(tk.END, f"p-value: {resultado_original[1]:.4f}\n\n")
            # Prueba en la serie diferenciada una vez
            serie_diff1 = serie.diff().dropna()
            resultado_diff1 = adfuller(serie_diff1)
            area_resultados_d.insert(tk.END, "Resultados para la primera diferenciación:\n")
            area_resultados_d.insert(tk.END, f"ADF Statistic: {resultado_diff1[0]:.4f}\n")
            area_resultados_d.insert(tk.END, f"p-value: {resultado_diff1[1]:.4f}\n\n")
        # Caso cuando d == 2 (prueba en serie original, primera y segunda diferenciación)
```

```
    if d == 2:
        # Prueba en la serie original
        resultado_original = adfuller(serie.dropna())
        area_resultados_d.insert(tk.END, "Resultados para la serie original:\n")
        area_resultados_d.insert(tk.END, f"ADF Statistic: {resultado_original[0]:.4f}\n")
        area_resultados_d.insert(tk.END, f"p-value: {resultado_original[1]:.4f}\n\n")
        # Prueba en la serie diferenciada una vez
        serie_diff1 = serie.diff().dropna()
        resultado_diff1 = adfuller(serie_diff1)
        area_resultados_d.insert(tk.END, "Resultados para la primera diferenciación:\n")
        area_resultados_d.insert(tk.END, f"ADF Statistic: {resultado_diff1[0]:.4f}\n")
        area_resultados_d.insert(tk.END, f"p-value: {resultado_diff1[1]:.4f}\n\n")
        # Prueba en la serie diferenciada dos veces
        serie_diff2 = serie_diff1.diff().dropna()
        resultado_diff2 = adfuller(serie_diff2)
        area_resultados_d.insert(tk.END, "Resultados para la segunda diferenciación:\n")
        area_resultados_d.insert(tk.END, f"ADF Statistic: {resultado_diff2[0]:.4f}\n")
        area_resultados_d.insert(tk.END, f"p-value: {resultado_diff2[1]:.4f}\n\n")
# Función ajustada para mostrar autocorrelaciones en la cuarta pestaña
def visualizar_autocorrelaciones():
    serie = df['Tasa_Por']
    p_q_d_value = int(combo_p_q.get())
    if p_q_d_value >= 1:
```

```
            serie = serie.diff().dropna()
        if p_q_d_value == 2:
            serie = serie.diff().dropna()
        fig, (ax1, ax2) = plt.subplots(2, 1, figsize=(10, 8))
        plot_acf(serie, ax=ax1, lags=20)
        plot_pacf(serie, ax=ax2, lags=20)
        ax1.set_title('Autocorrelación')
        ax2.set_title('Autocorrelación Parcial')
        plt.tight_layout()
        # Mostrar la gráfica en el frame de la cuarta pestaña
        mostrar_grafica(fig, frame_grafico_tab4)
    def evaluar_modelos():
        area_resultados_ajuste.delete("1.0", tk.END)  # Limpiar el área de texto antes de
mostrar nuevos resultados
        train_size = int(len(df) * 0.75)
        train, test = df['Tasa_Por'][:train_size], df['Tasa_Por'][train_size:]
        p = int(p_val.get())
        q = int(q_val.get())
        d = int(d_val.get())
        dicc_ARIMA={}
        for i in range(0, p+1):
            for j in range(0, d+1):
                for k in range(0, q+1):
```

```
                    if i == 0 and j == 0 and k == 0:
                        pass
                    else:
                        modo = (i, j, k)
                        model_x = ARIMA(train, order=modo)
                        model_x_fit = model_x.fit()
                        # Escribir los resultados en el área de texto
                        area_resultados_ajuste.insert(tk.END,
"========================\n")
                        area_resultados_ajuste.insert(tk.END, f"Modo {modo}\n")
                        area_resultados_ajuste.insert(tk.END, f"AIC: {round(model_x_fit.aic,
3)}\n")
                        area_resultados_ajuste.insert(tk.END, f"BIC: {round(model_x_fit.bic,
3)}\n")
                        area_resultados_ajuste.insert(tk.END, f"Log Likelihood:
{round(model_x_fit.llf)}\n")
                        lista_p=model_x_fit.pvalues.tolist()
                        if all(p < 0.05 for p in lista_p):
                            area_resultados_ajuste.insert(tk.END, f"Pasa la prueba \n")
dicc_ARIMA["ARIMA("+str(i)+","+str(j)+","+str(k)+")"]={"AIC":round(model_x_fit.aic, 3),
"BIC":round(model_x_fit.bic, 3)}
                        else:
                        area_resultados_ajuste.insert(tk.END, f"No Pasa la prueba \n")
```

```
                area_resultados_ajuste.insert(tk.END, f"P_values: {lista_p}\n")
                area_resultados_ajuste.insert(tk.END, "\n")
    area_resultados_ajuste.insert(tk.END, "\n")
    area_resultados_ajuste.insert(tk.END, "Modelos significativos:\n")
    for modelo, metricas in dicc_ARIMA.items():
        resultado = f"{modelo} -> AIC: {metricas['AIC']}, BIC: {metricas['BIC']}\n"
        area_resultados_ajuste.insert(tk.END, resultado)
def ajustar_evaluar_modelo():
    p = int(p_val_ajuste.get())
    q = int(q_val_ajuste.get())
    d = int(d_val_ajuste.get())
    train_size = int(len(df) * 0.75)
    train, test = df['Tasa_Por'][:train_size], df['Tasa_Por'][train_size:]
    modelo = ARIMA(train, order=(p, d, q))
    modelo_fit = modelo.fit()
    pred_train = modelo_fit.predict(start=train.index[0], end=train.index[-1], typ='levels')
    pred_test = modelo_fit.predict(start=test.index[0], end=test.index[-1], typ='levels')
    mse_train = mean_squared_error(train, pred_train)
    rmse_train = np.sqrt(mse_train)
    mae_train = mean_absolute_error(train, pred_train)
    mse_test = mean_squared_error(test, pred_test)
    rmse_test = np.sqrt(mse_test)
    mae_test = mean_absolute_error(test, pred_test)
```

```
            fig, ax = plt.subplots(figsize=(4, 2))
            ax.plot(train.index, train, label='Entrenamiento', color='blue')
            ax.plot(test.index, test, label='Prueba', color='red')
            ax.plot(train.index, pred_train, label='Predicción Entrenamiento', color='green',
linestyle='dashed')
            ax.plot(test.index, pred_test, label='Predicción Prueba', color='orange',
linestyle='dashed')
            ax.set_xlabel('Fecha')
            ax.set_ylabel('% Deserción')
            ax.set_title('Comparación del modelo ARIMA y los datos reales')
            ax.legend()
            ax.grid(True)
            mostrar_grafica(fig, frame_grafico_tab6)

            area_resultados_ajuste_m.delete("1.0", tk.END)
            area_resultados_ajuste_m.insert(tk.END, f"Resultados Entrenamiento:\nMSE:
{mse_train:.6f}\nMAE: {mae_train:.6f}\nRMSE: {rmse_train:.6f}\n")
            area_resultados_ajuste_m.insert(tk.END, f"Resultados Prueba:\nMSE:
{mse_test:.6f}\nMAE: {mae_test:.6f}\nRMSE: {rmse_test:.6f}\n")
        def predecir_modelo():
            p = int(p_val_pred.get())
            q = int(q_val_pred.get())
            d = int(d_val_pred.get())
```

```
            semestres = int(semestres_pred.get())
            train_size = int(len(df) * 0.75)
            train = df['Tasa_Por'][:train_size]
            modelo = ARIMA(train, order=(p, d, q))
            modelo_fit = modelo.fit()
            prediccion = modelo_fit.forecast(steps=semestres)
            area_resultados_prediccion.delete("1.0", tk.END)
            fechas_pred = pd.date_range(df['Fecha'].iloc[train_size], periods=semestres,
freq='6M')
            for fecha, valor in zip(fechas_pred, prediccion):
                area_resultados_prediccion.insert(tk.END, f"{fecha.date()}\t{valor:.6f}\n")
        # Crear la ventana principal
        ventana = tk.Tk()
        ventana.title("App de Predicción ARIMA")
        ventana.geometry("900x700")
        # Crear las pestañas
        tab_control = ttk.Notebook(ventana)
        # Crear un frame para las gráficas
        frame_grafico = tk.Frame(ventana)
        frame_grafico.pack(side=tk.BOTTOM, fill=tk.BOTH, expand=True)
        # Primera pestaña: Dataset
        tab1 = ttk.Frame(tab_control)
        tab_control.add(tab1, text='Dataset')
```

```
        ruta_archivo = tk.StringVar()
        # Frame para los controles superiores
        frame_controles = tk.Frame(tab1)
        frame_controles.pack(side=tk.TOP, fill=tk.X, padx=10, pady=10)
        # Etiqueta y cuadro de texto para la ruta del archivo
        ttk.Label(frame_controles, text="Ruta del Archivo:").pack(side=tk.LEFT, padx=5)
        ruta_entry = ttk.Entry(frame_controles, textvariable=ruta_archivo, width=50)
        ruta_entry.pack(side=tk.LEFT, padx=5)
        # Botón para cargar archivo
        boton_cargar = ttk.Button(frame_controles, text="Cargar", command=cargar_archivo)
        boton_cargar.pack(side=tk.LEFT, padx=5)
        # Botón para graficar
        boton_graficar = ttk.Button(frame_controles, text="Graficar",
command=graficar_dataset)
        boton_graficar.pack(side=tk.LEFT, padx=5)
        # Frame para la gráfica
        frame_grafico = tk.Frame(tab1)
        frame_grafico.pack(side=tk.BOTTOM, fill=tk.BOTH, expand=True)
        # Segunda pestaña: Entrenamiento y Prueba
        tab2 = ttk.Frame(tab_control)
        tab_control.add(tab2, text='Entrenamiento y Prueba')
        # Usamos un Frame para contener los controles
        frame_controles_tab2 = tk.Frame(tab2)
```

```
        frame_controles_tab2.pack(side=tk.TOP, pady=10)
        # Colocar la etiqueta, campo de texto y botón en la misma fila
        porcentaje_entrenamiento = tk.StringVar(value="75")
        ttk.Label(frame_controles_tab2, text="Porcentaje Entrenamiento").grid(row=0,
column=0, padx=5)
        porcentaje_entry = ttk.Entry(frame_controles_tab2,
textvariable=porcentaje_entrenamiento, width=10)
        porcentaje_entry.grid(row=0, column=1, padx=5)
        # Botón Visualizar en la misma fila
        boton_visualizar = ttk.Button(frame_controles_tab2, text="Visualizar",
command=visualizar_entrenamiento_prueba)
        boton_visualizar.grid(row=0, column=2, padx=5)
        # Frame para la gráfica en la segunda pestaña
        frame_grafico_tab2 = tk.Frame(tab2)
        frame_grafico_tab2.pack(side=tk.BOTTOM, fill=tk.BOTH, expand=True)
        # Tercera pestaña: Parámetro d
        tab3 = ttk.Frame(tab_control)
        tab_control.add(tab3, text='Parámetro d')
        # Usamos un Frame para contener los controles
        frame_controles_tab3 = tk.Frame(tab3)
        frame_controles_tab3.pack(side=tk.TOP, pady=10)
        # Colocar la etiqueta "Series", el combobox y el botón "Calcular" en la misma fila
        ttk.Label(frame_controles_tab3, text="Series").grid(row=0, column=0, padx=5)
```

```
        combo_d = ttk.Combobox(frame_controles_tab3, values=[0, 1, 2], state='readonly')
        combo_d.current(0)
        combo_d.grid(row=0, column=1, padx=5)
        # Botón Calcular en la misma fila
        boton_calcular_d = ttk.Button(frame_controles_tab3, text="Calcular",
command=calcular_dickey_fuller)
        boton_calcular_d.grid(row=0, column=2, padx=5)
        # Área de resultados en la parte inferior
        area_resultados_d = tk.Text(tab3, height=28)
        area_resultados_d.pack(pady=10)
        # Cuarta pestaña: Parámetros p y q
        tab4 = ttk.Frame(tab_control)
        tab_control.add(tab4, text='Parámetros p y q')
        # Frame para los controles de la cuarta pestaña
        frame_controles_tab4 = tk.Frame(tab4)
        frame_controles_tab4.pack(side=tk.TOP, pady=10)
        # Etiqueta "Diferenciación" al lado del combobox
        ttk.Label(frame_controles_tab4, text="Diferenciación").grid(row=0, column=0, padx=5)
        combo_p_q = ttk.Combobox(frame_controles_tab4, values=[0, 1, 2], state='readonly')
        combo_p_q.current(0)
        combo_p_q.grid(row=0, column=1, padx=5)
        # Botón Visualizar
```

```
        boton_visualizar_p_q = ttk.Button(frame_controles_tab4, text="Visualizar",
command=visualizar_autocorrelaciones)
        boton_visualizar_p_q.grid(row=0, column=2, padx=5)
        # Frame para la gráfica en la cuarta pestaña
        frame_grafico_tab4 = tk.Frame(tab4)
        frame_grafico_tab4.pack(side=tk.BOTTOM, fill=tk.BOTH, expand=True)
        # Quinta pestaña: Evaluación modelos
        tab5 = ttk.Frame(tab_control)
        tab_control.add(tab5, text='Evaluación modelos')
        # Usamos un Frame para contener los controles
        frame_controles_tab5 = tk.Frame(tab5)
        frame_controles_tab5.pack(side=tk.TOP, pady=20)
        # Colocamos las etiquetas, entradas y botones en la misma fila con 'grid'
        ttk.Label(frame_controles_tab5, text="Valor p:").grid(row=0, column=0, padx=10)
        p_val = tk.StringVar(value="1")
        ttk.Entry(frame_controles_tab5, textvariable=p_val, width=5).grid(row=0, column=1,
padx=10)
        ttk.Label(frame_controles_tab5, text="Valor q:").grid(row=0, column=2, padx=10)
        q_val = tk.StringVar(value="1")
        ttk.Entry(frame_controles_tab5, textvariable=q_val, width=5).grid(row=0, column=3,
padx=10)
        ttk.Label(frame_controles_tab5, text="Valor d:").grid(row=0, column=4, padx=10)
        d_val = tk.StringVar(value="1")
```

```
        ttk.Entry(frame_controles_tab5, textvariable=d_val, width=5).grid(row=0, column=5,
padx=10)
        # Botón Ajustar y Evaluar en la misma fila
        boton_ajustar_evaluar = ttk.Button(frame_controles_tab5, text="Ajustar y Evaluar",
command=evaluar_modelos)
        boton_ajustar_evaluar.grid(row=0, column=6, padx=10)
        # Área de resultados en la parte inferior
        area_resultados_ajuste = tk.Text(tab5, height=30,width=80)
        area_resultados_ajuste.pack(side=tk.TOP, anchor='n', pady=10)
        # Sexta pestaña: Ajuste modelo
        tab6 = ttk.Frame(tab_control)
        tab_control.add(tab6, text='Ajuste modelo')
        # Usamos un Frame para los controles (etiquetas, campos de texto, botón)
        frame_controles_tab6 = tk.Frame(tab6)
        frame_controles_tab6.pack(side=tk.TOP, pady=10)
        # Colocar las etiquetas, campos de texto y botón en la misma fila
        p_val_ajuste = tk.StringVar(value="1")
        q_val_ajuste = tk.StringVar(value="1")
        d_val_ajuste = tk.StringVar(value="1")
        ttk.Label(frame_controles_tab6, text="Valor p").grid(row=0, column=0, padx=5)
        ttk.Entry(frame_controles_tab6, textvariable=p_val_ajuste, width=5).grid(row=0,
column=1, padx=5)
        ttk.Label(frame_controles_tab6, text="Valor q").grid(row=0, column=2, padx=5)
```

```
        ttk.Entry(frame_controles_tab6, textvariable=q_val_ajuste, width=5).grid(row=0,
column=3, padx=5)
        ttk.Label(frame_controles_tab6, text="Valor d").grid(row=0, column=4, padx=5)
        ttk.Entry(frame_controles_tab6, textvariable=d_val_ajuste, width=5).grid(row=0,
column=5, padx=5)
        # Botón Ajustar y Evaluar en la misma fila
        boton_ajustar_evaluar = ttk.Button(frame_controles_tab6, text="Ajustar y Evaluar",
command=ajustar_evaluar_modelo)
        boton_ajustar_evaluar.grid(row=0, column=6, padx=5)
        # Ajustar el tamaño del área de texto para que no ocupe toda la pantalla
        area_resultados_ajuste_m = tk.Text(tab6, height=10, width=80)  # Ajusta la altura y el
ancho
        area_resultados_ajuste_m.pack(side=tk.TOP, anchor='n', pady=10)

        # Frame para la gráfica en la sexta pestaña
        frame_grafico_tab6 = tk.Frame(tab6)
        frame_grafico_tab6.pack(side=tk.BOTTOM, fill=tk.BOTH, expand=True)
        # Séptima pestaña: Predicción
        tab7 = ttk.Frame(tab_control)
        tab_control.add(tab7, text='Predicción')
        # Usamos un Frame para los controles (etiquetas, campos de texto, botón)
        frame_controles_tab7 = tk.Frame(tab7)
        frame_controles_tab7.pack(side=tk.TOP, pady=10)
```

```
        p_val_pred = tk.StringVar(value="1")
        q_val_pred = tk.StringVar(value="1")
        d_val_pred = tk.StringVar(value="1")
        semestres_pred = tk.StringVar(value="4")
        # Colocar las etiquetas, campos de texto y botón en la misma fila
        ttk.Label(frame_controles_tab7, text="Valor p").grid(row=0, column=0, padx=5)
        ttk.Entry(frame_controles_tab7, textvariable=p_val_pred, width=5).grid(row=0,
column=1, padx=5)
        ttk.Label(frame_controles_tab7, text="Valor q").grid(row=0, column=2, padx=5)
        ttk.Entry(frame_controles_tab7, textvariable=q_val_pred, width=5).grid(row=0,
column=3, padx=5)
        ttk.Label(frame_controles_tab7, text="Valor d").grid(row=0, column=4, padx=5)
        ttk.Entry(frame_controles_tab7, textvariable=d_val_pred, width=5).grid(row=0,
column=5, padx=5)
        ttk.Label(frame_controles_tab7, text="Semestres a predecir").grid(row=0, column=6,
padx=5)
        ttk.Entry(frame_controles_tab7, textvariable=semestres_pred, width=5).grid(row=0,
column=7, padx=5)
        # Botón Predecir en la misma fila
        boton_predecir = ttk.Button(frame_controles_tab7, text="Predecir",
command=predecir_modelo)
        boton_predecir.grid(row=0, column=8, padx=5)
        # Ajustar el tamaño del área de texto para que no ocupe toda la pantalla
```

```
        area_resultados_prediccion = tk.Text(tab7, height=28, width=80)  # Ajusta la altura y el
ancho
        area_resultados_prediccion.pack(side=tk.TOP, anchor='n', pady=10)
        # Agregar las pestañas al control
        tab_control.pack(expand=1, fill="both")
        ventana.mainloop()
```

*Nota***.** Código fuente del prototipo software. .