PONTIFICIA UNIVERSIDAD CATÓLICA DEL ECUADOR

FACULTAD DE INGENIERÍA

TEMA:

PREDICCIÓN DE VENTAS EN TIENDAS DE RETAIL FARMACEÚTICO ECUATORIANO MEDIANTE MODELOS AGRUPADOS DE APRENDIZAJE AUTOMÁTICO Y SERIES TEMPORALES.

AUTOR:

BECERRA ORTIZ ALEXANDER DAVID

DIRECTOR:

ORTIZ NAVARRETE MIGUEL DIMITRI

TRABAJO PREVIO A LA OBTENCIÓN DEL TÍTULO DE MAGÍSTER EN SISTEMAS DE INFORMACIÓN MENCIÓN DATA SCIENCE

QUITO, 2024

Dedicatoria

A la memoria de M. Paulina (QEPD), quien me acompañó durante gran parte de mi vida, brindándome su apoyo incondicional.

Agradecimientos

Agradezco a mi madre por no dudar de mis capacidades e inspirarme cada día a llegar más lejos.

Al Máster Miguel Ortiz, director de esta investigación, le expreso mi gratitud por los conocimientos impartidos, su valioso tiempo y su infinita paciencia, los cuales han sido fundamentales en la realización de esta tesis.

Índice de Contenido

Índice de Figuras

Introducción

1.1 Generalidades

La industria del retail farmacéutico en Ecuador, al igual que en muchos otros países, enfrenta numerosos desafíos relacionados con la precisión de los pronósticos de ventas. Consultoras como IQVIA, en su reciente informe, señalan que la industria farmacéutica en América Latina podría experimentar un crecimiento significativo. Se proyectaba un aumento del 17.4% para 2022 y del 16.6% para 2023, utilizando dólares constantes como unidad de medida. Además, se explica que entre 2022 y 2026, la actividad en la región crecerá a una tasa anual compuesta del 15.2%. En Ecuador, se espera un crecimiento del 4.2% durante el mismo período. La consultora destaca también que el sector institucional tendrá un desempeño superior al retail, con proyecciones de crecimiento del 16.5% en el quinquenio 2022-2026 para el retail en farmacias. (Atance, 2022)

En el entorno competitivo actual, donde la demanda puede ser volátil y está influenciada por múltiples factores internos y externos, las empresas de retail farmacéutico necesitan herramientas avanzadas que les permitan anticiparse a las necesidades del mercado. Estas herramientas no solo deben ser capaces de procesar grandes volúmenes de datos, sino también de identificar patrones complejos que no son evidentes mediante métodos tradicionales.

Uno de los principales factores externos que condujo a la expansión de tiendas de retail farmacéutico en Ecuador fue la pandemia de COVID-19. Este evento resaltó la importancia de la disponibilidad de medicamentos, así como de artículos de desinfección y suministros de limpieza. En 2024, todavía existen pequeños focos de reinfecciones que contribuyen a la volatilidad del fenómeno, además de las enfermedades estacionales que dificultan la precisión de los modelos clásicos de pronósticos.

El avance en técnicas de aprendizaje automático y modelos de series temporales ha abierto nuevas posibilidades para mejorar la precisión de los pronósticos de ventas. Entre estas técnicas, las redes neuronales LSTM (Long Short-Term Memory) destacan por su capacidad para gestionar dependencias a largo plazo en datos secuenciales, lo que las hace particularmente adecuadas para la predicción de ventas en entornos dinámicos y complejos como el retail farmacéutico.

Adicionalmente, la agrupación de puntos de venta en clústers homogéneos mediante técnicas de clustering, como K-means, permite personalizar los modelos de predicción. Esta personalización puede ser esencial para considerar las características específicas de cada punto de venta, así como patrones históricos de ventas, mejorando la precisión en los pronósticos.

Este proyecto propone desarrollar un modelo de predicción de ventas que combine técnicas de aprendizaje automático y series temporales, aplicadas a clústers de puntos de venta en una empresa de retail farmacéutico ecuatoriano. El objetivo es

proporcionar un enfoque más preciso y personalizado que los métodos tradicionales, mejorando la toma de decisiones estratégicas.

Los capítulos siguientes de este trabajo detallarán los antecedentes teóricos, la metodología propuesta, los resultados esperados y las conclusiones derivadas de la implementación y evaluación del modelo de predicción de ventas.

1.2 Planteamiento del problema

El sector de retail farmacéutico ecuatoriano enfrenta un desafío significativo: mejorar la precisión de los pronósticos de ventas para cada tienda de retail. La falta de herramientas precisas para pronosticar ventas puede resultar en pérdidas sustanciales debido a la falta de stock o exceso de inventario, afectando negativamente tanto la rentabilidad como la percepción del cliente. En un entorno altamente competitivo, la demanda puede ser volátil y está influenciada por múltiples factores internos y externos, lo que dificulta aún más la tarea de predecir ventas con precisión.

La pandemia de COVID-19, en particular, subrayó la importancia de la disponibilidad de medicamentos y productos de desinfección, destacando la volatilidad del mercado debido a los cambios en la demanda y patrones de consumo. Aunque la situación ha mejorado en 2024, existen pequeños focos de reinfecciones y enfermedades estacionales que contribuyen a la incertidumbre y volatilidad en las ventas.

Jason Brownlee afirma que la predicción de series temporales ha estado dominada por métodos lineales como ARIMA debido a que son bien comprendidos y efectivos para muchos problemas. Sin embargo, estos métodos clásicos también presentan varias limitaciones. Por ejemplo, se enfocan en datos completos, lo que significa que los datos faltantes o corruptos generalmente no son compatibles. Además, asumen relaciones lineales, excluyendo distribuciones conjuntas más complejas. Otra limitante es la dependencia de una estructura temporal fija, lo que requiere diagnosticar y especificar la relación entre observaciones en diferentes tiempos y el número de observaciones rezagadas usadas como entrada. Se centran en datos univariados, a pesar de que muchos problemas del mundo real tienen múltiples variables de entrada. Por último, señala como limitante de estos métodos el enfocarse en pronósticos de un solo paso, mientras que muchos problemas reales requieren pronósticos con un horizonte temporal largo. (Brownlee, Deep Learning for Time Series Forecasting, 2018)

Los métodos de aprendizaje automático son particularmente adecuados para abordar problemas de predicción de series temporales que implican múltiples variables de entrada, relaciones no lineales complejas y datos incompletos. (Brownlee, Deep Learning for Time-Series Analysis, 2017)

Es por ello la necesidad de explorar enfoques más avanzados y adaptativos, como las técnicas de aprendizaje automático y modelos de series temporales, que manejan grandes volúmenes de datos y detectan patrones complejos. En particular el presente proyecto busca probar si las redes neuronales LSTM (Long Short-Term

Memory) son efectivas para gestionar dependencias a largo plazo en datos secuenciales, ofreciendo una solución prometedora para la predicción de ventas en el sector retail farmacéutico.

Adicionalmente, la agrupación de puntos de venta en clústers homogéneos mediante técnicas de clustering, como por ejemplo K-means, DBSCAN, modelos Jerárquicos, entre otros. Permitan personalizar los modelos de predicción según las características específicas de cada grupo de puntos de venta. Buscando mejorar significativamente la precisión de las predicciones al considerar las particularidades de cada grupo de tiendas.

1.3 Objetivos

1.3.1 Objetivo General

Desarrollar un modelo de predicción de ventas para una empresa de retail farmacéutico en Ecuador, utilizando técnicas avanzadas de aprendizaje automático y series temporales, con el fin de mejorar la precisión de los pronósticos y apoyar la toma de decisiones estratégicas en la empresa.

1.3.2 Objetivos Específicos

1.3.2.1 Recolectar y preprocesar datos históricos de ventas de los puntos de venta.

- Identificar y manejar valores faltantes, datos atípicos y normalizar la información recopilada.

- Consolidar bases de datos provenientes de distintas fuentes para obtener un conjunto de datos integral y coherente.

1.3.2.2 Agrupar los puntos de venta utilizando técnicas de clustering basadas en similitudes de patrones de ventas.

- Analizar las características comunes de venta entre los diferentes puntos de venta para crear clústers homogéneos.
- Validar la efectividad de los clústers formados y ajustar los parámetros del algoritmo de clustering si es necesario.

1.3.2.3 Desarrollar y entrenar modelos de series temporales utilizando técnicas avanzadas de aprendizaje automático como: LSTM (Long Short-Term Memory), ARIMA, Holt Winters y modelos clásicos para cada grupo de puntos de venta.

- Seleccionar los hiperparámetros adecuados para los modelos.
- Entrenar los modelos con los datos preprocesados y validados de cada clúster.
- Evaluar y validar la precisión de los modelos mediante métricas como el error cuadrático medio (RMSE), el error absoluto medio (MAE) y la precisión adaptativa.
- Comparar los resultados obtenidos con los modelos tradicionales.

1.4 Alcance

Este proyecto se enfocará en tiendas de una empresa de retail farmacéutico ecuatoriano. Específicamente en 152 farmacias distribuidas en 11 de las 24 provincias

del Ecuador. Esta diversidad geográfica proporciona una base sólida para un análisis exhaustivo y representativo.

El proyecto implicará la recolección y preprocesamiento de datos históricos de ventas a partir de enero 2019, así como información geográfica de las farmacias. Estos datos serán cruciales para entender los patrones de venta y las características específicas de cada ubicación. La recolección de datos también incluirá la identificación y manejo de valores faltantes, la eliminación de datos atípicos y la normalización de la información para asegurar su calidad y coherencia. Este conjunto de datos consolidado será la base para el desarrollo de modelos predictivos robustos y precisos.

Posteriormente, se procederá a la agrupación de los puntos de venta mediante técnicas de clustering, como K-means. Este método permitirá agrupar las farmacias en clústers homogéneos basados en similitudes en patrones de ventas. La creación de estos clústers facilitará la personalización de los modelos de predicción según las características específicas de cada grupo de farmacias, lo que mejorará la precisión de las predicciones y su aplicabilidad en diferentes contextos.

Finalmente, se desarrollarán y entrenarán modelos de series temporales utilizando técnicas avanzadas de aprendizaje automático como LSTM (Long Short-Term Memory) para cada grupo de farmacias. La precisión de estos modelos será evaluada utilizando métricas como el error cuadrático medio (RMSE), el error absoluto medio (MAE) y la precisión adaptativa.

2 Revisión literaria

2.1 Predicción de Ventas en el Retail Farmacéutico

La predicción de ventas en el sector retail farmacéutico es un área crítica para la optimización de la cadena de suministro y la mejora de la eficiencia operativa. Tradicionalmente, se han utilizado métodos estadísticos como la regresión lineal y los modelos autorregresivos integrados de media móvil (ARIMA) para predecir ventas. Estos métodos, aunque útiles, tienen limitaciones significativas en su capacidad para manejar datos no lineales y estacionales, características comunes en los datos de ventas farmacéuticas. (Brownlee, Deep Learning for Time Series Forecasting, 2018)

El sector farmacéutico enfrenta desafíos únicos debido a la variabilidad de la demanda, influenciada por factores como la estacionalidad de enfermedades, la introducción de nuevos productos, campañas de vacunación y eventos inesperados como pandemias.

Bajo el contexto de tiendas de retail ecuatoriano, la pandemia de COVID-19 destacó la importancia de la precisión en los pronósticos de ventas, dado que la demanda de productos sanitarios y medicamentos experimentó fluctuaciones drásticas.

2.2 Técnicas de Aprendizaje Automático en la Predicción de Ventas

Las técnicas de aprendizaje automático como redes neuronales recurrentes (RNN), y en particular las redes Long Short-Term Memory (LSTM) son capaces de manejar secuencias de datos complejas y capturar dependencias a largo plazo, lo que las hace ideales para la predicción de series temporales. (Brownlee, Deep Learning for Time-Series Analysis, 2017)

Varios estudios han demostrado la efectividad de las LSTM en la mejora de la precisión de los pronósticos de ventas. Por ejemplo, un estudio de Fischer y Krauss publicado en el European Jornal of Operational Research mostró que los modelos LSTM superan a los modelos ARIMA y las redes neuronales feedforward tradicionales en la predicción de series temporales financieras, lo que sugiere que estas técnicas también podrían ser aplicables y beneficiosas en el contexto del retail farmacéutico. (Fischer & Krauss, 2018)

2.3 Clustering en el Retail Farmacéutico

El clustering es una técnica de aprendizaje no supervisado que agrupa datos en clústers basados en similitudes. El clustering en tiendas de retail permite segmentar los puntos de venta en grupos homogéneos, facilitando la personalización de los modelos de predicción y mejorando su precisión. Este enfoque no solo optimiza los pronósticos de ventas, sino que también mejora la gestión del inventario y las estrategias de marketing al adaptarse a las características específicas de cada grupo de puntos de venta. Técnicas como K-means son comúnmente utilizadas para este

propósito, permitiendo identificar patrones en los datos de ventas y agrupar tiendas con características similares. (John, Shobayo, & Ogunleye, 2023)

La segmentación de puntos de venta en clústers homogéneos ofrece múltiples beneficios. Según investigaciones recientes, el clustering puede mejorar significativamente la precisión de los pronósticos de ventas al permitir que los modelos de predicción se adapten a las características específicas de cada grupo de tiendas. Esto es particularmente relevante en las tiendas de retail, donde las diferencias en la ubicación geográfica, el perfil demográfico de los clientes y los patrones históricos de ventas pueden influir considerablemente en la demanda de productos (John, Shobayo, & Ogunleye, 2023).

2.4 Metodología de Ciencia de Datos: CRISP-DM

La propuesta metodológica para este proyecto se centra en desarrollar un modelo de predicción de ventas para una empresa de retail farmacéutico ecuatoriano utilizando una combinación de técnicas avanzadas de ciencia de datos y aprendizaje automático.

La metodología CRISP-DM (Cross Industry Standard Process for Data Mining) será utilizada para estructurar el proceso de desarrollo del modelo de predicción de ventas.

Fase 1: Comprensión del Negocio

El primer paso en la metodología CRISP-DM es la comprensión del negocio. En esta fase, se busca definir claramente los objetivos del proyecto y los requisitos del

negocio. Es crucial entender el contexto del problema y cómo este impacto afecta a la empresa. Esta fase involucra reuniones con stakeholders, análisis de documentos y comprensión de los procesos de negocio para alinear el proyecto con las necesidades estratégicas de la empresa.

Fase 2: Comprensión de los Datos

Posterior al entendimiento del negocio, la siguiente fase es la comprensión de los datos. En ella se recopila toda la información relevante y se evalúa su calidad. Se lleva a cabo un análisis exploratorio de datos (EDA) para identificar patrones, tendencias y posibles problemas de calidad. Esta fase es fundamental para familiarizarse con los datos y asegurar que sean adecuados para los análisis posteriores.

Fase 3: Preparación de los Datos

La tercera fase de la metodología CRISP-DM comprende la preparación de los datos, se realizan varias tareas para garantizar la calidad y consistencia de los datos. Esto incluye la recolección de datos históricos de ventas, tipo de los productos y características del punto de venta como su información geográfica. Se lleva a cabo el preprocesamiento de los datos, que abarca la limpieza, manejo de valores faltantes, eliminación de datos atípicos y normalización de la información. Además, se consolida la información de distintas fuentes para formar un conjunto de datos integral y coherente.

Fase 4: Modelado

La fase de modelado implica la aplicación de diversas técnicas de machine learning para desarrollar modelos predictivos. En el contexto de este proyecto, se utilizan las técnicas de clustering k-means, clustering jerárquico y DBSCAN para segmentar los puntos de venta en grupos homogéneos. Para la predicción de series temporales, se emplean modelos clásicos cómo la regresión lineal y avanzados entre los que destacan las redes neuronales LSTM, ARIMA, Holt-Winters. Cada uno de estos últimos entrena con los datos preprocesados y se ajustan los hiperparámetros para optimizar su rendimiento.

Fase 5: Evaluación

Para cumplir con el modelado de Clustering se emplea el coeficiente de Silhouette (silueta) evalúa qué tan bien se agrupan los puntos de datos en sus respectivos clústeres, considerando tanto la cohesión interna como la separación entre clústeres.

Cálculo del Coeficiente de Silhouette

Para cada punto de datos ji, el coeficiente de silueta s(ji) se calcula utilizando dos valores:

- A(i): La distancia media entre el punto ji y todos los demás puntos en el mismo clúster.
- B(i): La distancia media entre el punto ji y todos los puntos en el clúster más cercano al cual ji no pertenece.

$$s(i) = \frac{B(i) - A(i)}{\max(A(i), B(i)}$$

Interpretación del Coeficiente de Silueta

El coeficiente de silueta promedio para un conjunto de datos proporciona una medida general de la calidad del clustering:

- **Coeficiente alto (cercano a 1):** Significa que la mayoría de los puntos están bien agrupados y los clústeres son claramente distinguibles.
- **Coeficiente bajo (cercano a 0):** Indica que los clústeres pueden no estar bien definidos y que los puntos están en los límites de los clústeres.
- **Coeficiente negativo:** Sugiere que muchos puntos están mal agrupados, indicando un mal ajuste del modelo de clustering.

Error Cuadrático Medio (RMSE) y el Error Absoluto Medio (MAE)

Una vez que los modelos están desarrollados, se procede a su evaluación. Esta fase implica la validación de la precisión y robustez de los modelos utilizando métricas como el error cuadrático medio (RMSE) y el error absoluto medio (MAE). Los resultados se comparan con modelos tradicionales para evaluar las mejoras obtenidas.

2.4.1 Aplicación de la teoría y conceptos

La metodología descrita se basa en principios fundamentales de la minería de datos, el aprendizaje automático y el análisis de series temporales. La aplicación de técnicas como el clustering y las redes neuronales LSTM se fundamenta en su

capacidad para manejar datos complejos y mejorar la precisión de los pronósticos de ventas. A continuación, se describen brevemente los conceptos teóricos subyacentes:

2.4.1.1 Técnicas de Machine Learning

El machine learning, o aprendizaje automático, es una rama de la inteligencia artificial que permite a los sistemas aprender y mejorar a partir de la experiencia sin ser explícitamente programados. Entre sus principales usos se encuentra el de analizar grandes volúmenes de datos, identificar patrones complejos y hacer predicciones precisas sobre ventas futuras. A continuación, se describen varias técnicas de machine learning que son relevantes para este propósito.

2.4.1.2 Aprendizaje Supervisado

El aprendizaje supervisado implica el entrenamiento de un modelo en un conjunto de datos etiquetados, donde el algoritmo aprende a mapear entradas a salidas deseadas. Ejemplos incluyen la regresión lineal, regresión logística y máquinas de soporte vectorial (SVM).

2.4.1.3 Aprendizaje No Supervisado

El aprendizaje no supervisado involucra el entrenamiento de un modelo en datos no etiquetados, identificando patrones o estructuras ocultas. Técnicas comunes incluyen el clustering y la reducción de dimensionalidad. Es útil para segmentar puntos de venta en clústers homogéneos, lo que facilita la personalización de estrategias y modelos de predicción. (Aggarwal & Reddy, 2016)

2.4.1.4 Aprendizaje Semi-Supervisado y Auto-Supervisado

Estos enfoques combinan elementos de aprendizaje supervisado y no supervisado, utilizando tanto datos etiquetados como no etiquetados para mejorar la precisión del modelo. Es especialmente útil en contextos donde los datos etiquetados son limitados o costosos de obtener. (Van Engelen & Hoos, 2019)

2.4.1.5 Clustering

El clustering es una técnica de aprendizaje no supervisado que agrupa datos en clústers basados en similitudes. Se basa en la identificación de patrones o estructuras en un conjunto de datos sin supervisión previa. A diferencia de los métodos supervisados, no requiere etiquetas predefinidas para el entrenamiento. Los algoritmos de clustering buscan particionar los datos en subconjuntos (clústers) de manera que los objetos dentro de un mismo clúster sean más similares entre sí que con los de otros clústers (Berkhin, 2020). A continuación. Se describen los algoritmos más comunes de clustering.

2.4.1.5.1 K-means

El método de clusterización k-means, llamado así donde “k” representa al número de clúster en los que se va a particionar al conjunto de datos y el término “means” (media) hace alusión a los centroides de los “k” clústers. Esta técnica es una de las más populares, además de ser las primeras en usarse para dicha finalidad.

K-means generalmente produce resultados eficientes cuando los grupos tienen una estructura compacta y convexa. Su objetivo es minimizar la suma de las distancias

cuadradas entre cada punto de datos y el centroide de su clúster asignado a lo largo de las iteraciones.

2.4.1.5.2 Hierarchical Clustering

Estos métodos jerárquicos se enfocan en agrupar clústeres para formar uno nuevo o separar uno existente para crear otros dos, de manera que, al realizar sucesivamente este proceso de aglomeración o división, se minimice alguna distancia o se maximice alguna medida de similitud. Se dividen en aglomerativos (ascendentes) y disociativos (descendentes). (Gallardo, 2009)

Una de las principales ventajas de estos algoritmos es que no requieren a priori el especificar un número de clústeres. A diferencia de otros métodos, lo que permite explorar y decidir sobre la estructura óptima a lo largo del análisis. En cuanto a su principal desventaja es la escalabilidad, en sentido de alta complejidad computacional. Además de ser sensibles al ruido y a los valores atípicos.

2.4.1.5.3 DBSCAN (Density-Based Spatial Clustering of Applications with Noise)

Density-Based Spatial Clustering of Applications with Noise más conocido por sus siglas “DBSCAN”, es un algoritmo de clustering que se fundamenta en la densidad y, a diferencia de K-means, no requiere definir previamente el número de clústeres que se formarán. Es especialmente útil para identificar valores atípicos y para distinguir clústeres de baja densidad en contraste con aquellos de alta densidad dentro de un mismo conjunto de datos.

El concepto de "densidad", hace referencia a la concentración de puntos de datos en una región del espacio de características. Un clúster denso es una agrupación de puntos de datos que están muy cercanos entre sí. Estos algoritmos basados en “densidad” identifican clústeres como regiones de alta densidad separadas por regiones de baja densidad (también conocidos como ruido o valores atípicos).

2.4.1.5.4 Gaussian Mixture Models (GMM)

Asumen que los datos se distribuyen según una combinación de varias distribuciones gaussianas. Utilizan un enfoque probabilístico para asignar puntos de datos a clústers, permitiendo una mayor flexibilidad en la forma de los clústers. Esta técnica es particularmente útil cuando los clústers tienen formas elípticas o tamaños variables. (Reynolds, 2020)

2.4.1.5.5 Spectral Clustering

Utiliza técnicas de álgebra lineal para reducir la dimensionalidad de los datos antes de aplicar un algoritmo de clustering como K-means. Es eficaz para encontrar clústers no lineales en datos de alta dimensionalidad. Requiere la construcción de una matriz de afinidad que capture las similitudes entre puntos de datos. Este método es útil para identificar estructuras complejas en datos y es especialmente potente cuando los clústers no tienen formas regulares. (Luxburg, 2018)

2.4.1.6 Modelos Clásicos de Predicción de Series Temporales

Los modelos clásicos de predicción de series temporales han sido ampliamente utilizados en el análisis y pronóstico de datos temporales debido a su

simplicidad y efectividad en ciertas condiciones. A continuación, se describen algunos de los modelos más comunes:

2.4.1.6.1 ARIMA (Autoregressive Integrated Moving Average)

ARIMA es un modelo estadístico que combina autoregresión (AR), integración (I) y media móvil (MA) para analizar y predecir datos de series temporales. Es útil para datos univariados con tendencias lineales y patrones estacionales, aunque puede ser limitado al asumir relaciones lineales entre las observaciones (Hyndman & Athanasopoulos, 2018).

2.4.1.6.2 SARIMA (Seasonal ARIMA)

Es una extensión de ARIMA que incorpora componentes estacionales en el modelo, adecuado para datos con patrones estacionales regulares. SARIMA agrega parámetros para capturar la estacionalidad, mejorando la precisión en contextos con ciclos repetitivo (Box, Jenkins, & Gregory, 2015).

2.4.1.6.3 Holt-Winters Exponential Smoothing

Un método de suavizamiento exponencial que incluye términos para la tendencia y la estacionalidad. Holt-Winters es efectivo para datos con patrones estacionales y tendencias lineales o exponenciales, utilizando componentes aditivos o multiplicativos según el tipo de datos. Este método es ampliamente utilizado para pronósticos a corto y mediano plazo en series temporales que exhiben una estructura estacional clara (Hyndman & Athanasopoulos, 2018).

2.4.1.7 Redes Neuronales LSTM

Las redes neuronales LSTM (Long Short-Term Memory) son un tipo de red neuronal recurrente (RNN) diseñadas para aprender y recordar patrones a largo plazo en datos secuenciales. Estas redes son especialmente efectivas para la predicción de series temporales debido a su capacidad para gestionar dependencias temporales y relaciones no lineales.

Las LSTM fueron introducidas para resolver el problema de los gradientes que se desvanecen, común en las RNN tradicionales. Las unidades LSTM incorporan una estructura de memoria especial que permite almacenar y recuperar información durante largos períodos de tiempo. Esta estructura se compone de celdas de memoria, puertas de entrada, salida y olvido, que regulan el flujo de información y permiten a la red mantener y actualizar la memoria de manera eficiente (Brownlee, Deep Learning for Time Series Forecasting, 2018).

2.4.1.7.1 Estructura de las LSTM:

- Celdas de Memoria: Actúan como un acumulador que preserva información durante múltiples pasos de tiempo. Se actualizan en cada paso de tiempo mediante las operaciones de las puertas de entrada, salida y olvido.
- Puerta de Entrada: Controla cuánto de la nueva información debe ser almacenada en la celda de memoria. Utiliza una función sigmoide para decidir qué partes de la entrada actual se agregan a la memoria.

- Puerta de Olvido: Determina qué información de la celda de memoria anterior debe ser descartada. Utiliza una función sigmoide para decidir qué información debe ser eliminada.
- Puerta de Salida: Controla la salida de la celda de memoria, determinando qué parte de la información almacenada debe ser utilizada en la siguiente predicción.

2.5 Aplicaciones Prácticas.

El uso de técnicas avanzadas de aprendizaje automático y clustering en la predicción de ventas ha mostrado resultados prometedores en estudios recientes. Por ejemplo, un estudio publicado en el International Journal of Forcasting demostró que la integración de técnicas de series temporales con algoritmos de clustering puede mejorar la precisión de los pronósticos de demanda en el sector minorista. Estos hallazgos son respaldados por investigaciones adicionales que sugieren que la personalización de los modelos de predicción mediante el clustering puede reducir los errores de pronóstico y optimizar la gestión de inventarios (Babai & Nikolopoulos, 2013).

Además, empresas del sector retail farmacéutico han comenzado a adoptar estas técnicas con éxito. Por ejemplo, IQVIA ha reportado mejoras significativas en sus capacidades de predicción de ventas mediante el uso de modelos de aprendizaje automático avanzados, lo que ha resultado en una mejor planificación de inventarios y una reducción de costos operativos (IQVIA, 2023).

2.6 Estudios de Caso y Análisis Comparativo

Para ilustrar la aplicación práctica de estas técnicas, se pueden considerar estudios de caso específicos en el sector retail farmacéutico. Por ejemplo, un estudio publicado en el International Journal of Production Economics, analiza el uso de redes neuronales y técnicas de clustering en una cadena de farmacias en China, demostrando una mejora significativa en la precisión de los pronósticos de ventas y la eficiencia operativa. Este estudio comparó varios enfoques, incluyendo modelos tradicionales y avanzados, destacando las ventajas de las técnicas de aprendizaje automático. (Wang & Gunasekaran, 2019)

2.7 Resultados Esperados

La aplicación de la teoría y los conceptos mencionados se espera que resulte en modelos de predicción de ventas que sean más precisos y robustos que los métodos tradicionales. Estos modelos permitirán a la empresa optimizar su cadena de suministro, mejorar la gestión del inventario y tomar decisiones estratégicas más informadas, contribuyendo así a una mayor eficiencia operativa y una mejor satisfacción del cliente.

3 Marco metodológico

3.1 Materiales

Para llevar a cabo el presente proyecto, se utilizarán diversos materiales y recursos, tanto físicos como digitales, que son esenciales para la recolección, procesamiento y análisis de los datos necesarios. A continuación, se detallan los materiales requeridos:

3.1.1 Datos

3.1.1.1 Datos Históricos de Ventas:

Datos de ventas de 152 puntos de venta, con un periodo de información de 5 años.

3.1.1.2 Datos Geográficos:

- Coordenadas geográficas (latitud y longitud) de cada farmacia objeto de estudio, así como su parroquia y provincia a la que pertenecen.

3.1.1.3 Categorización ATC de los artículos.

- Datos relacionados con la clasificación de medicamentos según el sistema o la parte del cuerpo en la que ejercen su principal efecto terapéutico.

3.2 Metodología de Ciencia de Datos: CRISP-DM

La metodología de ciencia de datos CRISP-DM en sus distintas fases se enfocará en el desarrollo y entrenamiento de modelos de predicción de ventas utilizando

técnicas avanzadas de aprendizaje automático y series temporales. Los pasos clave incluyen:

1. **Comprensión del Negocio.**
2. **Comprensión de los Datos.**
3. **Preparación de los Datos.**
4. **Modelado.**
5. **Evaluación.**

3.2.1 Herramientas de Software

3.2.1.1 Herramientas de Preprocesamiento de Datos:

- Python: Lenguaje de programación para la limpieza y preprocesamiento de datos, utilizando bibliotecas como Pandas, NumPy y Scikit-learn.
- Jupyter Notebooks: Entorno interactivo para el desarrollo y documentación del análisis de datos.

3.2.1.2 Herramientas de Clustering:

- Scikit-learn: Biblioteca de aprendizaje automático en Python que proporciona implementaciones de algoritmos de clustering como K-means, clustering jerárquico y DBSCAN.

3.2.1.3 Herramientas de Modelado y Predicción:

- TensorFlow y Keras: Bibliotecas de aprendizaje profundo para el desarrollo y entrenamiento de modelos LSTM.
- Prophet: Herramienta de Facebook para la predicción de series temporales, que se utilizará como referencia y comparación con los modelos LSTM.

3.2.1.4 Herramientas de Visualización de Datos:

- Matplotlib y Seaborn: Bibliotecas de Python para la creación de gráficos y visualizaciones de datos.

3.2.1.5 Bases de Datos Relacionales:

- SQL: Se utilizará SQL para la gestión y consulta de datos almacenados en bases de datos relacionales, facilitando el acceso y la manipulación eficiente de grandes volúmenes de datos.

3.2.2 Infraestructura de Hardware

3.2.2.1 Computadoras y Servidores:

- Computadoras con capacidad suficiente para manejar grandes volúmenes de datos y ejecutar algoritmos de aprendizaje automático.
- Acceso a servidores en la nube de Google Cloud para el almacenamiento y procesamiento de datos de manera eficiente.

3.2.2.2 Almacenamiento de Datos:

- Bases de datos relacionales (MySQL, PostgreSQL) para el almacenamiento y gestión de los datos recolectados.
- Almacenamiento en la nube para facilitar el acceso y la colaboración en los datos.

4 Resultados

4.1 Comprensión del Negocio

La comprensión del negocio es una fase crucial en la metodología CRISP-DM (Cross-Industry Standard Process for Data Mining), cuyo objetivo principal es obtener un entendimiento profundo de los objetivos estratégicos y operacionales de la empresa. Esto implica identificar problemas y oportunidades de negocio y formular los objetivos del proyecto de análisis de datos de manera clara y específica.

La empresa corresponde a un grupo empresarial dedicado a la distribución y comercialización de productos farmacéuticos y artículos de primera necesidad. Sus tiendas de retail tienen presencia en todas las provincias del Ecuador, excepto Galápagos.

El enfoque del modelo de negocio se basa en la expansión constante a través de la apertura de nuevos puntos de venta cada año. Un aspecto esencial de este modelo es la promoción de franquicias, que ofrecen importantes oportunidades de emprendimiento, especialmente para mujeres cabeza de familia.

4.1.1 Misión y visión de la empresa.

La misión de la empresa comercializar, con pasión y compromiso, productos que crean experiencias de salud, bienestar y conveniencia para los clientes. Destaca su compromiso con la excelencia en el servicio al cliente que justifica su visión de liderar

en la comercialización de productos y servicios de salud y bienestar, reconocidos por una cultura de excelencia centrada en el cliente.

4.1.2 Objetivos Estratégicos

El primer paso en la comprensión del negocio es identificar y entender los objetivos estratégicos. La empresa de retail farmacéutico ecuatoriano se dedica a mejorar la calidad de vida de las familias, proporcionando acceso conveniente y asequible a productos farmacéuticos y de cuidado personal. Al alinear el proyecto de predicción de ventas con estos objetivos, se asegura que los resultados sean relevantes y valiosos para la empresa.

El pronóstico de ventas se alinea estrechamente con sus objetivos estratégicos, particularmente con el objetivo de mantener un mínimo porcentaje de stock out en todas las categorías de productos, tales como Cuidado y Belleza, Consumo, Especialidad, Farma, Infantil y OTC y Bienestar. Al implementar un sistema de predicción de ventas basado en la metodología CRISP-DM, el proyecto proporcionará pronósticos precisos que permitirán una correcta planeación en las compras. Esto asegurará que la empresa pueda anticiparse a la demanda y mantener niveles óptimos de inventario, evitando tanto el exceso como la escasez de productos.

Al optimizar la gestión de inventarios mediante predicciones precisas, el proyecto contribuirá significativamente a la mejora de la eficiencia operativa. Al mantener el stock adecuado, la empresa no solo evitará pérdidas de ventas debido a la falta de productos, sino que también reducirá costos asociados con el

almacenamiento excesivo. Por lo que el presente proyecto, no solo apoyará el cumplimiento del objetivo de minimizar el stock out, sino que también mejorará la satisfacción del cliente al asegurar la disponibilidad continua de productos esenciales.

4.1.3 Entrevistas y Reuniones con Stakeholders

Para una comprensión profunda de las necesidades y expectativas, se consideraron entrevistas y reuniones con los principales stakeholders, incluyendo, gerentes de línea, responsables de la cadena de suministro y personal de TI. Permitiendo recoger información sobre los desafíos actuales y las oportunidades de mejora en la predicción de ventas y gestión de inventarios.

Las oportunidades plasmadas en los objetivos estratégicos son vitales para asegurar el desempeño óptimo y su crecimiento en el mercado. Desde la perspectiva de la cadena de suministro, es crucial mantener un porcentaje mínimo de "stock out" tanto en la Bodega Central como en los Puntos de Venta. Esto se logra mediante la adecuada planificación y alineación de los niveles de inventario con la demanda real. La disponibilidad constante del inventario no solo garantiza el cumplimiento de las actividades comerciales dirigidas por Trade Marketing, sino que también mejora la satisfacción del cliente al asegurar que los productos necesarios estén siempre disponibles.

Es importante destacar que la demanda de artículos varía significativamente entre los diferentes puntos de venta. Estas variaciones están influenciadas por factores regionales, provinciales y cantonales, así como por la sucursal específica a la

que pertenece cada punto de venta. La Inteligencia de Negocios juega un papel fundamental en la identificación de estas oportunidades. Herramientas como los tableros de control permiten a la empresa comprender mejor las realidades del negocio y tomar decisiones informadas. Este enfoque facilita la adaptación a las dinámicas del mercado y optimiza las operaciones, alineando estrechamente la oferta con la demanda específica de cada ubicación.

4.1.4 Análisis de Documentación Interna

La empresa promulga diversos canales de comunicación como fuentes oficiales de información los que nos permite familiarizarnos con los objetivos estratégicos, políticas, procesos, entre otros documentos de suma importancia. Permitiendo identificar las áreas críticas donde los modelos predictivos pueden ofrecer el mayor beneficio y contribuir al cumplimiento de los objetivos estratégicos de la empresa.

4.1.5 Evaluación del Entorno Externo

Es fundamental entender el entorno externo en el que opera le empresa. La industria del retail farmacéutico en Ecuador enfrenta desafíos como la variabilidad de la demanda por factores estacionales, cambios en la regulación sanitaria y eventos como la pandemia de COVID-19. Estos factores externos influyen directamente a que los modelos sean precisos y también resilientes a las fluctuaciones del mercado.

4.1.6 Infraestructura Tecnológica

Se considera importante la infraestructura tecnológica, impulsados por la visión de la gerencia de Inteligencia de Negocios, la empresa cuenta con una infraestructura avanzada que soporta sus operaciones. Esta infraestructura tanto hardware como software, así como soluciones en la nube que garantizan la eficiencia y escalabilidad en sus procesos.

La combinación de una infraestructura tecnológica robusta con soluciones avanzadas en la nube permite mantener una operación eficiente, escalable y alineada con sus objetivos estratégicos, asegurando que puedan adaptarse rápidamente a las dinámicas del mercado y ofrecer un servicio de alta calidad a sus clientes.

4.1.7 Identificación de Requisitos de Datos

Un aspecto fundamental en la comprensión del negocio es determinar los requisitos de datos necesarios para crear modelos de predicción de ventas precisos y eficaces. Esto implica una evaluación minuciosa de la información disponible y necesaria. La empresa dispone de una amplia gama de datos estructurados, entre los que se incluyen:

Datos Históricos de Ventas: Información detallada sobre las transacciones de ventas realizadas en diversas farmacias, que permiten analizar patrones y tendencias históricas, cruciales para las predicciones futuras.

Información de Facturación: Registros de facturación que ofrecen una visión completa de los ingresos generados, desglosados por productos, clientes y períodos de tiempo específicos.

Datos de Farmacias: Información específica de cada punto de venta, incluyendo su ubicación geográfica, tamaño y características operativas, ayudando a comprender las particularidades de cada farmacia y su impacto en las ventas.

Datos de Productos: Información detallada sobre los productos vendidos, que incluye categorías y precios, esencial para analizar el desempeño de distintos productos.

La calidad de los datos es evaluada constantemente por técnicos especializados, así como también asegurando su disponibilidad. Gracias a ello podemos asegurar de que los modelos de predicción de ventas sean robustos y fiables.

4.1.8 Cadena de valor.

La cadena de valor se encuentra en los sectores farmacéutico y de productos de consumo, con un enfoque principal en la distribución y comercialización de estos bienes.

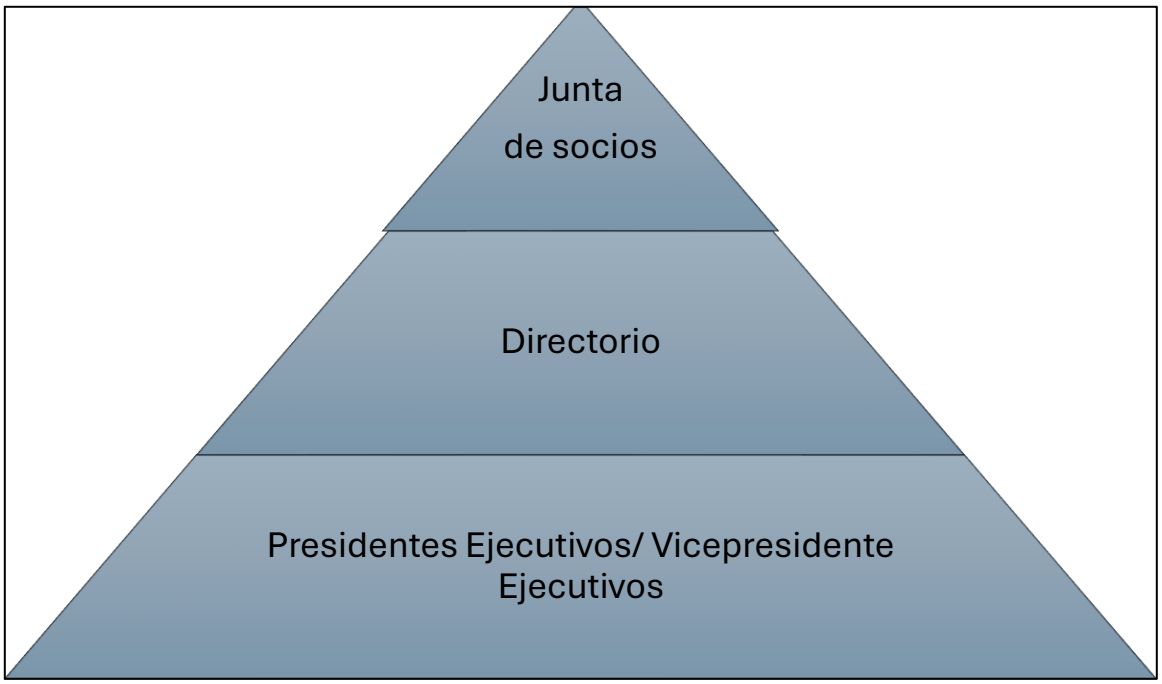
Esta cadena de valor se estructura en tres etapas esenciales. La primera etapa es la compra, donde se adquieren productos farmacéuticos y de consumo provenientes de diversos proveedores. La segunda corresponde al almacenamiento, que implica la gestión de inventarios en sus instalaciones, asegurando que los productos se conserven adecuadamente y estén disponibles cuando se necesiten. Finalmente, la tercera etapa es la distribución y venta, que consiste en la entrega de

productos a nivel nacional, abarcando una amplia gama de clientes, incluidos pequeñas cadenas de farmacias, distribuidoras farmacéuticas, hospitales y casas de salud.

4.1.9 Estructura Organizacional.

La empresa cuenta con una estructura organizativa representada en la ilustración 1. La presente se encuentra diseñada para asegurar un sólido gobierno corporativo y la toma de decisiones estratégicas de manera eficiente. La autoridad suprema en la empresa recae en la Junta General de Socios, seguida por un Directorio. Este Directorio, compuesto por cinco miembros, es elegido por la Junta General de ocios para desempeñar funciones de liderazgo y supervisión dentro de la organización.

Ilustración 1 Estructura Organizacional de la empresa, Elaborado por: Autor.



4.1.10 Organigrama Empresarial

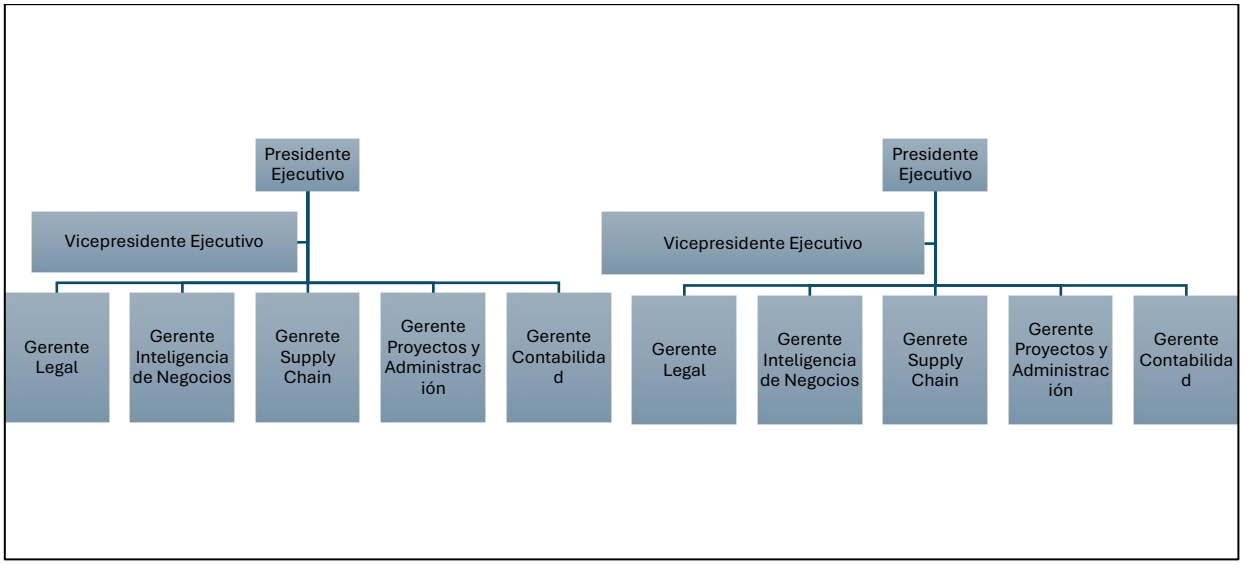
La estructura organizacional se representa en la ilustración 2. En la parte superior representada por dos presidentes ejecutivos, cada uno encargado de supervisar diferentes áreas funcionales de la organización. Bajo la supervisión del primer presidente ejecutivo, se encuentran:

- Gerencia Legal.
- Gerencia de Inteligencia de Negocios.
- Gerencia de Supply Chain.
- Gerencia de Auditoría Interna.
- Gerencia de Proyectos y Administración.
- Gerencia de Contabilidad.

Por su parte el segundo presidente ejecutivo se encuentra liderando las siguientes gerencias:

- Gerencia Legal.
- Gerencia de Auditoría Interna.
- Gerencia de Administración de Inventarios.
- Gerencia de Compras.

Ilustración 2 Organigrama. Elaborado por: Autor



4.1.11 Objetivos de Minería de Datos

- Desarrollar un modelo predictivo que pronostique las ventas mensuales totales por farmacia con un margen de error inferior al 10%.

Métricas de Evaluación:

- Error Cuadrático Medio (MSE).
- Error Absoluto Medio (MAE).
- Precisión del modelo.

4.1.12 Generación del Plan de Proyecto

Plan de Proyecto Detallado:

- **Fase 1: Recolección y Preparación de Datos:** Recopilar datos de ventas históricas; limpiar y preparar los datos.

- **Fase 2: Análisis Exploratorio de Datos:** Analizar tendencias y patrones en las ventas totales.
- **Fase 3: Análisis de Clústers:** Aplicar técnicas de clustering para segmentar los puntos de venta en grupos homogéneos. Utilizar algoritmos como K-means, DBSCAN o modelos de mezcla gaussiana para identificar similitudes y diferencias entre las tiendas. Validar y ajustar los clústers según sea necesario.
- **Fase 4: Desarrollo de Modelos Predictivos:** Desarrollar y probar diferentes modelos predictivos.
- **Fase 5: Evaluación de Modelos:** Evaluar el rendimiento de los modelos.
- **Fase 6: Implementación y Monitoreo:** Implementar el modelo y monitorear su rendimiento.

Roles y Responsabilidades:

- **Analista de Inteligencia de Negocios**
    - Desarrollo y evaluación de modelos.
    - Recolección y preparación de datos.
    - Traducción de objetivos de negocio a minería de datos.
    - Evaluación de resultados.

Esta estrategia asegurará que la empresa pueda utilizar los pronósticos de ventas para optimizar la gestión del inventario y la planificación de recursos, mejorando así su eficiencia operativa y la satisfacción del cliente.

4.2 Comprensión de los Datos

El objetivo principal de esta fase es obtener una comprensión profunda de los datos disponibles, su calidad y su estructura. Esta comprensión es fundamental para el desarrollo de modelos predictivos precisos y efectivos, que puedan capturar las complejidades del mercado y proporcionar pronósticos fiables para la empresa.

4.2.1 Descripción de los Datos

En el contexto del proyecto de predicción de ventas, se utilizarán varios tipos de datos que son cruciales para el desarrollo de modelos precisos y efectivos. Los datos se dividen en las siguientes categorías:

1. **Datos Históricos de Ventas:** Incluyen registros detallados de ventas de 152 farmacias, abarcando un periodo mínimo de 60 meses. Estos datos proporcionan información sobre el comportamiento de las ventas a lo largo del tiempo, lo que es esencial para identificar tendencias y patrones estacionales.
2. **Datos Geográficos:** Se recopilarán datos de coordenadas geográficas (latitud y longitud) de cada una de las farmacias, así como su provincia y cantón al que pertenecen.

4.2.2 Proceso de Recolección y Preparación de Datos

Recolección de Datos, mediante la obtención de información histórica de ventas a partir de datos estructurados ya almacenados. Para el presente proyecto, los datos se extraerán mediante el uso de lenguaje SQL de las tablas almacenadas en el Data Warehouse (DWH) de la empresa.

Data Warehouse

El Data Warehouse es una fuente centralizada que integra y almacena grandes volúmenes de datos, proporcionando un acceso eficiente y consolidado a la información necesaria. Las tablas específicas que se utilizarán incluyen:

- **dwh.farmacomercial:** Contiene datos detallados sobre las transacciones comerciales, permitiendo un análisis profundo de las ventas realizadas.
- **dwh.farmafarmacia:** Almacena información específica de cada punto de venta, incluyendo su ubicación y características operativas.
- **dwh.farmaproductos:** Proporciona detalles sobre los productos, incluyendo categorías comerciales, tipo de molécula, forma farmacéutica, tipo de artículo, precio, fecha de creación, estado del producto en Bodega Central, entre otras.

4.2.3 Exploración de Datos

La fase de exploración de datos es esencial en el proceso de análisis, ya que proporciona una comprensión profunda de las características y patrones presentes en

los datos recopilados. A continuación, se describen las principales características de las tablas alojadas en el repositorio del Data Warehouse.

4.2.3.1 Descripción de las Tablas del Data Warehouse:

1. **Tabla dwh.farmacomercial:**

   - **Contenido:** Para el presente proyecto se emplean las columnas referentes al código del punto de venta (bodega), valor venta de la transacción (valorventa), año y mes de la transacción (aniomes).
   - **Uso:** Esencial para el pronóstico de ventas (series de tiempo).
   - **Registros:** Se visualiza en la ilustración 3, al realizar el conteo de registros en la farmacomercial, esta almacena 803 millones de registros al 31 de julio 2024. Su primera factura almacenada corresponde al 01 de enero del 2015, se muestra en la ilustración 4.

Ilustración 3 Registros de la tabla Farmacomercial

```
In [2]:
   ...:
   ...: registros_fc = run_query("""SELECT COUNT(*)
   ...:                             FROM dwh.farmacomercial
   ...:                            """)
   ...:
   ...: registros_fc
Out[2]:
    count(*)
0  803947998
```

Ilustración 4 Temporalidad tabla farmacomercial

```
In [3]: temporalidad_fc = run_query("""
   ...:                             SELECT min(fechafactura), max(fechafactura)
   ...:                             FROM dwh.farmacomercial
   ...:                             """)
   ...: temporalidad_fc
Out[3]:
    min(fechafactura)   max(fechafactura)
0 2015-01-01 00:04:00 2024-07-31 09:03:46
```

- **Valores nulos y blancos:** La tabla farmacomercial, en las columnas codfarmacia, valorventa y aniomes no dispone de valores nulos ni blancos como podemos observar en la ilustración 5.

Ilustración 5 Conteo del número de registros nulos y blancos de la tabla farmacomercial

```
In [8]:
   ...: nulos_blancos_fc = run_query("""
   ...:     SELECT
   ...:         SUM(CASE WHEN codfarmacia IS NULL OR TRIM(CAST(codfarmacia AS STRING)) = '' THEN 1 ELSE 0 END) AS null_blank_codfarmacia,
   ...:         SUM(CASE WHEN aniomes IS NULL OR TRIM(CAST(aniomes AS STRING)) = '' THEN 1 ELSE 0 END) AS null_blank_aniomes,
   ...:         SUM(CASE WHEN valorventa IS NULL OR TRIM(CAST(valorventa AS STRING)) = '' THEN 1 ELSE 0 END) AS null_blank_valorventa
   ...:     FROM dwh.farmacomercial;
   ...: """)
   ...:
   ...: # Mostrar los resultados
   ...: print(nulos_blancos_fc)
   null_blank_codfarmacia  null_blank_aniomes  null_blank_valorventa
0                       0                   0                      0
```

2. **Tabla dwh.farmafarmacia:**

- **Contenido:** Contiene información específica sobre cada farmacia (bodega), como su nombre (farmacia), ubicación (coordenadas de latitud y longitud), así como su provincia, cantón, tipo de sucursal, marca, fecha de inicio de sus operaciones, su estado (activo, inactivo), venta promedio, tipo (franquicia o propia de la empresa).

- **Uso:** Esta información es esencial para clisterizar a los puntos de venta y analizar el rendimiento de las ventas en diferentes cantones, provincias, regiones y características inherentes a la farmacia.

- **Registros:** La tabla farmafarmacia dispone de 1671 registros al 31 de julio 2024, el conteo lo podemos visualizar en la ilustración 6.

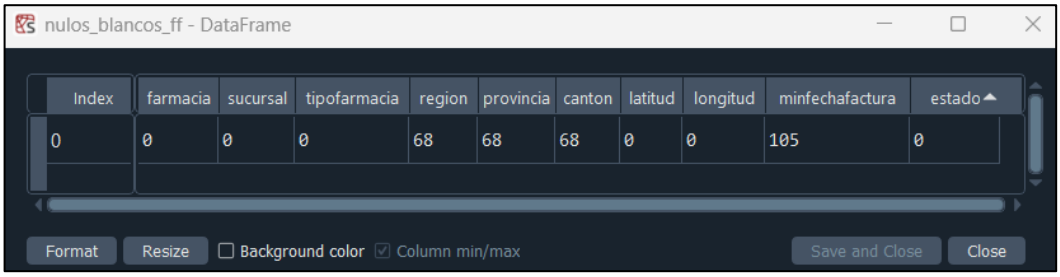
Ilustración 6 Conteo del número de registros tabla farmafarmacia

```
In [16]: registros_ff = run_query("""SELECT COUNT(*)
    ...:                                FROM dwh.farmafarmacia
    ...:                               """)
    ...:
    ...: registros_ff
Out[16]:
   count(*)
0      1671
```

- **Valores nulos y blancos:** La tabla farmafarmacia, evaluada en las columnas farmacia, sucursal, tipofarmacia, región, provincia, cantón, latitud, longitud, minfechafactura y estado. Presenta blancos, 68 registros de provincia, región y cantón. Además, en la variable de mínima fecha de factura (minfechafactura) con 105 registros en blanco.

  Estos valores blancos se presentan en puntos de venta inactivos o registros temporales hasta el inicio de sus operaciones, podemos visualizarlos en la ilustración 7.

Ilustración 7 Número de registros en blanco por variables de la tabla farmafarmacia.



3. **Tabla dwh.farmaproductos:**

   - **Contenido:** Incluye características sobre los productos, como el código del producto (codproducto), nombre (nombreproducto), tipo del artículo servicio o inventario (esvirtual), categorías comerciales (unegocioc) y especificaciones.

   - **Uso:** Permite el análisis de ventas segmentado por ciertas categorías comerciales del producto. Esto tiene la finalidad de enfocar el modelo hacia artículos farmacéuticos, se probará el omitir o no artículos virtuales (servicios).

   - **Registros:** La tabla farmaproductos dispone de 67326 registros de productos activos e inactivos, se incluyen registros inactivos debido al ciclo de vida de los productos. El conteo se visualiza en la ilustración 8.

Ilustración 3 Conteo de registros de la tabla farmaproductos

```
In [20]: registros_fp = run_query("""SELECT COUNT(*)
    ...:                             FROM dwh.farmaproductos
    ...:                            """)
    ...:
    ...: registros_fp
Out[20]:
   count(*)
0     67326
```

4.2.3.2 Estructura del DataFrame.

Para cumplir con el objetivo de caracterizar las tiendas de retail farmacéutico, se integraron ciertos campos utilizando consultas en lenguaje SQL (Anexo 1 Consultas en Lenguaje SQL). El dataframe resultante fusiona características de la farmacia y la venta en categoría ATC y la participación de estas en la venta. El DataFrame resultante, denominado "df_pdv_cluster_atc1" es descrito en la ilustración 9.

Ilustración 4 Tipos de variables "df_pdv_cluster_atc1"

```
Out[2]:
atc_1
codfarmacia                              object
farmacia                                 object
sucursal                                 object
tipofarmacia                             object
region                                   object
provincia                                object
canton                                   object
latitud                                 float64
longitud                                float64
minfechafactura                  datetime64[ns]
A - APARATO DIGEST.Y METABOL            float64
B - SANGRE Y ORGANOS HEMATOP            float64
C - APARATO CARDIOVASCULAR              float64
D - DERMATOLOGICOS                      float64
G - PROD.GENITO URINARIOS               float64
H - HORMONAS                            float64
J - ANTIINFECCIOSOS VIA GENE            float64
L - ANTINEOPLAS Y AGENT INMUN           float64
M - APARATO LOCOMOTOR                   float64
N - SISTEMA NERVIOSO                    float64
P - ANTIPARASITARIOS                    float64
R - APARATO RESPIRATORIO                float64
S - ORGANOS DE LOS SENTIDOS             float64
meses_desde_apertura                      int64
total_venta_ATC                         float64
A - APARATO DIGEST.Y METABOL_Part       float64
B - SANGRE Y ORGANOS HEMATOP_Part       float64
C - APARATO CARDIOVASCULAR_Part         float64
D - DERMATOLOGICOS_Part                 float64
G - PROD.GENITO URINARIOS_Part          float64
H - HORMONAS_Part                       float64
J - ANTIINFECCIOSOS VIA GENE_Part       float64
L - ANTINEOPLAS Y AGENT INMUN_Part      float64
M - APARATO LOCOMOTOR_Part              float64
N - SISTEMA NERVIOSO_Part               float64
P - ANTIPARASITARIOS_Part               float64
R - APARATO RESPIRATORIO_Part           float64
S - ORGANOS DE LOS SENTIDOS_Part        float64
dtype: object
```

Una vez caracterizada la farmacia, el siguiente paso consiste en evaluar modelos de pronóstico de ventas, se construye el DataFrame integrado por consultas en lenguaje SQL de tabas de registros de ventas mensuales, características del punto de venta y el clúster resultante. El DataFrame "df" contiene las siguientes variables y su tipo al que pertenece descrito en la ilustración 10.

Ilustración 10 DataFrame "df", tipo de columnas.

```
In [85]: df.dtypes
Out[85]:
codfarmacia              object
farmacia                 object
sucursal                 object
tipofarmacia             object
region                   object
provincia                object
canton                   object
latitud                 float64
longitud                float64
aniomes                   int64
cluster_kmeans            int64
meses_desde_apertura      int64
valorventa              float64
dtype: object
```

4.2.4 Calidad de los Datos

La calidad de los datos es un aspecto crucial que influye directamente en la precisión y efectividad de los modelos predictivos. Para garantizar la calidad de los datos, se seguirán los siguientes pasos:

Manejo de Valores Faltantes: Se identificarán y tratarán los valores faltantes mediante técnicas como la imputación de datos, que permite rellenar las ausencias basándose en valores similares o patrones identificados en los datos.

Eliminación de Datos Atípicos: Los datos atípicos, o valores extremos, serán identificados y eliminados o ajustados para evitar que distorsionen los resultados de los modelos.

Normalización de Datos: La normalización de datos implica ajustar los valores a una escala común, lo que facilita la comparación y análisis de los datos. Esto es especialmente importante cuando se trabaja con datos provenientes de diferentes fuentes con distintas unidades de medida.

4.3 Preparación de los Datos

Una vez desarrolladas las fases de entendimiento del negocio y de los datos. Continuamos con la tercera fase del modelado CRISP-DM, que establece la preparación de la data, desde la selección, limpieza, construcción, integración y formateo de los datos.

4.3.1 Preparación de los Datos Modelado Clúster

El primer DataFrame se estructura con la finalidad de caracterizar por patrones de venta de medicamentos bajo la clasificación ATC (Anatomical Therapeutic Classification System), en su primer nivel. Básicamente contempla la clasificación de medicamentos y productos farmacéuticos según el órgano o sistema en el que actúan y sus propiedades terapéuticas, farmacológicas y químicas. El primer nivel (ATC1), representa al grupo anatómico principal.

4.3.1.1 Selección de los Datos Modelado Clúster

En esta fase, se seleccionan los datos relevantes para la caracterización de los puntos de venta. Se decide qué datos son necesarios y se extraen de las fuentes correspondientes.

Se generan consulta en lenguaje SQL donde se seleccionan los datos relacionados con farmacias como el nombre, código, sucursal, tipo de farmacia, región, provincia, cantón, latitud, longitud, fecha mínima de factura, código del producto, nombre de producto, así como su ATC1 al que pertenece, se promedian las ventas de un período específico (enero a junio de 2024), limitando la selección a farmacias de la cadena que se encuentren activas.

4.3.1.2 Limpieza de los Datos para Modelado Clúster.

Cómo lo describimos anteriormente, la tabla “farmacomercial” la cuál almacena los registros de ventas no dispone de nulos ni blancos. Sin embargo, no era el caso de “farmafarmacia” referente a la información del punto de venta, al analizarla se presentaban ciertos registros blancos. Al filtrar la tabla por registros activos estos se omiten, por lo que se concluye que son farmacias inactivas o de próxima apertura.

Para asegurar que los datos sean adecuados para el análisis se omiten farmacias no relevantes, es el caso de la farmacia Móvil, la cual se usa para Branding de la marca. Se filtran las farmacias cuya fecha de apertura sea anterior a junio 2023 para garantizar que haya al menos 12 meses de datos disponibles de información.

4.3.1.3 Construir los Datos para Modelado Clúster

En esta etapa, se crean nuevas variables o transformaciones de datos que pueden mejorar el análisis o el rendimiento del modelo.

Cálculo de Meses desde la Apertura: Se añade una nueva variable que calcula los meses transcurridos desde la apertura de cada farmacia hasta una fecha de corte (junio de 2024).

Cálculo del Total de Ventas por Farmacia: Se calcula la suma total de las ventas para cada farmacia en todas las categorías ATC1, que es la clasificación farmacológica de los productos.

Cálculo del Porcentaje de Venta por Categoría: Se calcula el porcentaje de ventas de cada categoría ATC1 con respecto al total de ventas de la farmacia.

4.3.1.4 Integración de los Datos Modelado Clúster

Se combinan los datos de diferentes fuentes en un solo DataFrame que será utilizado para el análisis.

Unión de Datos con Información de Producto ATC1: Se realiza una unión (merge) entre los datos de ventas y un archivo externo que contiene la clasificación ATC1 de los productos.

Reestructuración de Datos con Pivot Table: Los datos se reorganizan en una tabla pivote donde las categorías ATC1 se convierten en columnas y los valores corresponden a la suma de ventas promedio.

4.3.1.5 Formateo de los Datos Modelado Clúster

El formateo de datos es un proceso crucial dentro de la etapa de preparación de datos, en el cual se transforman, estructuran y organizan los datos de manera que cumplan con los requisitos específicos de los modelos analíticos que se van a aplicar. Esta fase asegura que los datos estén en un formato adecuado, lo que es fundamental para la precisión y eficacia de cualquier análisis o modelo predictivo.

Fechas y Horas: Las columnas que contienen información de fechas han sido convertidas a objetos de tipo datetime. Esto es esencial para realizar cálculos y operaciones temporales con precisión, como la comparación de fechas, la generación de secuencias temporales, o la agregación de datos en intervalos específicos.

Códigos de Identificación: Los códigos asociados a productos y puntos de venta, como codproducto y codfarmacia, han sido formateados como cadenas de texto (str). Este formateo es importante porque asegura que estos códigos se traten como identificadores únicos, evitando su interpretación errónea como valores numéricos, lo que podría alterar el análisis, especialmente en operaciones como agrupaciones o uniones de datos basadas en estos identificadores.

4.3.2 Preparación de los Datos Modelado Pronósticos

Una vez caracterizados los puntos de venta se integran a un segundo data set en el que se dispone de la información histórica de ventas asociada a cada tienda de retail, ATC1 correspondiente al y clúster de pertenencia del punto de venta.

4.3.2.1 Selección de los Datos Modelado Pronósticos

El primer paso en la preparación de los datos fue seleccionar las variables y registros pertinentes para el análisis. Se utilizaron datos de ventas de productos farmacéuticos, clasificados de acuerdo con el sistema ATC1 (Anatomical Therapeutic Chemical Classification System), que agrupa los medicamentos según el órgano o sistema sobre el que actúan, así como sus propiedades terapéuticas. La extracción de datos se realizó desde un Data Warehouse utilizando consultas SQL.

Para garantizar la pertinencia de la información, se seleccionaron exclusivamente los datos relacionados con farmacias que estuvieron operativas entre enero de 2019 y mayo de 2024. Además, se agruparon previamente los productos según su clasificación ATC1, permitiendo realizar consultas segmentadas para cada grupo de productos farmacéuticos. Limpieza de los datos para modelado LTSM.

4.3.2.2 Limpieza de los Datos Modelado Pronósticos

Con los datos seleccionados, el siguiente paso fue su depuración. Esta etapa incluyó la eliminación de registros no necesarios y la conversión de formatos para asegurar la coherencia y la compatibilidad de los datos con las fases posteriores del análisis.

Se filtraron los datos para que solo se incluyeran los puntos de venta que formaban parte del análisis de clústeres realizado previamente. Para ello, se comparó la lista de códigos de farmacia (codfarmacia) con los códigos de la tabla de clústeres. Asimismo, se verificó que las fechas estuvieran correctamente formateadas como

objetos datetime, lo cual facilitó la realización de cálculos temporales y la medición de diferencias de tiempo.

4.3.2.3 Construir los Datos Modelado Pronósticos

En este punto, se crearon nuevas variables a partir de los datos existentes con el objetivo de enriquecer el análisis. En particular, se desarrolló una variable que mide los meses transcurridos desde la apertura de cada farmacia hasta el mes correspondiente a cada registro de ventas (aniomes). Esta variable es fundamental para analizar cómo la antigüedad de las farmacias influye en sus patrones de ventas.

El cálculo de esta variable se llevó a cabo combinando las fechas de apertura con las fechas correspondientes a cada mes de ventas, lo que permitió medir con precisión el tiempo que cada farmacia ha estado en operación.

4.3.2.4 Integración de los Datos Modelado Pronósticos

La última etapa de la preparación de datos fue la integración, donde se combinaron distintos conjuntos de datos en un único DataFrame, listo para su análisis en modelos predictivos.

Se integraron los datos de ventas con la información sobre clústeres obtenida previamente, añadiendo una columna que identifica el clúster al que pertenece cada farmacia. Posteriormente, los datos fueron agrupados en función de diversas variables clave, como codfarmacia, aniomes y cluster_kmeans, y se sumaron las ventas correspondientes a cada combinación de estas variables. Este proceso permitió

estructurar un conjunto de datos ordenado y organizado, adecuado para su uso en la modelización predictiva.

Finalmente, el DataFrame resultante fue ordenado por aniomes y codfarmacia, permitiendo así una secuencia temporal coherente y preparando los datos para su análisis y la generación de pronósticos.

4.4 Modelado

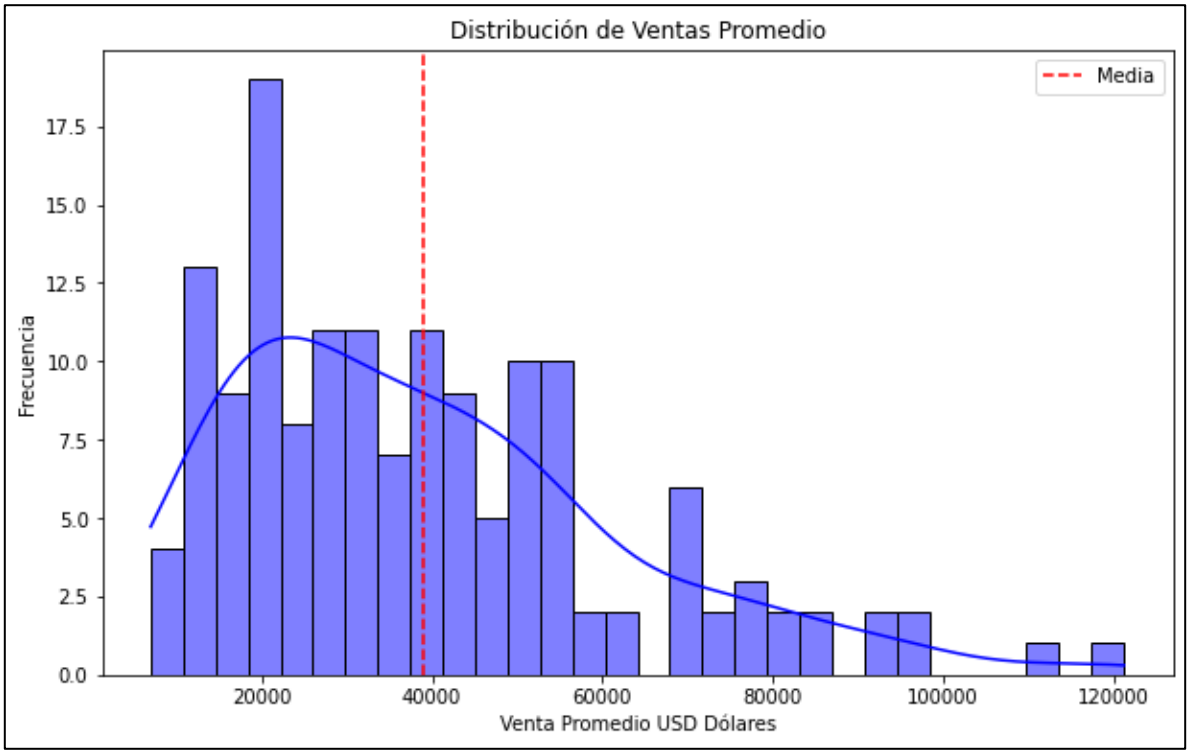
4.4.1 Análisis Estadístico y Visualización de Datos:

Una vez definidas las fuentes de datos, se procede a aplicar técnicas estadísticas para analizar y cuantificar las relaciones y dependencias entre las variables. Para comenzar el análisis del DataFrame “df_pdv_atc1”, compuesto por la información de características del punto de venta (farmafarmacia), registros de venta (farmacomercial), tipo de producto (farmaproductos) creado mediante consultas SQL presentes en el Anexo 1. Se utilizarán técnicas estadísticas y métodos de visualización de datos para entender mejor las características y patrones presentes en los datos. A continuación, se presenta una serie de pasos para realizar este análisis:

Distribución de Ventas Promedio

La visualización de la distribución de la variable “ventapromedio” nos ayuda a entender cómo se distribuyen las ventas entre las diferentes farmacias. Utilizando un histograma (Ilustración 11), se observar la dispersión y la tendencia central de las ventas.

Ilustración 5 Venta promedio mensual por punto de venta.



La ilustración 11 nos permite visualizar cómo se encuentran distribuidas las ventas promedio de las tiendas de retail, la distribución presenta un sesgo positivo, es decir, está inclinada hacia la derecha. Esto implica que la mayoría de las farmacias tienen ventas promedio relativamente bajas, mientras que un menor número de farmacias reporta ventas significativamente más altas. Esta característica sugiere que, aunque muchas farmacias generan ingresos moderados, existen algunas que logran destacarse con ventas considerablemente mayores.

La mayor concentración de ventas promedio se encuentra en el rango de USD 10,000 a USD 60,000. Este intervalo parece ser el más común en el sector, lo que indica que la mayoría de las farmacias de este tipo posicionarse dentro de este margen. Este

rango refleja la realidad de muchas tiendas que operan en un entorno competitivo y manejan volúmenes de ventas consistentes, pero no exorbitantes.

Además, una cola larga hacia la derecha, lo que sugiere la presencia de un número reducido de farmacias con ventas promedio mucho más elevadas, alcanzando cifras cercanas a USD 120,000. Este fenómeno puede estar asociado a farmacias ubicadas en zonas de alta demanda o aquellas que han logrado captar un segmento de mercado más amplio o especializado.

El valor promedio de las ventas, representado por una línea discontinua roja en el gráfico, se sitúa cerca de los USD 40,000. Este promedio es un indicador útil para comparar el rendimiento de una farmacia individual con la tendencia general del sector. Las estadísticas descriptivas para la columna “ventapromedio” en el DataFrame “df_pdv_atc1” son las siguientes:

- Cantidad de Observaciones: Se analizaron un total de 152 farmacias.
- Media: La venta promedio por farmacia es de aproximadamente USD 39,081.80
- Desviación Estándar (std): La desviación estándar de USD 22,948.46 indica una alta dispersión en las ventas.
- Valor Mínimo (min): La farmacia con la venta más baja reportó ventas de aproximadamente USD 6,987.91.
- Primer Cuartil (25%): El 25% de las farmacias reportaron ventas inferiores a USD 20,320.59. Una cuarta parte de las farmacias tiene ventas relativamente bajas.

- Mediana (50%): La mediana de las ventas es de USD 33,912.97, el 50% de las farmacias tiene ventas por debajo de este valor. Es inferior a la media, lo que indica una distribución sesgada hacia la derecha.
- Tercer Cuartil (75%): El 75% de las farmacias reportaron ventas por debajo de USD 51,816.38, mientras que el 25% restante tiene ventas superiores a este valor.
- Valor Máximo (max): La farmacia con la mayor venta total alcanzó aproximadamente USD 121,207.23, lo que muestra una alta variabilidad en el rendimiento de las farmacias.
- Moda (mode): La moda, o el valor más frecuente, es USD 6,987.91, indicando que hay una farmacia con ventas considerablemente bajas en comparación con las demás.
- Varianza (variance): La varianza de 526,631,909.57 refleja una variabilidad significativa en las ventas totales, consistente con la alta desviación estándar.

La descripción del Data Frame “df_pdv_atc1” contribuye al análisis gráfico, las ventas promedio en las diferentes farmacias es de USD 69,334 con una desviación estándar de USD 33,171, lo que refleja cierta variabilidad. El rango de las ventas promedio va desde USD 32,631 hasta USD 112,634.

Ventas Promedio por ubicación geográfica.

En la ilustración 12 podemos notar la diferencia existente entre el promedio de ventas en los distintos cantones del Ecuador. Esta identificación nos ayuda a

identificar diferencias en las ventas dependiendo de la localización. Las observaciones se presentan a continuación.

Samborondón destaca como el cantón con la venta promedio más alta, alcanzando aproximadamente USD 80,000. Esta cifra sugiere un mercado con alto poder adquisitivo y una demanda significativa de productos farmacéuticos, probablemente impulsada por factores socioeconómicos favorables en esta área.

Ambato y Cayambe. Con ventas promedio alrededor de los USD 60,000 y USD 50,000, respectivamente. Estos cantones muestran una actividad dinámica en el mercado farmacéutico, lo que puede ser indicativo de un entorno comercial próspero y en expansión en estas zonas.

En una posición intermedia se encuentran los cantones de Ibarra, Quito, y Daule, donde las ventas promedio rondan los USD 40,000. Estos cantones representan mercados estables, con un tamaño considerable en la industria farmacéutica, reflejando una demanda sólida y consistente de productos de salud.

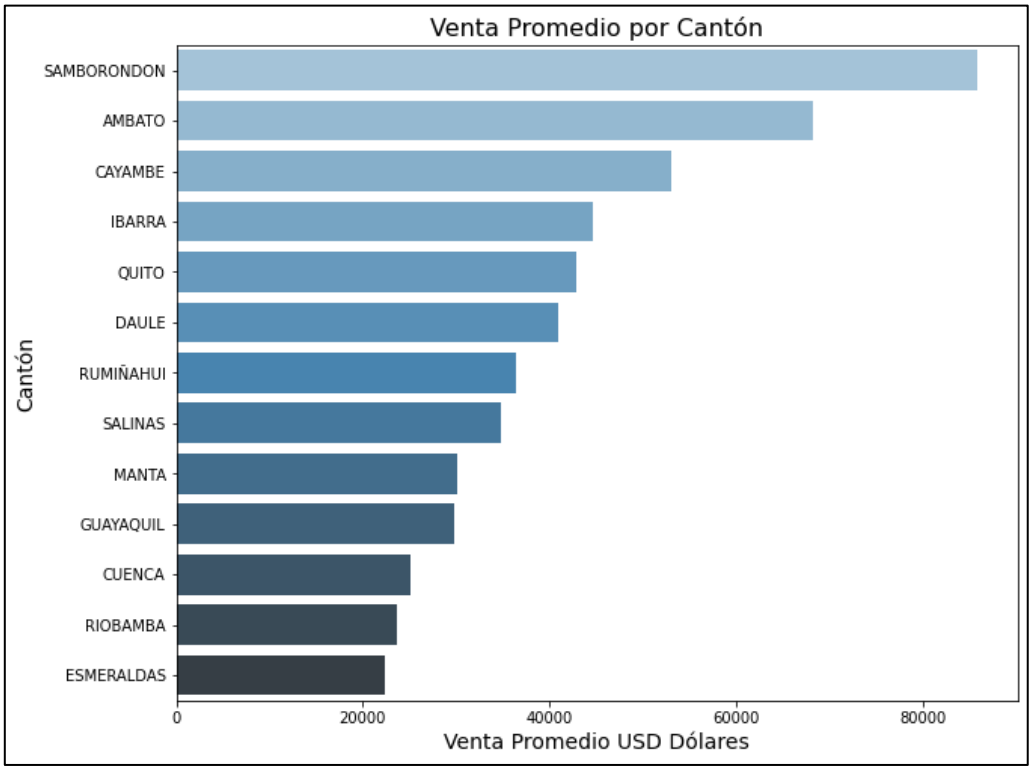
Por otro lado, Rumiñahui, Salinas, y Manta muestran ventas promedio que oscilan entre los USD 30,000 y USD 40,000, lo que sugiere un mercado moderado. Estas áreas pueden tener potencial para crecimiento, pero actualmente se encuentran en un nivel intermedio en cuanto a la venta de productos farmacéuticos.

Es notable que, a pesar de ser importantes ciudades en Ecuador, Guayaquil y Cuenca presentan ventas promedio más bajas en comparación con otros cantones como

Samborondón o Ambato. Esto podría ser un reflejo de la alta competencia interna o una mayor fragmentación del mercado en estas ciudades.

Finalmente, Riobamba y Esmeraldas se ubican con ventas promedio por debajo de los USD 30,000. Esto podría indicar un menor dinamismo económico o una demanda más dispersa en estas regiones, lo que las posiciona como áreas con menor actividad en el mercado farmacéutico en comparación con los cantones líderes.

Ilustración 12 Ventas Promedio por Provincia.



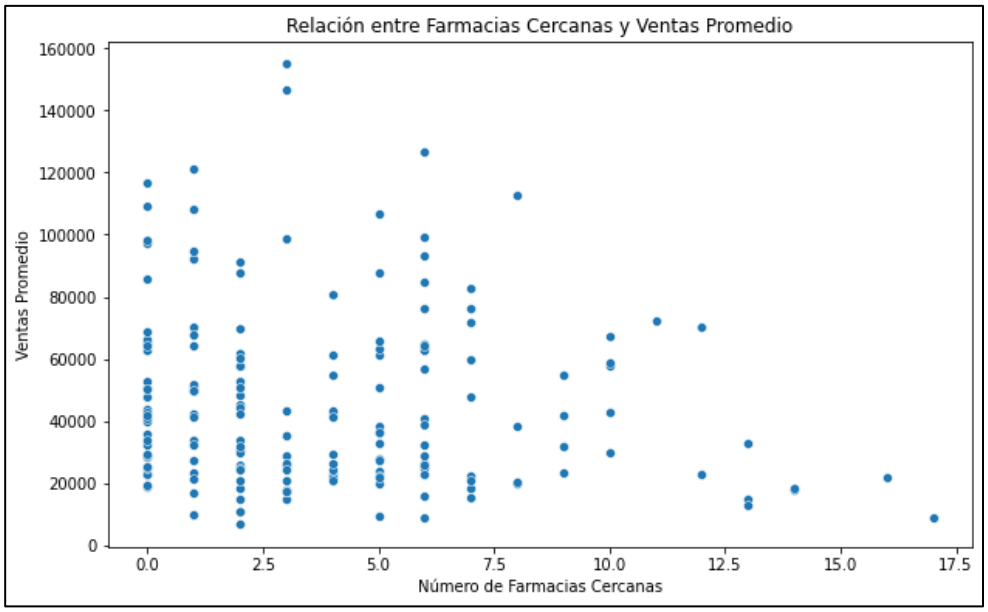
Relación entre el Número de Farmacias Cercanas y Ventas Promedio

Para evaluar la distancia la distancia entre dos farmacias pertenecientes a la empresa se calcula utilizando la función haversine, que mide la distancia en kilómetros entre dos puntos geográficos dados sus coordenadas de latitud y longitud. Después de

comparar una farmacia con todas las demás, el valor del contador de farmacias cercanas se almacena en la columna “nearby_farmacias” para cada una de las en el DataFrame, lo que permite obtener una cuenta precisa de cuántas farmacias están ubicadas en las proximidades de cada farmacia, considerando un radio de 800 metros.

Mediante el uso de la ilustración 13, visualizamos una relación débil entre el número de farmacias cercanas (nearby_farmacias). Por ejemplo, notamos que a partir de 10 farmacias cercanas las ventas promedio no superan el valor de 80.000 dólares mensuales.

Ilustración 6 "Diagrama de dispersión Farmacias Cercanas y Ventas Promedio"



Al evaluar la distancia de 800 metros como radio para considerar farmacias cercanas revela una tendencia en las ventas promedio a medida que aumenta el número de farmacias cercanas estas influyen de manera negativa en las ventas promedio.

El coeficiente de correlación entre el número de farmacias cercanas y las ventas promedio, con un radio de 800 metros, es -0.13 indicando una relación negativa débil entre las dos variables. En términos simples, sugiere que a medida que aumenta el número de farmacias cercanas dentro de un radio de 800 metros, las ventas promedio tienden a disminuir ligeramente, aunque la relación sigue siendo débil.

Al segmentar los datos por provincia el coeficiente de correlación de Guayas es de -0.25 lo que sugiere que factores geográficos podrían proporcionar un análisis más completo de las ventas promedio en relación con las farmacias cercanas.

Ventas Promedio y Fecha Mínima de Factura

Para el proyecto se consideran aquellos puntos de venta con inicio de operaciones anterior a junio 2024. Esto debido a la información necesaria para pronosticar la venta. Además, al analizar las ventas promedio. Estas varían en función de la fecha mínima de factura (minfechafactura) puede revelar tendencias temporales y la evolución de las ventas a lo largo del tiempo.

Como visualizamos en la ilustración 14. Las farmacias de mayor antigüedad (Fecha mínima de factura) muestran una amplia dispersión en ventas promedio, con algunas alcanzando valores muy altos. Esto sugiere que, en sus primeros años, estas farmacias han logrado establecerse fuertemente en el mercado. Sin embargo, las farmacias más recientes, sus ventas promedio parecen estar más dispersas y tienden a ser menores en comparación con las farmacias más antiguas. Esta tendencia nos

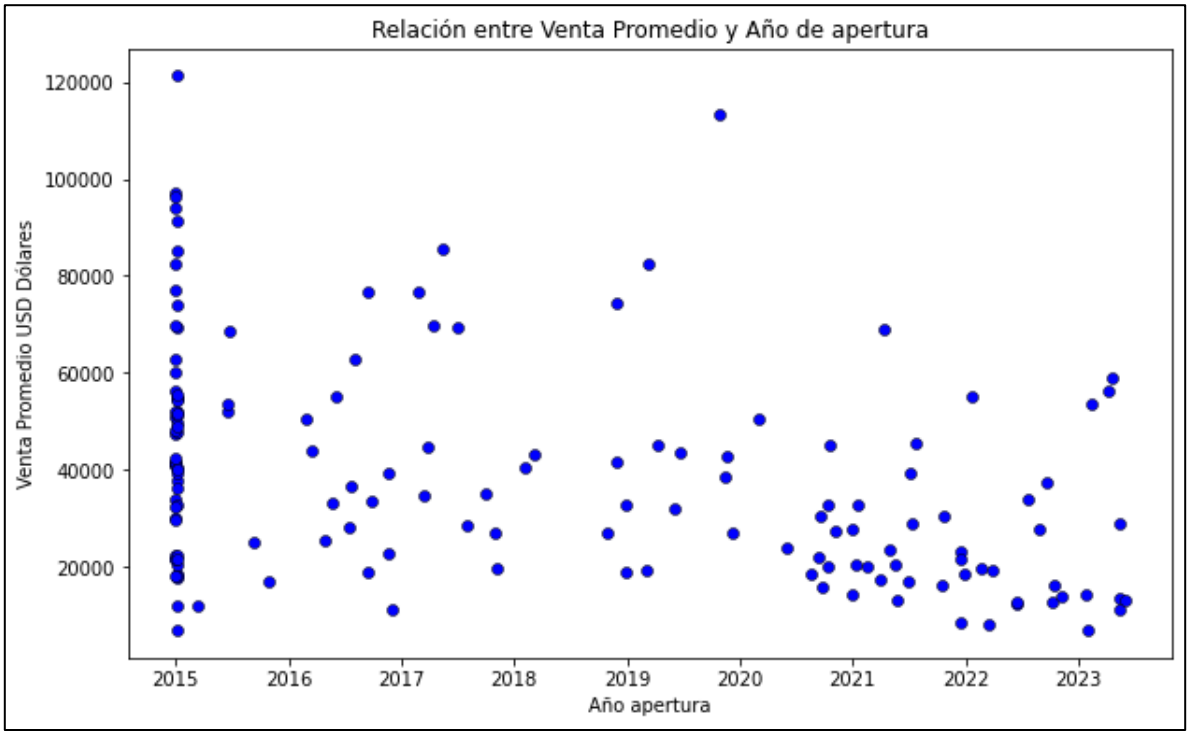
acerca a que las nuevas farmacias enfrentan desafíos significativos en alcanzar el mismo nivel de ventas que sus contrapartes más establecidas.

Este patrón puede indicar que la antigüedad y la experiencia en el mercado contribuyen a un mejor rendimiento, posiblemente debido a una mayor lealtad de los clientes, una mejor reputación y un entendimiento más profundo del mercado local. Además de un mercado saturado en ciertos sectores, además la variabilidad en las ventas promedio también es notablemente alta para estas farmacias, lo que sugiere que la antigüedad por sí sola no garantiza el éxito.

El valor del coeficiente es de -0.48 Un valor negativo indica una relación inversa entre la fecha mínima de factura y las ventas promedio. Esto significa que las farmacias más nuevas tienden a tener ventas promedio más bajas, coincidiendo con el análisis gráfico de la dispersión.

Las farmacias recién establecidas, en especial aquellas después de 2020, tienden a tener ventas promedio más bajas. Este fenómeno podría atribuirse a varios factores, como la alta competencia, cambios en el comportamiento del consumidor o la necesidad de tiempo para que las nuevas farmacias se consoliden en el mercado. La alta variabilidad en las ventas promedio de las farmacias más antiguas también sugiere que otros factores, además de la antigüedad, juegan un papel crucial en el rendimiento de ventas, como la ubicación, la gestión y las estrategias comerciales.

Ilustración 7 Ventas promedio actual vs año apertura (minfechafactura)



Ventas por Categoría ATC.

ATC, hace referencia al sistema de clasificación (Anatomical Therapeutic Chemical) fue establecido por la Organización Mundial de la Salud (OMS) específicamente diseñado y gestionado por "WHO Collaborating Centre for Drug Statistics Methodology". En esta clasificación, los medicamentos se agrupan de acuerdo con el órgano o sistema corporal sobre el que actúan, además de sus propiedades terapéuticas, farmacológicas y químicas. La ilustración 15 nos ayuda a comprender la distribución de las ventas promedio expresadas en miles de dólares por cada categoría en las farmacias de estudio, podemos interpretarla por categoría detalla a continuación.

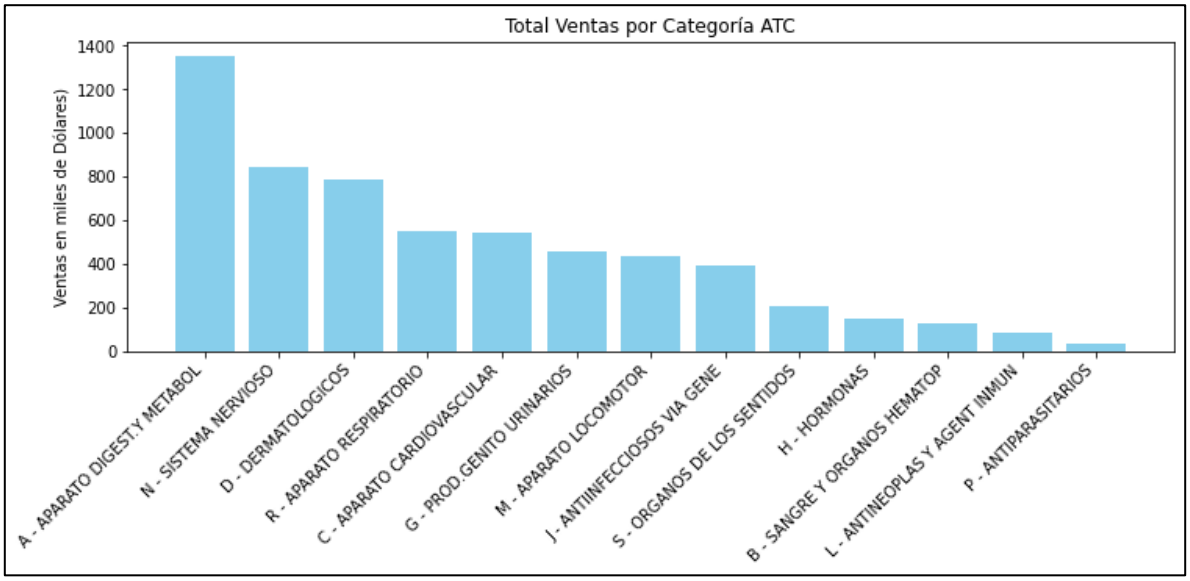
Medicamentos para el Aparato Digestivo y Metabólico “A - APARATO DIGEST.Y METABOL" lideran en ventas con un valor superior a 1.2 millones de dólares, lo que refleja una alta demanda de productos relacionados con el sistema digestivo y metabólico. Este dato puede sugerir que las enfermedades relacionadas con el sistema digestivo son prevalentes en la población atendida, lo que impulsa las ventas de productos dentro de esta categoría.

Además, la ilustración 15 nos ayuda a comprender el fuerte Desempeño del Sistema Nervioso y Dermatológicos "N - SISTEMA NERVIOSO" y "D - DERMATOLOGICOS" mostrando cifras de ventas significativas, superando los 800 mil dólares cada una. Lo que nos indica una alta demanda de medicamentos que afectan el sistema nervioso, como los psicotrópicos, y de productos dermatológicos, posiblemente debido a una alta incidencia de condiciones relacionadas con la piel.

Otras Categorías Relevantes: Las categorías como "R - APARATO RESPIRATORIO" y "C - APARATO CARDIOVASCULAR" también presentan cifras considerables de ventas, lo que podría estar relacionado con la prevalencia de enfermedades respiratorias y cardiovasculares. Este patrón es consistente con el aumento de enfermedades crónicas y las enfermedades del corazón en la población.

Categorías Menos Demandadas: En el otro extremo, las categorías como "B - SANGRE Y ORGANOS HEMATOP" y "P - ANTIPARASITARIOS" muestran las ventas más bajas. Esto podría deberse a una menor prevalencia de enfermedades que requieren estos tipos de medicamentos, o a una menor rotación de productos en estas áreas.

Ilustración 8 Total Ventas Promedio por Categoría ATC



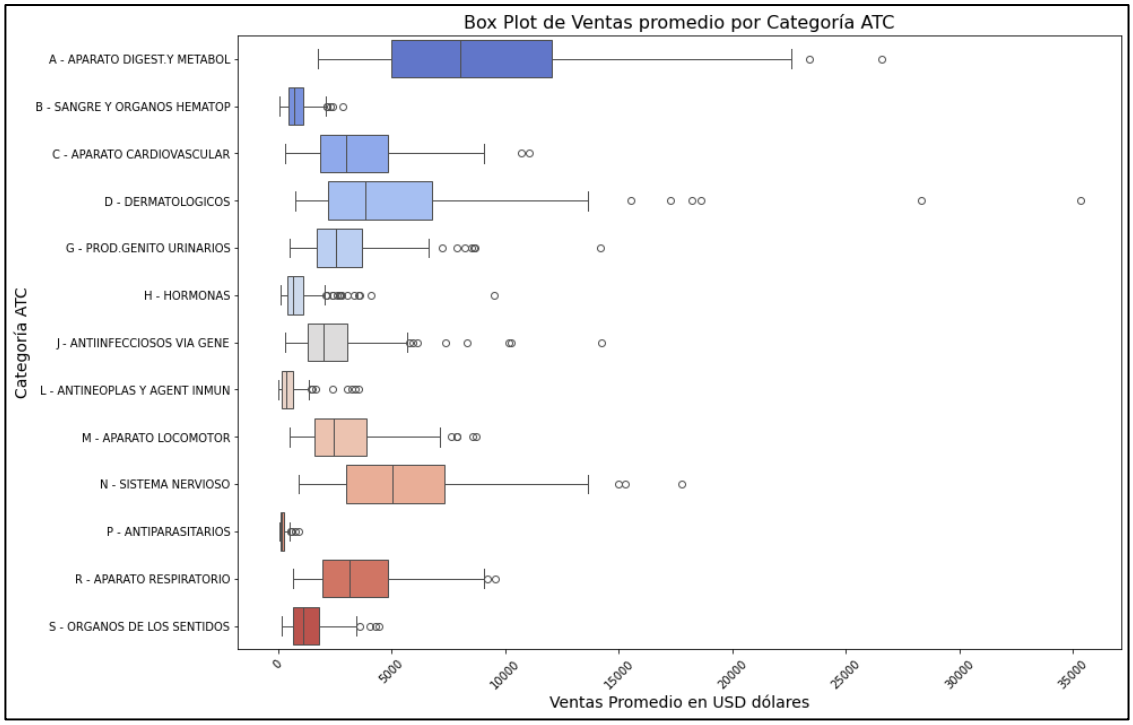
Identificación de valores atípicos por categoría ATC.

Apoyándonos en la ilustración 16, en términos generales las distribuciones de ventas promedio por Categoría ATC, presentan una distribución asimétrica, donde las medianas no están centradas. Esto indica que las ventas promedio no están distribuidas de manera simétrica, y tienden a estar sesgadas hacia uno u otro extremo en estos casos hacia la derecha. Lo que sugiere la presencia de farmacias con ventas significativamente mayores.

Además, la venta por categoría ATC presenta valores atípicos, lo cual es evidente por los puntos individuales que aparecen fuera de los “bigotes” de los box plots como lo percibimos en la ilustración 16, estos puntos que visiblemente se alejan de las cajas. Otro detalle es el rango intercuartílico (IQR), presentados como la longitud de la caja. Si esta se muestra amplia significa que hay una mayor dispersión de las ventas

promedio dentro de esa categoría ATC. Es decir, las farmacias tienen promedio que varían considerablemente. Si la caja es estrecha, la mayoría de las farmacias dentro de esa categoría tienen ventas promedio similares, reflejando una menor variabilidad o dispersión de los datos.

Ilustración 9 Box Plot, Ventas Promedio por Categoría ATC



Correlación de ubicación geográfica y venta por categoría ATC.

El análisis de correlación proporciona una visión clara de cómo se relacionan la participación de ventas de las distintas categorías de productos farmacéuticos con respecto a la provincia a la que pertenecen.

La ilustración 17 corresponde al Mapa de Calor de Correlación entre la participación de venta de Categorías ATC y la provincia a la que pertenece el punto de

venta, mostrando las relaciones entre diferentes categorías de productos y la ubicación geográfica. Visualmente podemos interpretar de la siguiente manera:

GUAYAS

- Correlación positiva moderada con varias categorías ATC, especialmente con "A - APARATO DIGEST.Y METABOL" (0.48), lo que sugiere que las farmacias en Guayas tienden a tener una mayor participación en ventas en esta categoría.
- Correlación negativa notable con "N - SISTEMA NERVIOSO" (-0.41) y "S - ORGANOS DE LOS SENTIDOS" (-0.42), lo que indica que estas categorías tienen una menor participación en ventas en esta provincia.

PICHINCHA

- Correlación negativa con "A - APARATO DIGEST.Y METABOL" (-0.37), sugiriendo que esta categoría tiene una menor participación en ventas en Pichincha en comparación con otras provincias.
- Correlación positiva con "N - SISTEMA NERVIOSO" (0.57) y "S - ORGANOS DE LOS SENTIDOS" (0.73), lo que indica que estas categorías son más significativas en las ventas de esta provincia.

SANTA ELENA

- No presenta correlaciones muy fuertes con ninguna categoría ATC, aunque tiene una correlación moderada con "A - APARATO DIGEST.Y METABOL" (0.20).

Esto podría indicar que esta categoría es ligeramente más relevante en las ventas en esta provincia.

AZUAY

- Presenta correlaciones bajas o moderadas con la mayoría de las categorías. Sin embargo, muestra una correlación negativa con "B - SANGRE Y ORGANOS HEMATOP" (-0.23), lo que sugiere una menor relevancia de esta categoría en las ventas de esta provincia.

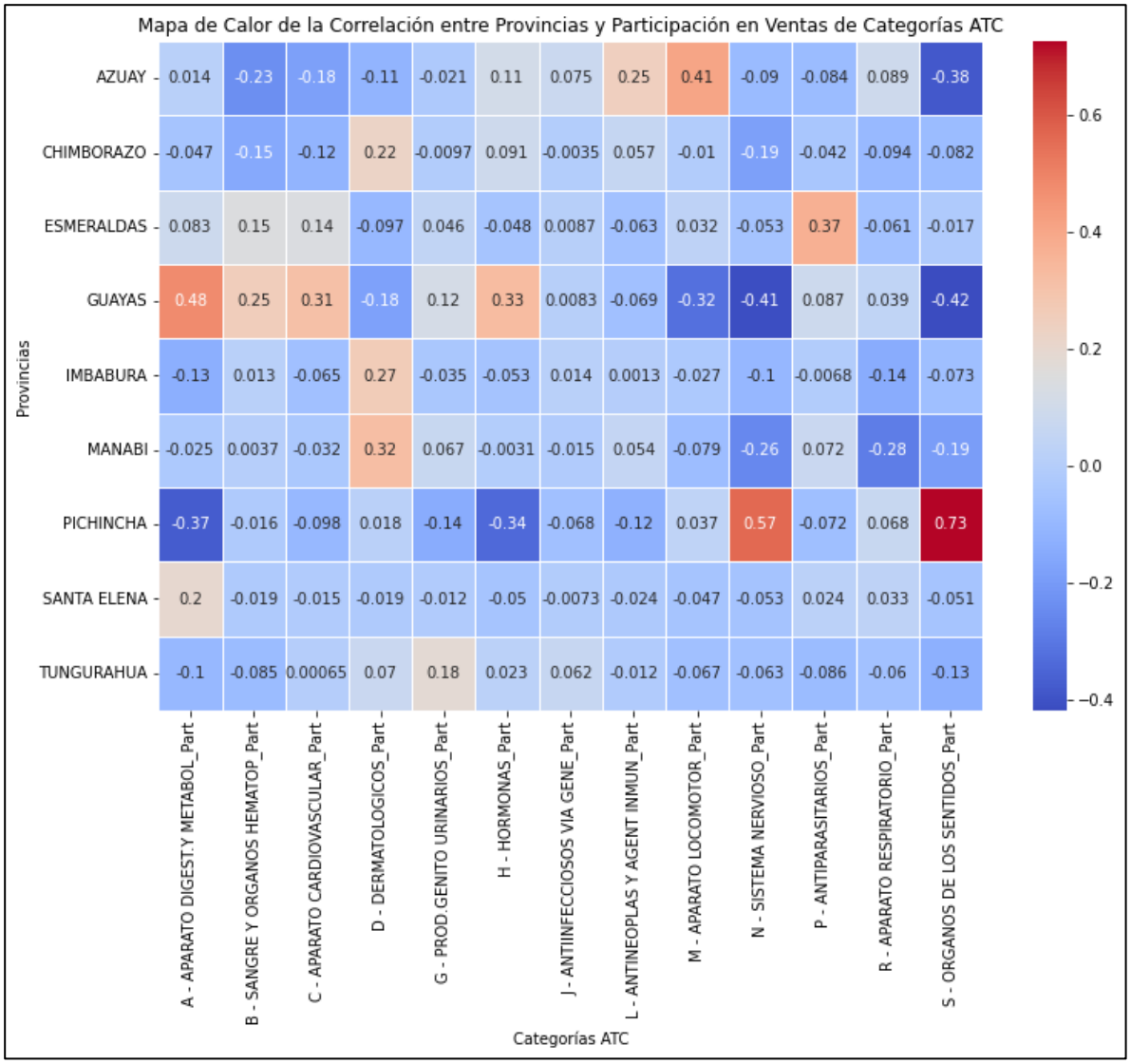
CHIMBORAZO

- Correlación positiva con "D - DERMATOLOGICOS" (0.22) y negativa con "N - SISTEMA NERVIOSO" (-0.19). Lo anterior sugiere que las ventas de productos dermatológicos son más significativas, mientras que las relacionadas con el sistema nervioso son menos importantes en esta provincia.

ESMERALDAS

- Muestra una correlación positiva con "B - SANGRE Y ORGANOS HEMATOP" (0.15) y "C - APARATO CARDIOVASCULAR" (0.14), lo que indica una mayor participación de estas categorías en las ventas en esta provincia.

Ilustración 10 Mapa de Calor, Correlación entre Categorías ATC



4.4.2 Modelado Clustering

Las técnicas seleccionadas para la clusterización son Kmeans, DBSCAN y Clustering Jerárquico. Para asegurar la validez y efectividad de cada uno de estos algoritmos, así como el comparar el de mejor rendimiento sobre el conjunto de datos. Se establece el coeficiente de Silhouette.

4.4.2.1 Construcción del Modelo de Clustering

Para la construcción del modelo se emplea PYTHON, mediante el uso de librerías PANDAS, NUMPY, e IMPALA. Esta última necesaria para la extracción mediante lenguaje de consulta estructurada (SQL). En el modelo de seleccionan específicamente las variables de participación porcentual, las cuales representan la proporción de ventas en distintas categorías ATC. Es decir, cada registro corresponderá a la participación porcentual asociada a cada una de las categorías de los medicamentos en función del órgano o sistema sobre el que actúan y sus propiedades terapéuticas, la selección de variables se muestra en la ilustración 18.

No se requiere normalización adicional, ya que las variables seleccionadas son participaciones porcentuales que ya están en una escala consistente y comparable. Además, la suma de las participaciones ya está controlada para cada farmacia y sus porcentajes proporcionan una interpretación directa y útil.

Ilustración 18 Selección de Variables para el Modelado Clustering.

```
# Seleccionar las variables
variables = ['A - APARATO DIGEST.Y METABOL_Part',
             'B - SANGRE Y ORGANOS HEMATOP_Part',
             'C - APARATO CARDIOVASCULAR_Part',
             'D - DERMATOLOGICOS_Part',
             'G - PROD.GENITO URINARIOS_Part',
             'H - HORMONAS_Part',
             'J - ANTIINFECCIOSOS VIA GENE_Part',
             'L - ANTINEOPLAS Y AGENT INMUN_Part',
             'M - APARATO LOCOMOTOR_Part',
             'N - SISTEMA NERVIOSO_Part',
             'P - ANTIPARASITARIOS_Part',
             'R - APARATO RESPIRATORIO_Part',
             'S - ORGANOS DE LOS SENTIDOS_Part']

df_selected = df[variables]
```

4.4.2.1.1 K-Means Clustering

Definidas las variables se determina el número óptimo de clústers, el cuál calcula los valores de "Within-Cluster Sum of Squares" (WCSS) para un rango de números de clústers (de 1 a 10). El objetivo es minimizar el WCSS al asignar puntos a los clústers de manera que dentro de un clúster estén lo más cerca posible del centroide. Cómo se visualiza la ilustración 19, El código itera sobre diferentes números de clústers. Se crea un modelo de KMeans con i clústers. Seguido de la implementación del parámetro “K-menas++” asegurando una inicialización de centroides más eficiente, ayudando a mejorar la convergencia del algoritmo.

Ilustración 19 Cálculo de valores WCSS

```
wcss = []
for i in range(1, 11):
    kmeans = KMeans(n_clusters=i, init='k-means++', max_iter=300, n_init=10, random_state=42)
    kmeans.fit(df_selected)
    wcss.append(kmeans.inertia_)
```

Se define el número máximo de iteraciones que el algoritmo ejecutará para intentar minimizar el WCSS (300 iteraciones). Además de especificar que el algoritmo se ejecutará 10 veces con diferentes inicializaciones y tomará la mejor configuración en términos de WCSS.

Por último, se establece una semilla para asegurar que los resultados sean reproducibles (42), los resultados de WCSS se muestran a en la ilustración 20.

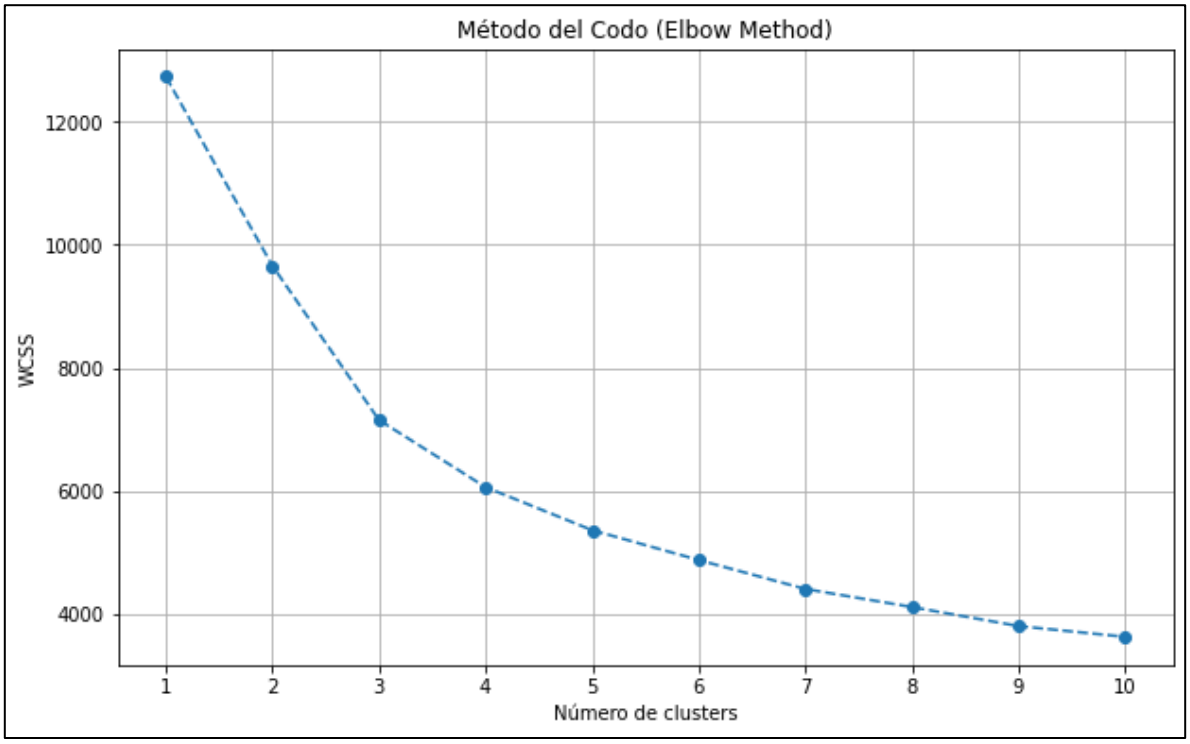
Ilustración 11 Resultados WCSS

```
In [3]: wcss
Out[3]:
[12732.152831024032,
 9659.387549489398,
 7151.92877363191,
 6061.696687314653,
 5362.153890933405,
 4876.703478542492,
 4406.560555675045,
 4113.021939845195,
 3803.7635925492423,
 3628.7649294081198]
```

Como visualizamos en la ilustración 20 donde de 1 a 2 clústers, el WCSS disminuye significativamente de 12732.15 a 9659.39. Es decir, al dividir los datos en dos clústers se reduce considerablemente la variabilidad dentro de los clústers, lo que es esperado. De 2 a 3 clústers, el WCSS sigue disminuyendo significativamente a 7151.93, lo que sugiere que dividir en tres clústers proporciona una mejor segmentación. A medida que continuamos aumentando el número de clústers (de 4 a 10), la reducción en WCSS comienza a ser menos pronunciada. Por ejemplo, de 6 a 7 clústers, el WCSS solo disminuye de 4876.70 a 4406.56.

Para elegir el número adecuado de clústeres se plasma el método del codo (Elbow Method) donde visualizamos que la disminución en WVSS se vuelve cada vez menos significativa al agregar más clústeres (ilustración 21). Es decir, el número adicionales que otorguemos a partir de este punto (número) no mejoran sustancialmente la comparación de los datos.

Ilustración 12 Método del Codo, Elección del número adecuado de Clústers.



La ilustración 21 muestra un "codo" alrededor de 3 clústers. Posterior a este punto, aunque el WCSS sigue disminuyendo, la tasa de disminución es mucho menor. Esto sugiere que agregar más clústers no mejora significativamente la compactación interna de los clústers. Por lo que, el número óptimo de clústers seleccionado es 3, ya que equilibra bien la reducción del WCSS y la simplicidad del modelo.

Finalmente aplicamos el algoritmo de kMeans con 3 cómo número de clústers. el método fit_predict ajusta el modelo a los datos y asignamos a cada observación un número de clúster específico (0, 1 o 2). Estos se almacenan en una nueva columna llamada cluster_kmeans en el DataFrame df_pdv_cluster_atc1.

La interpretación de los resultados sugiere ciertas tendencias generales que se describen al calcular estadísticas descriptivas para cada clúster descritas a continuación.

El primer clúster (clúster 0), sus estadísticos se visualizan en la ilustración 22, se caracteriza por dolencias del aparato digestivo, representa una media del 16.71% con una leve desviación estándar de 2.39%. Sugiere participación significativa y relativamente estable dentro de este clúster. Por su parte medicamentos antiinfecciosos tiene una alta media de participación (17.93%) sin embargo una desviación estándar considerable (7.14%).

Para finalizar aquellas dolencias del sistema nervioso presentan una media del 10.17% y una desviación estándar de 6.76%, lo que sugiere que esta categoría también es relevante, aunque con cierta variabilidad.

Ilustración 13 Estadísticos Clúster 0, Kmeans

Cluster	ATC1 Category	Mean	Std
0	A - APARATO DIGEST.Y METABOL_Part	16,71	2,39
0	B - SANGRE Y ORGANOS HEMATOP_Part	1,64	0,65
0	C - APARATO CARDIOVASCULAR_Part	6,01	2,20
0	D - DERMATOLOGICOS_Part	7,40	3,10
0	G - PROD.GENITO URINARIOS_Part	10,96	4,22
0	H - HORMONAS_Part	5,92	4,84
0	J - ANTIINFECCIOSOS VIA GENE_Part	17,93	7,14
0	L - ANTINEOPLAS Y AGENT INMUN_Part	4,07	3,21
0	M - APARATO LOCOMOTOR_Part	10,13	6,76
0	N - SISTEMA NERVIOSO_Part	10,17	3,98
0	P - ANTIPARASITARIOS_Part	0,55	0,71
0	R - APARATO RESPIRATORIO_Part	6,10	1,23
0	S - ORGANOS DE LOS SENTIDOS_Part	2,41	1,23

El segundo clúster (clúster 1), sus estadísticos se visualizan en la ilustración 23 destacando en dermatológicos con una media alta del 23.04% y una desviación estándar de 5.04%. Esto indica que esta categoría domina las ventas en este clúster. También caracterizados por aparato digestivo con una participación significativa (20.89%) y desviación estándar de 2.16% Otra de las categorías que representa a antiinfecciosos presenta una media de 5.78% con una desviación estándar de 2.13%, mostrando que esta dolencia es menos dominante en este segmento de farmacias.

Ilustración 14 Estadísticos Clase 1, Kmeans

Cluster	ATC1 Category	Mean	Std
1	A - APARATO DIGEST.Y METABOL_Part	20,89	2,16
1	B - SANGRE Y ORGANOS HEMATOP_Part	1,76	0,42
1	C - APARATO CARDIOVASCULAR_Part	7,46	1,54
1	D - DERMATOLOGICOS_Part	23,04	5,04
1	G - PROD.GENITO URINARIOS_Part	8,04	1,64
1	H - HORMONAS_Part	1,91	1,24
1	J - ANTIINFECCIOSOS VIA GENE_Part	5,78	2,13
1	L - ANTINEOPLAS Y AGENT INMUN_Part	1,52	1,10
1	M - APARATO LOCOMOTOR_Part	6,93	1,03
1	N - SISTEMA NERVIOSO_Part	11,07	1,61
1	P - ANTIPARASITARIOS_Part	0,51	0,15
1	R - APARATO RESPIRATORIO_Part	7,68	1,63
1	S - ORGANOS DE LOS SENTIDOS_Part	3,39	1,09

El tercer clúster (clúster 2), Sus estadísticos se visualizan en la ilustración 24, presenta la participación más alta a aquellas dolencias digestivas entre los clústers, con una media del 23.73% y una desviación estándar de baja (2.36%), lo que indica una gran importancia y consistencia dentro de este clúster. También encontramos dolencias asociadas al aparato cardiovascular con una participación significativa

(9.63%) con una desviación estándar de 2.51%, lo que sugiere que esta categoría es relevante. Por último, asociado al sistema nervioso con una alta participación 15.04%.

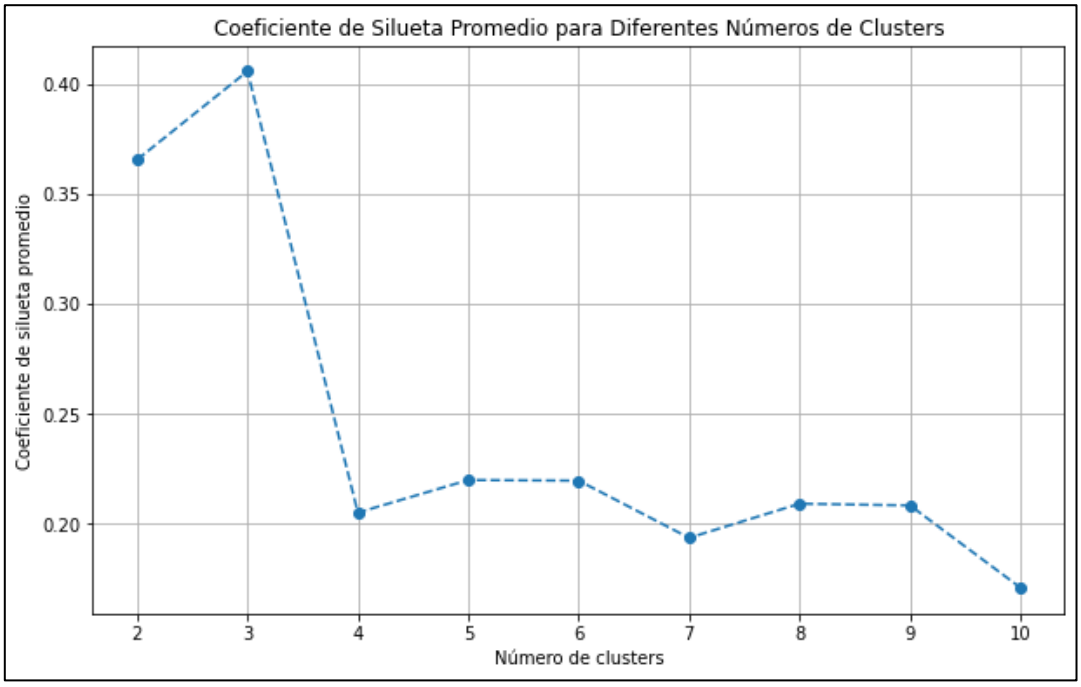
Ilustración 154 Estadísticos Clúster 2, Kmeans

Cluster	ATC1 Category	Mean	Std
2	A - APARATO DIGEST.Y METABOL_Part	23,73	2,36
2	B - SANGRE Y ORGANOS HEMATOP_Part	2,21	0,67
2	C - APARATO CARDIOVASCULAR_Part	9,63	2,51
2	D - DERMATOLOGICOS_Part	11,13	2,64
2	G - PROD.GENITO URINARIOS_Part	7,41	1,26
2	H - HORMONAS_Part	2,25	1,39
2	J - ANTIINFECCIOSOS VIA GENE_Part	6,09	1,63
2	L - ANTINEOPLAS Y AGENT INMUN_Part	1,17	1,20
2	M - APARATO LOCOMOTOR_Part	7,36	1,45
2	N - SISTEMA NERVIOSO_Part	15,04	2,49
2	P - ANTIPARASITARIOS_Part	0,52	0,19
2	R - APARATO RESPIRATORIO_Part	9,98	1,54
2	S - ORGANOS DE LOS SENTIDOS_Part	3,48	0,96

Al evaluar alternativas del número de clúster empleados se descartan por el coeficiente de silueta visualizado en la ilustración 25. El cuál presenta el promedio más alto cuando se utilizan 3 clústers, con un valor de aproximadamente 0.41. Esto indica que, para 3 clústers, los datos están mejor agrupados, y los clústers están bien separados entre sí. A partir de 4 clústers, el coeficiente de silueta promedio disminuye notablemente cercano a 0.2 y posterior a este se estabiliza en valores más bajos afectando a la calidad de la segmentación, posiblemente porque los clústers se vuelven menos distintivos o más solapados.

Al analizar por diferentes números de clúster bajo el esquema del coeficiente de silueta nos permite tomar la decisión de tres clústers es el número óptimo para una mejora agrupación del algoritmo de kmeans.

Ilustración 16 Coeficiente de Silueta



4.4.2.1.2 DBSCAN Clustering

Para analizar la segmentación más adecuada al modelo DBSCAN, se probaron diferentes combinaciones de los parámetros clave: eps, que establece el radio máximo para considerar puntos como vecinos, y min_samples, que define el número mínimo de puntos requeridos para formar un clúster. El objetivo es encontrar la combinación de parámetros que maximice el coeficiente de silueta. Esta métrica mide la calidad de los clústers en términos de su cohesión interna y separación respecto a otros clústers. En la ilustración 26, se imprimen los resultados obtenidos al aplicar DBSCAN con diferentes configuraciones de eps y min_samples:

Ilustración 17 Resultados DBSCAN, maximización del coeficiente de silueta.

eps	min_samples	silhouette_score
2	3	-0.254786
2	4	-1
2.5	3	-0.313503
2.5	4	-0.193834
3	3	-0.210983
3	4	-0.216839
3.5	3	-0.12258
3.5	4	-0.103206

Cómo observamos en la ilustración 26, los parámetros Óptimos: eps 3.5 y min_samples 4. Esta combinación arrojó el mejor (menos negativo) coeficiente de silueta de -0.1032. Aunque aún negativo, sugiere que los clústers formados con esta configuración tienen un nivel de cohesión y separación ligeramente mejor que con otras combinaciones.

Los coeficientes de silueta negativos indican que, en promedio, los puntos están más cerca de los centroides de otros clústers que de su propio clúster, lo que sugiere un solapamiento o clústers mal definidos.

Los resultados también muestran que cuando min_samples es mayor (como 4), DBSCAN tiende a clasificar más puntos como ruido, especialmente con valores más bajos de eps, como 2.0. Esto se evidencia en un coeficiente de silueta de -1.000 cuando

eps es igual a 2.0 y min_samples equivalente a 4, lo que sugiere que la mayoría de los puntos fueron clasificados como ruido o agrupados en un único clúster.

La interpretación de los estadísticos para DBSCAN arrojan los siguientes resultados bajo el esquema de DBSCAN, considerando los clústers formados y el ruido identificado. Clúster -1 (Ruido, ilustración 29): Los puntos clasificados bajo clúster -1 han sido identificados por DBSCAN como ruido, es decir, no pertenecen a ningún clúster significativo, sus estadísticos se presentan en la ilustración 27.

Ilustración 18 Estadísticos Clúster -1 DBSCAN

Cluster	ATC1 Category	Mean	Std
-1	A - APARATO DIGEST.Y METABOL_Part	23,13	3,64
-1	B - SANGRE Y ORGANOS HEMATOP_Part	2,12	0,72
-1	C - APARATO CARDIOVASCULAR_Part	9,03	2,87
-1	D - DERMATOLOGICOS_Part	13,02	6,38
-1	G - PROD.GENITO URINARIOS_Part	7,87	2,19
-1	H - HORMONAS_Part	2,77	2,41
-1	J - ANTIINFECCIOSOS VIA GENE_Part	7,38	4,51
-1	L - ANTINEOPLAS Y AGENT INMUN_Part	1,69	1,85
-1	M - APARATO LOCOMOTOR_Part	7,49	2,68
-1	N - SISTEMA NERVIOSO_Part	12,87	2,87
-1	P - ANTIPARASITARIOS_Part	0,54	0,29
-1	R - APARATO RESPIRATORIO_Part	9,06	2,20
-1	S - ORGANOS DE LOS SENTIDOS_Part	3,05	1,03

Clúster 0: Agrupa un conjunto de puntos que DBSCAN ha identificado como densamente conectados. Sus estadísticos se visualizan en la ilustración 28. Donde dolencias del aparato digestivo con una participación media del 22.70% y una desviación estándar de 1.39%, esta categoría muestra una alta consistencia dentro del clúster, indicando que es una categoría clave y estable. Para aquellos del sistema nervioso, mantiene una alta participación media del 16.16% y una desviación estándar

de 1.68%, lo que sugiere que es una categoría dominante y relativamente estable. Antiinfecciosos presenta una media de 5.77% con una desviación estándar menor (1.16%), lo que indica una presencia importante pero menos dominante que en otros clústers, con baja variabilidad.

Ilustración 28 Estadísticos Chuste 0, DBSCAN

Cluster	ATC1 Category	Mean	Std
0	A - APARATO DIGEST.Y METABOL_Part	22,70	1,39
0	B - SANGRE Y ORGANOS HEMATOP_Part	2,11	0,57
0	C - APARATO CARDIOVASCULAR_Part	9,50	2,08
0	D - DERMATOLOGICOS_Part	11,75	2,41
0	G - PROD.GENITO URINARIOS_Part	7,41	0,94
0	H - HORMONAS_Part	1,89	0,51
0	J - ANTIINFECCIOSOS VIA GENE_Part	5,77	1,16
0	L - ANTINEOPLAS Y AGENT INMUN_Part	0,95	0,56
0	M - APARATO LOCOMOTOR_Part	7,42	1,04
0	N - SISTEMA NERVIOSO_Part	16,16	1,68
0	P - ANTIPARASITARIOS_Part	0,49	0,15
0	R - APARATO RESPIRATORIO_Part	9,92	1,00
0	S - ORGANOS DE LOS SENTIDOS_Part	3,93	0,75

El Clúster 1, que contiene otro conjunto de puntos que DBSCAN ha agrupado como densamente conectados.

Los estadísticos del clúster 1 se presentan en la ilustración 29. Donde dolencias digestivas muestra una participación media del 22.04% con una desviación estándar muy baja (0.47%), demostrando una gran consistencia y una importancia significativa en este clúster. Aquellos del sistema nervioso es la categoría más dominante en este clúster con una media de 18.49% y una desviación estándar muy baja (0.77%). Dermatológicos, aunque no es la más dominante, tiene una participación media del 10.29% con baja variabilidad (desviación estándar de 0.87%).

Ilustración 29 Estadísticos Clúster 1. DBSCAN

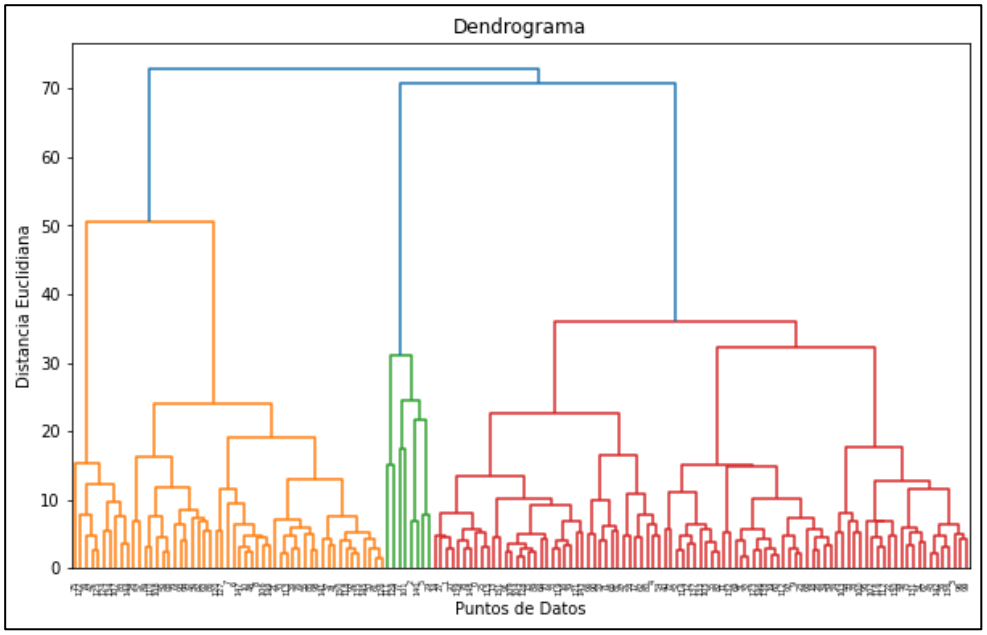
Cluster	ATC1 Category	Mean	Std
1	A - APARATO DIGEST.Y METABOL_Part	22,04	0,47
1	B - SANGRE Y ORGANOS HEMATOP_Part	2,19	0,86
1	C - APARATO CARDIOVASCULAR_Part	6,43	0,89
1	D - DERMATOLOGICOS_Part	10,29	0,87
1	G - PROD.GENITO URINARIOS_Part	7,84	0,96
1	H - HORMONAS_Part	1,95	0,17
1	J - ANTIINFECCIOSOS VIA GENE_Part	5,94	0,33
1	L - ANTINEOPLAS Y AGENT INMUN_Part	0,61	0,31
1	M - APARATO LOCOMOTOR_Part	7,55	0,76
1	N - SISTEMA NERVIOSO_Part	18,49	0,77
1	P - ANTIPARASITARIOS_Part	0,63	0,15
1	R - APARATO RESPIRATORIO_Part	11,85	0,91
1	S - ORGANOS DE LOS SENTIDOS_Part	4,19	0,59

Clustering Jerárquico

El dendograma (ilustración 30) muestra cómo se combinan los clústers en función de una medida de distancia. Se utilizó el método de enlace Ward, que minimiza la varianza dentro de cada clúster al combinarlos. Ideal para mantener los clústers compactos y homogéneos. Las "bifurcaciones" en el dendrograma representan los puntos donde los clústers se combinan, y la altura de estas bifurcaciones indica la distancia entre los clústers fusionados.

Para identificar el número óptimo de clústers, se puede observar el dendrograma (ilustración 30) buscando la mayor distancia vertical que se pueda cortar horizontalmente sin intersectar demasiadas ramas. Determinando que 3 clústers es el número óptimo. El modelo empleado corresponde a “Agglomerative Clustering”, donde cada punto de datos se asigna a uno de los tres clústers identificados.

Ilustración 190 Dendograma Clustering Jerárquico



A continuación, estadísticas descriptivas (media y desviación estándar) para las participaciones porcentuales de diferentes categorías ATC, segmentadas por clústers identificados mediante clustering jerárquico:

Primer clúster (clúster 0, ilustración 31): Incluye farmacias que comparten características similares en cuanto a la distribución de sus ventas entre las diferentes categorías ATC. Aparato digestivo presenta una participación media del 21.92%, con una desviación estándar de 2.08%. Esto indica que esta categoría tiene una importancia significativa y consistente dentro de este clúster. Dermatológicos, destaca con una alta participación media del 17.71%, aunque con una desviación estándar más alta (5.28%), lo que sugiere una variabilidad considerable en esta categoría entre las farmacias de este clúster. Sistema nervioso tiene una participación media del 12.87% con una desviación estándar de 2.31%, lo que muestra que esta

categoría es relevante pero también presenta variabilidad en su importancia dentro del clúster.

Ilustración 20 Estadísticos Clúster 0. Jerárquico

Cluster	ATC1 Category	Mean	Std
0	A - APARATO DIGEST.Y METABOL_Part	21,94	2,09
0	B - SANGRE Y ORGANOS HEMATOP_Part	1,80	0,42
0	C - APARATO CARDIOVASCULAR_Part	7,88	1,61
0	D - DERMATOLOGICOS_Part	17,73	5,33
0	G - PROD.GENITO URINARIOS_Part	8,07	1,88
0	H - HORMONAS_Part	2,01	1,04
0	J - ANTIINFECCIOSOS VIA GENE_Part	6,14	1,82
0	L - ANTINEOPLAS Y AGENT INMUN_Part	1,20	0,89
0	M - APARATO LOCOMOTOR_Part	7,27	1,62
0	N - SISTEMA NERVIOSO_Part	12,83	2,31
0	P - ANTIPARASITARIOS_Part	0,53	0,16
0	R - APARATO RESPIRATORIO_Part	9,11	1,85
0	S - ORGANOS DE LOS SENTIDOS_Part	3,50	1,07

Segundo Clúster (Clúster 1. Ilustración 32): La participación de aparato digestivo presenta una media es mayor, con un 24.09%, y una desviación estándar de 2.47%, indicando que esta categoría es muy importante y relativamente consistente en las farmacias agrupadas aquí. Aparato cardiovascular, tiene una participación significativa del 10.16% con una mínima desviación. En cuanto al sistema nervioso muestra una participación media del 15.38%, hay variabilidad en la relevancia de esta categoría entre las farmacias de este clúster.

Tercer Clúster (Clúster 2, Ilustración 33): Incluye farmacias donde la distribución de las ventas está dominada por diferentes categorías ATC en comparación con los otros dos clústers. Para aparato digestivo presenta la participación más baja entre los clústers, con una media del 16.54% indicando que

esta categoría es menos dominante en este clúster. Antiinfecciosos, presenta la participación más alta en este clúster con una media del 18.78% sin embargo su desviación estándar de 7.13%, lo que sugiere una alta importancia, pero también una considerable variabilidad en su participación. De igual manera para sistema nervioso participación media del 10.17% con una desviación estándar de 4.25%

Ilustración 32 Estadísticos Clúster 1, Jerárquico

Cluster	ATC1 Category	Mean	Std
1	A - APARATO DIGEST.Y METABOL_Part	24,09	2,47
1	B - SANGRE Y ORGANOS HEMATOP_Part	2,35	0,68
1	C - APARATO CARDIOVASCULAR_Part	10,16	2,56
1	D - DERMATOLOGICOS_Part	9,95	1,85
1	G - PROD.GENITO URINARIOS_Part	7,26	1,11
1	H - HORMONAS_Part	2,34	1,53
1	J - ANTIINFECCIOSOS VIA GENE_Part	6,05	1,73
1	L - ANTINEOPLAS Y AGENT INMUN_Part	1,24	1,32
1	M - APARATO LOCOMOTOR_Part	7,28	1,24
1	N - SISTEMA NERVIOSO_Part	15,38	2,60
1	P - ANTIPARASITARIOS_Part	0,51	0,20
1	R - APARATO RESPIRATORIO_Part	9,95	1,61
1	S - ORGANOS DE LOS SENTIDOS_Part	3,43	0,95

Ilustración 21 Estadísticos Clúster 2, Jerárquico

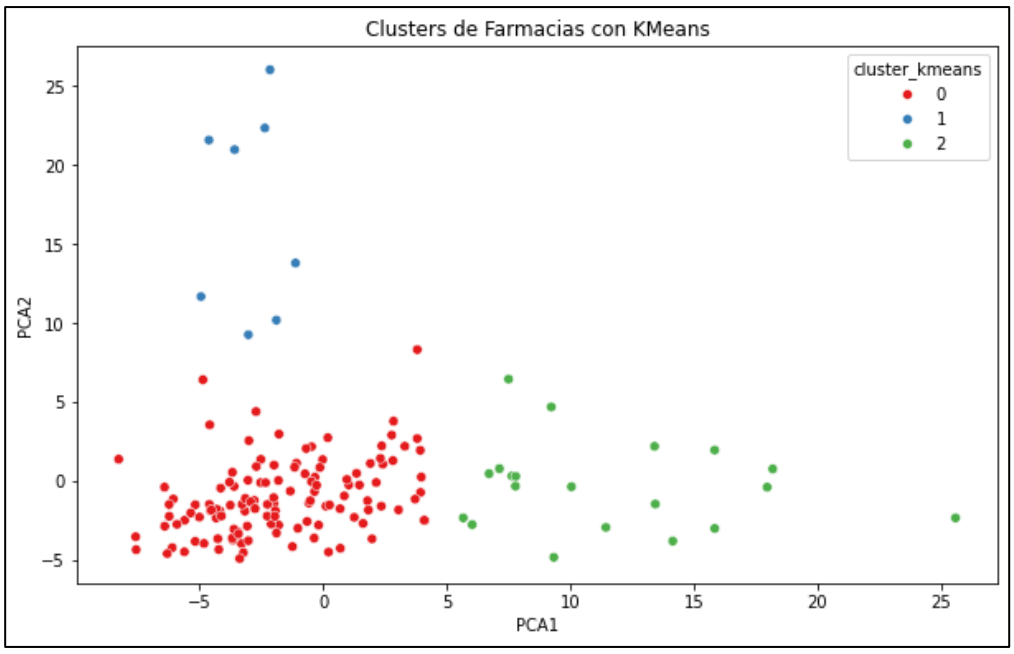
Cluster	ATC1 Category	Mean	Std
2	A - APARATO DIGEST.Y METABOL_Part	16,54	2,49
2	B - SANGRE Y ORGANOS HEMATOP_Part	1,66	0,69
2	C - APARATO CARDIOVASCULAR_Part	5,76	2,21
2	D - DERMATOLOGICOS_Part	6,58	2,01
2	G - PROD.GENITO URINARIOS_Part	10,40	4,13
2	H - HORMONAS_Part	6,18	5,11
2	J - ANTIINFECCIOSOS VIA GENE_Part	18,78	7,13
2	L - ANTINEOPLAS Y AGENT INMUN_Part	4,41	3,26
2	M - APARATO LOCOMOTOR_Part	10,58	7,08
2	N - SISTEMA NERVIOSO_Part	10,17	4,25
2	P - ANTIPARASITARIOS_Part	0,56	0,75
2	R - APARATO RESPIRATORIO_Part	5,91	1,16
2	S - ORGANOS DE LOS SENTIDOS_Part	2,48	1,29

4.4.2.2 Evaluación y comparación de las Técnicas de Modelado Clustering

KMeans Clustering:

- Coeficiente de Silueta: 0.4058
- Interpretación: Este valor indica que KMeans ha logrado una buena separación entre clústers y una fuerte cohesión interna. De todos los modelos evaluados, KMeans obtuvo el coeficiente de silueta más alto, lo que sugiere que es la mejor opción para segmentar estos datos.
- Nos apoyamos en la ilustración 34 que refiere a la visualización de clústers mediante K-Means, se utiliza reducción de dimensiones PCA para poder representarlos en dos dimensiones. Notamos que los clústers están separados, especialmente en el eje PCA1. Los datos están bien agrupados lo que sugiere que los clústers tienen características distintivas.

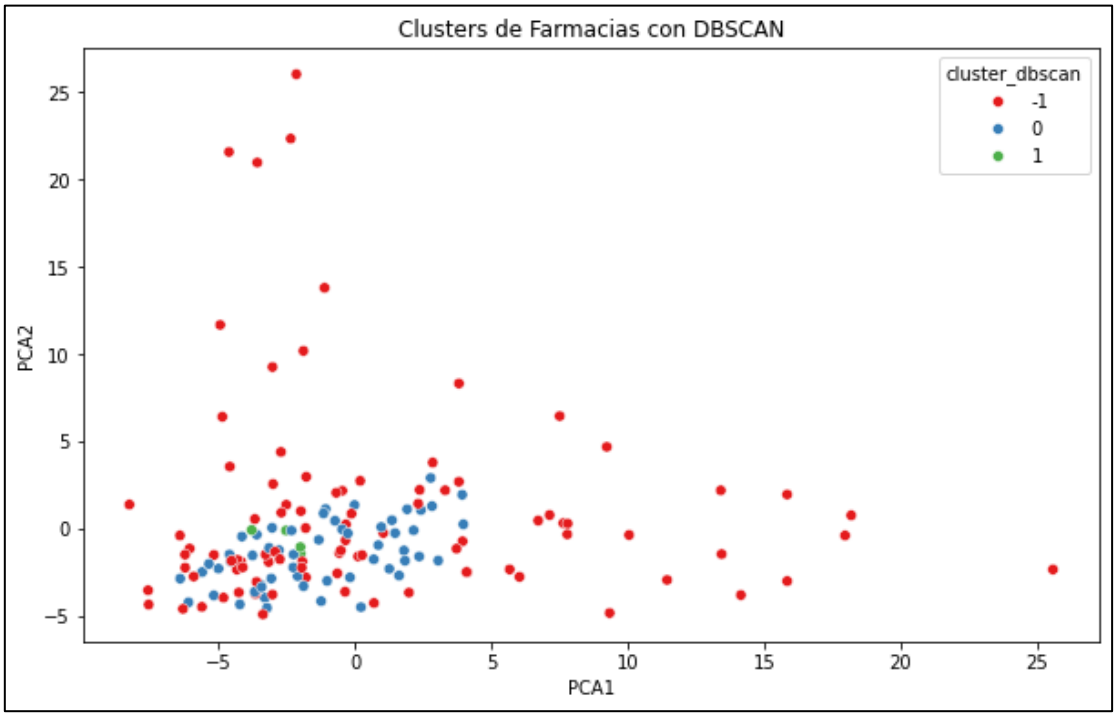
Ilustración 22 Clusterización Kmeans.



DBSCAN:

- Coeficiente de Silueta: -0.1032
- Interpretación: El coeficiente negativo indica dificultades para formar clústers significativos. Esto podría deberse a una falta de estructura de densidad clara en los datos para ser capturada por el algoritmo. Sugiriendo que DBSCAN no es adecuado para este conjunto de datos.
- La ilustración 35 corresponde a la visualización de clústers obtenidos mediante DBSCAN, utilizando PCA (Análisis de Componentes Principales) para reducir la dimensionalidad de los datos y poder representarlos en dos dimensiones. Al compararla con la ilustración 34 (KMeans), DBSCAN clasifica muchos puntos como ruido. Además, no se muestran claramente definidos.

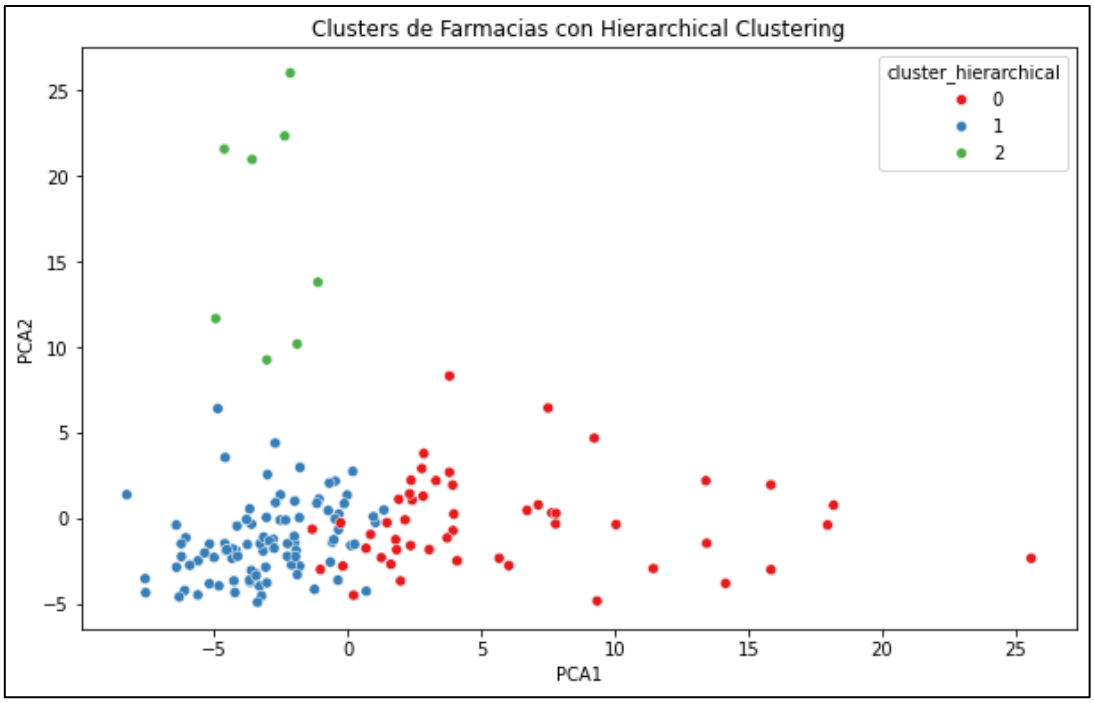
Ilustración 23 Clustering DBSCAN



Clustering Jerárquico:

- Coeficiente de Silueta: 0.2409
- Interpretación: El clustering jerárquico produjo un coeficiente de silueta positivo, lo que indica que ha logrado una segmentación aceptable, aunque no tan fuerte como la de KMeans. Esto sugiere que este método pudo identificar algunos patrones en los datos, pero los clústers no están tan claramente definidos como con KMeans, podemos visualizar esto en la ilustración 36.

Ilustración 24 Clustering Jerárquico

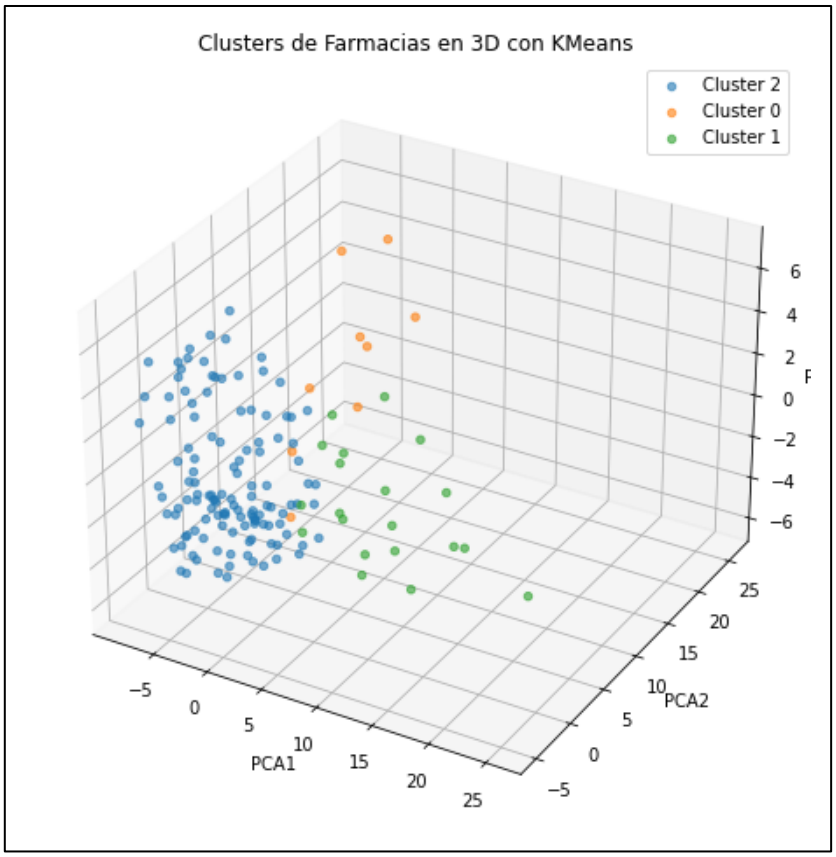


En general, KMeans es la opción recomendada para la segmentación de las farmacias según las categorías ATC, mientras que DBSCAN puede descartarse para este análisis. El clustering jerárquico, aunque menos eficaz que KMeans, sigue siendo

útil para análisis complementarios, especialmente si se desea explorar la estructura jerárquica de los datos.

A continuación, la ilustración 37 nos permite visualizar los clústers de farmacias generados mediante el algoritmo KMeans. Cada punto representa una farmacia y está coloreado según el clúster al que pertenece. Los ejes corresponden a las tres primeras componentes principales (PCA1, PCA2, y PCA3), obtenidas a través de un análisis de componentes principales (PCA), permitiendo reducir la dimensionalidad de los datos originales. Facilitando la comprensión de cómo se distribuyen las farmacias en el espacio tridimensional y evaluar la separación entre los clústers formados.

Ilustración 25 Tres primeras componentes principales. Clústers Kmeans.



4.4.3 Modelado de Pronósticos

Dado el contexto del problema de pronósticos de series temporales, se opta por emplear modelos que pueden manejar datos secuenciales y captar patrones complejos las técnicas usadas provienen de Modelos Avanzados, Modelos Tradicionales de Series Temporales y Modelos Clásicos de Regresión Lineal.

Modelos Avanzados: Se elige utilizar el modelo LSTM (Long Short-Term Memory), una variante de las redes neuronales recurrentes (RNN) que está especialmente diseñada para gestionar datos secuenciales y captar dependencias a largo plazo.

Modelos Tradicionales de Series Temporales: Se selecciona el modelo ARIMA (AutoRegressive Integrated Moving Average), ampliamente reconocido por su eficacia en el análisis de series temporales.

Modelos Clásicos de Regresión y Análisis Temporal: Para completar la gama de técnicas de modelado, se incorporan enfoques más tradicionales como la regresión lineal, Random Forest, Holt-Winters y Support Vector Regression (SVR). Estos modelos se seleccionan con el propósito de proporcionar puntos de referencia y evaluar si los modelos menos complejos podían competir en rendimiento con los modelos más avanzados.

Se establece cómo las métricas de evaluación el MSE (Mean Squared Error) y el MAE (Mean Absolute Error), las cuales cuantifican la magnitud del error de las predicciones en comparación con los valores reales. Adicional a las métricas establecidas MSE y MAE se opta por la precisión adaptada, definiéndola como el

porcentaje de predicciones que caen dentro de un rango de tolerancia del 30% respecto a los valores reales, ofreciendo una medida de cuán cercanas están las predicciones a los valores reales dentro de un margen aceptable.

4.4.3.1 Construcción del Modelo de Pronóstico

La siguiente fase de la metodología CRISP-DM establece la construcción de los diferentes modelos. Cada modelo fue ajustado según las características específicas de los datos y la técnica seleccionada.

4.4.3.1.1 Construcción del Modelo LSTM

Como parte del proceso de construcción del modelo LSTM, se incluyó la normalización de los datos de ventas, el código empleado se presenta en la ilustración 38. Es proceso implica escalar los valores al rango [0, 1], ayudando a mejorar la eficiencia del entrenamiento del modelo y asegura que todas las variables entren en la red en un rango comparable.

Ilustración 3826 Escalamiento de los datos previo a LSTM.

```
# Escalar los datos de ventas
scaler = MinMaxScaler(feature_range=(0, 1))
valorventa_scaled = scaler.fit_transform(farmacia_df['valorventa'].values.reshape(-1, 1))
```

Se establece un proceso (ilustración 39) para crear secuencias temporales que sirvieron como entradas para el modelo LSTM. Estas secuencias permitieron que el modelo capturara patrones temporales en los datos, lo que es esencial para predecir con precisión las ventas futuras.

Ilustración 39 Función para crear secuencia en los datos, previo a LSTM

```python
# Función para crear secuencias de datos para el LSTM
def create_sequences(data, time_steps):
    sequences = []
    labels = []
    for i in range(len(data) - time_steps):
        seq = data[i:(i + time_steps)]
        label = data[i + time_steps]
        sequences.append(seq)
        labels.append(label)
    return np.array(sequences), np.array(labels)

# Crear secuencias con una ventana de tiempo de 3
X, y = create_sequences(valorventa_scaled, time_steps=3)
```

El modelo LSTM fue construido utilizando los hiperparámetros óptimos previamente determinados para cada clúster de farmacias (Anexo 2 Modelado LSTM). Se implementa una arquitectura de red que incluye una capa LSTM capaz de captar dependencias a largo plazo, seguida de una capa densa para generar la predicción final. Se añade una capa de Dropout con un valor del 20% para reducir el riesgo de sobreajuste, y se utiliza Early Stopping para detener el entrenamiento si la pérdida en el conjunto de validación no mejoraba tras 10 épocas consecutivas. El código se detalla en la ilustración 40.

Ilustración 27 Construcción del modelo LSTM

```python
# Construir el modelo LSTM con Dropout para evitar sobreajuste
model = Sequential()
model.add(Input(shape=(X_train.shape[1], 1)))  # Definir la forma de entrada
model.add(LSTM(units=params['neurons'], return_sequences=False))
model.add(Dropout(0.2))  # Añadir Dropout del 20% para mitigar el sobreajuste
model.add(Dense(1))  # Capa densa de salida
model.compile(optimizer=Adam(learning_rate=params['learning_rate']), loss='mean_squared_error')

# Configurar Early Stopping para detener el entrenamiento si no mejora
early_stopping = EarlyStopping(monitor='val_loss', patience=10, restore_best_weights=True)

# Entrenar el modelo LSTM con los hiperparámetros óptimos y Early Stopping
history = model.fit(X_train, y_train, validation_data=(X_test, y_test),
                    epochs=params['epochs'], batch_size=params['batch_size'],
                    callbacks=[early_stopping], verbose=0)

# Realizar predicciones en el conjunto de prueba
y_pred = model.predict(X_test)
```

4.4.3.1.2 Construcción del Modelo ARIMA

Para construir el modelo ARIMA, se establece un plan de búsqueda de parámetros (ilustración 41), donde se exploraron diferentes combinaciones de los valores p (autoregresivos), d (diferenciación), y q (promedio móvil). Esta búsqueda permite identificar la configuración óptima que minimizara el error en las predicciones, asegurando así un rendimiento robusto del modelo.

Para cada farmacia, se construyó un modelo ARIMA ajustado a los datos históricos de ventas (se evidencia en Anexo 2 Modela ARIMA). Posterior se realizan múltiples iteraciones para encontrar los valores óptimos de p, d, y q, y el modelo fue ajustado en consecuencia. Este enfoque permitió captar tanto las tendencias a corto como a largo plazo en los datos de ventas.

Ilustración 28 Algoritmo de búsqueda de mejores parámetros para modelado ARIMA

```
# Búsqueda de los mejores parámetros p, d, q
for p in p_values:
    for d in d_values:
        for q in q_values:
            try:
                model = ARIMA(valorventa_series, order=(p,d,q))
                model_fit = model.fit()
                y_pred = model_fit.predict(start=len(valorventa_series)//2, end=len(valorventa_series)-1)
                y_true = valorventa_series[len(valorventa_series)//2:]
                mse = mean_squared_error(y_true, y_pred)

                if mse < best_score:
                    best_score, best_cfg = mse, (p,d,q)
            except:
                continue

print(f"Mejores parámetros ARIMA para farmacia {codfarmacia}: {best_cfg} con MSE: {best_score}")

# Entrenar el modelo ARIMA final con los mejores parámetros
model = ARIMA(valorventa_series, order=best_cfg)
model_fit = model.fit()
y_pred = model_fit.predict(start=len(valorventa_series)//2, end=len(valorventa_series)-1)
y_true = valorventa_series[len(valorventa_series)//2:]
```

4.4.3.1.3 Construcción de Modelos Clásicos

Se implementaron modelos de regresión lineal, su construcción se presenta en la ilustración 42, Random Forest (ilustración 43), Holt-Winters (ilustración 44) y SVR (ilustración 45) para cada una de las farmacias. Estos modelos fueron ajustados a los datos históricos de ventas y se utilizaron para generar predicciones que luego se compararon con los valores reales.

Ilustración 292 Construcción de Modelos Clásicos, Regresión Lineal

```
# Regresión Lineal
if len(valorventa_series) > 0:
    model_lr = LinearRegression()
    model_lr.fit(X_train, y_train)
    pred_rl = model_lr.predict(X_test)
    df_pronosticos.loc[farmacia_df.index[-len(X_test):], 'valorventa_regresion_lineal'] = pred_rl.flatten()

    # Calcular Durbin-Watson para Regresión Lineal en el conjunto de prueba
    residuos_rl = y_test.flatten() - pred_rl.flatten()
    dw_stat_rl = durbin_watson(residuos_rl)
    dw_stats_rl.append(dw_stat_rl)
    mse_rl.append(mean_squared_error(y_test, pred_rl))
    mae_rl.append(mean_absolute_error(y_test, pred_rl))
```

Ilustración 30 Construcción de Modelos Clásicos, Random Forest

```
# Random Forest
model_rf = RandomForestRegressor(n_estimators=100, random_state=42)
model_rf.fit(X_train, y_train.flatten())
pred_rf = model_rf.predict(X_test)
df_pronosticos.loc[farmacia_df.index[-len(X_test):], 'valorventa_rf'] = pred_rf.flatten()

# Calcular Durbin-Watson para Random Forest en el conjunto de prueba
residuos_rf = y_test.flatten() - pred_rf.flatten()
dw_stat_rf = durbin_watson(residuos_rf)
dw_stats_rf.append(dw_stat_rf)
mse_rf.append(mean_squared_error(y_test, pred_rf))
mae_rf.append(mean_absolute_error(y_test, pred_rf))
```

Ilustración 44 Construcción de Modelos Clásicos, Holt Winters

```
# Holt-Winters
try:
    if len(y_train) >= 24:  # Asegurar que hay suficientes datos para ciclos estacionales
        model_hw = ExponentialSmoothing(y_train, seasonal='add', seasonal_periods=12).fit()
    else:
        model_hw = SimpleExpSmoothing(y_train).fit()

    pred_hw = model_hw.predict(start=len(y_train), end=len(y_train) + len(y_test) - 1)
    df_pronosticos.loc[farmacia_df.index[-len(y_test):], 'valorventa_holt_winters'] = pred_hw.flatten()

    # Calcular Durbin-Watson para Holt-Winters en el conjunto de prueba
    residuos_hw = y_test.flatten() - pred_hw.flatten()
    dw_stat_hw = durbin_watson(residuos_hw)
    dw_stats_hw.append(dw_stat_hw)
    mse_hw.append(mean_squared_error(y_test, pred_hw))
    mae_hw.append(mean_absolute_error(y_test, pred_hw))
except ValueError as e:
    print(f"Error en Holt-Winters para farmacia {codfarmacia}: {e}")
```

Ilustración 31 Construcción de Modelos Clásicos, SVR

```
# SVR
model_svr = SVR()
model_svr.fit(X_train, y_train.flatten())
pred_svr = model_svr.predict(X_test)
df_pronosticos.loc[farmacia_df.index[-len(X_test):], 'valorventa_svr'] = pred_svr.flatten()

# Calcular Durbin-Watson para SVR en el conjunto de prueba
residuos_svr = y_test.flatten() - pred_svr.flatten()
dw_stat_svr = durbin_watson(residuos_svr)
dw_stats_svr.append(dw_stat_svr)
mse_svr.append(mean_squared_error(y_test, pred_svr))
mae_svr.append(mean_absolute_error(y_test, pred_svr))
```

4.4.3.2 Evaluar el Modelado de Pronósticos

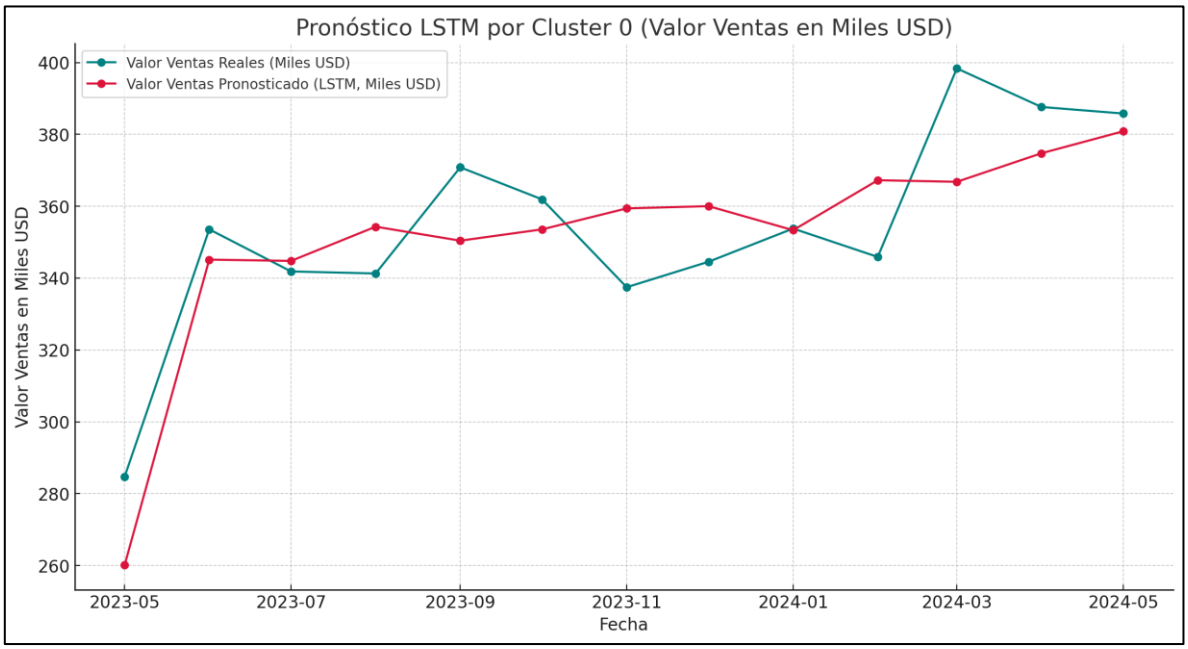
La evaluación de los modelos de pronóstico se ha realizado utilizando métricas clave como el Error Cuadrático Medio (MSE), el Error Absoluto Medio (MAE), además de precisión adaptativa. Estas métricas permiten una comprensión detallada del rendimiento de cada modelo en la predicción de ventas en los clústers 0, 1 y 2 pertenecientes a las tiendas de retail farmacéutico.

4.4.3.2.1 LSTM

El modelo LSTM, conocido por su habilidad para captar patrones temporales complejos, mostró un desempeño variado en los diferentes clústeres de farmacias. En el clúster 0, registró un MSE de 21,388,108 y un MAE de 3,409.62, alcanzando una precisión del 60.04%. Aunque estos resultados indican que el modelo maneja bien las dinámicas temporales, aún muestra un margen de error considerable en este clúster, que incluye 9 tiendas de retail con una participación atípica en ventas ATC1.

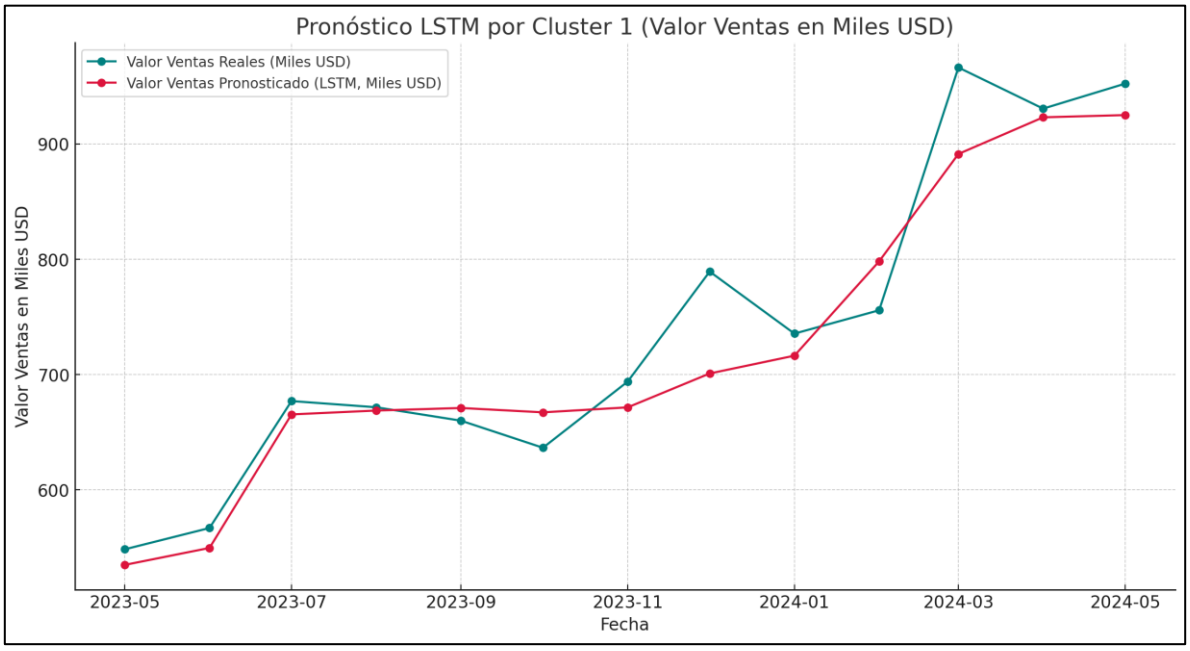
La ilustración 46 nos ayuda a percibir las algunas desviaciones notables en el clúster 0 conformado por farmacias atípicas. El modelo no capturó toda la variabilidad para este grupo de farmacias.

Ilustración 32 Ventas pronosticadas y reales para Clúster 0. Modelo LSTM



En el clúster 1, el modelo LSTM mejoró su rendimiento, logrando un MSE de 19,083,484, un MAE de 3,031.93 y una precisión del 71.30%. Apoyados en la ilustración 47 notamos que las predicciones siguieron de cerca la tendencia general de las ventas reales, especialmente entre mayo y noviembre de 2023, donde las diferencias se aprecian fueron mínimas. Aunque presenta desviaciones menores, el modelo captura correctamente los picos y valles de las ventas. A partir de enero de 2024, denota precisión es los aumentos significativos, y hacia mayo de 2024, las predicciones convergieron casi perfectamente con los valores reales, mostrando un rendimiento sólido y consistente.

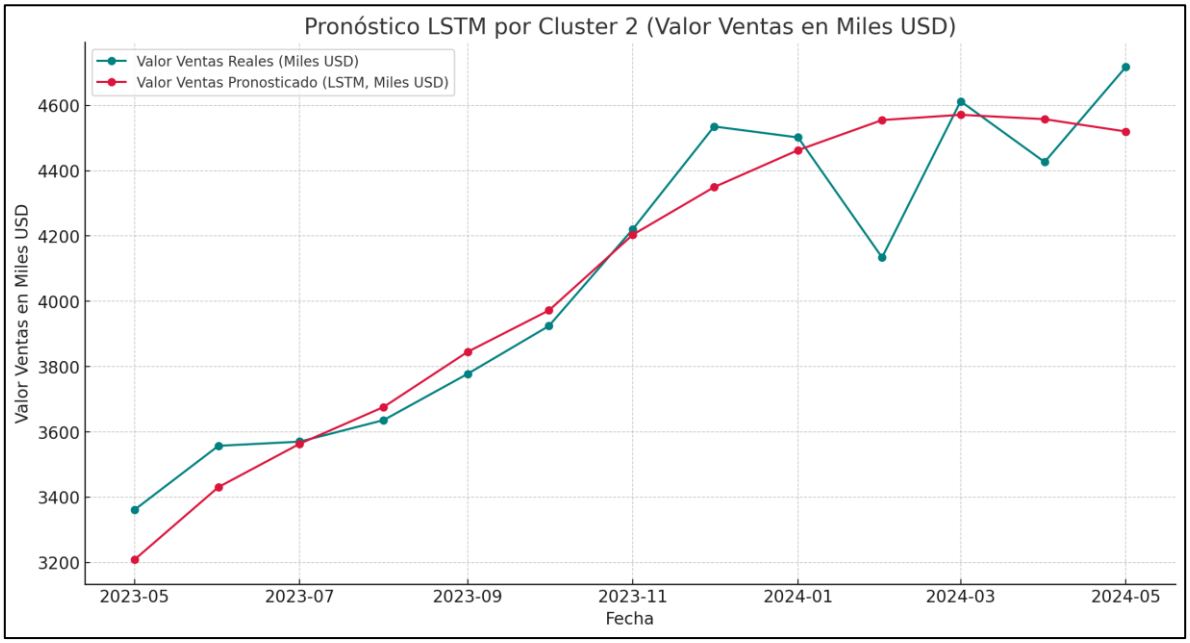
Ilustración 33 Ventas pronosticadas y reales para Clúster 1. Modelo LSTM



En el clúster 2, que agrupa la mayoría de las tiendas de retail, el modelo LSTM logró un desempeño destacado, con un MSE de 10,376,866, un MAE de 2,314.68 y una precisión del 77.20%. La ilustración 48 refleja una alineación bastante cercana entre

las predicciones del modelo y las ventas reales, lo que indica que LSTM manejó efectivamente los patrones temporales y las fluctuaciones en las ventas. A lo largo del tiempo, las predicciones siguieron de cerca las tendencias de crecimiento, con mínimas desviaciones. Esta capacidad para predecir con precisión en el clúster más representativo refuerza la confiabilidad de LSTM como un modelo sólido para pronosticar ventas en entornos de retail complejos.

Ilustración 48 Ventas pronosticadas y reales para Clúster 2. Modelo LSTM

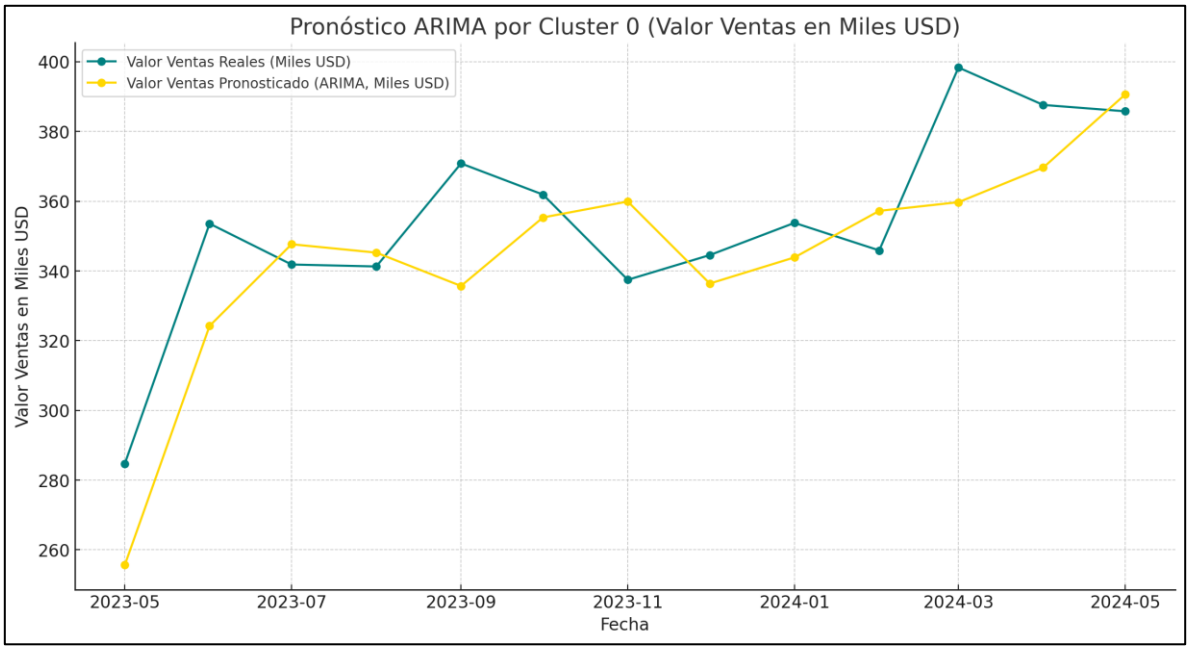


4.4.3.2.2 ARIMA

El modelo ARIMA mostró un rendimiento aceptable en el clúster 0, registrando un MSE de 26,220,412, un MAE de 3,531.11 y una precisión del 57.92%. En la ilustración 49, se observa que ARIMA sigue de manera moderada las tendencias de las ventas reales, aunque con algunas desviaciones notables. A pesar de que el modelo

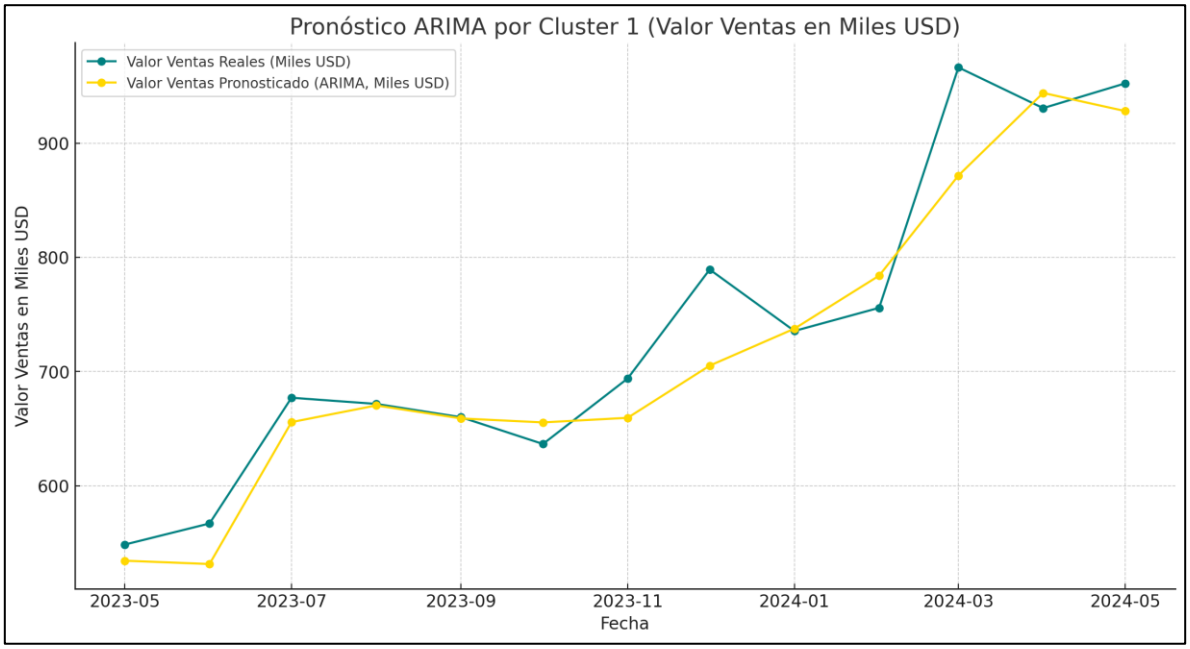
logró captar las tendencias generales, las diferencias en ciertos picos y valles reflejan que ARIMA no maneja completamente la variabilidad en las ventas.

Ilustración 49 Ventas pronosticadas y reales para Clúster 0. Modelo ARIMA



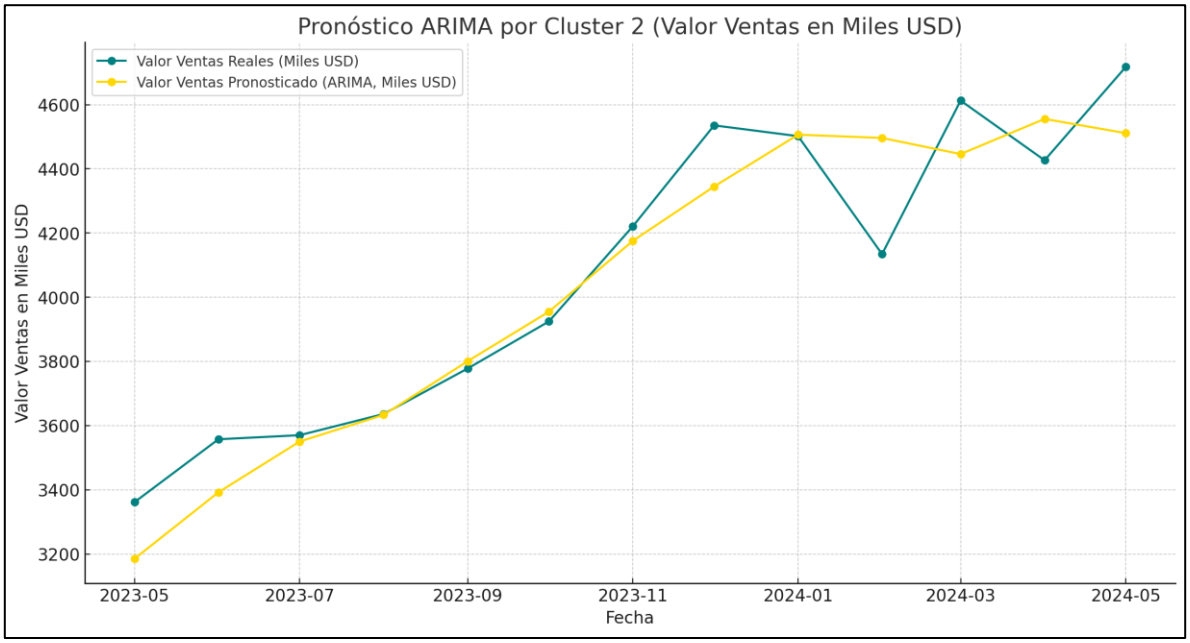
En el clúster 1, el modelo ARIMA mostró un mejor rendimiento, alcanzando un MSE de 23,429,338, un MAE de 3,062.64 y una precisión del 71.92%. En la ilustración 50 refleja que el modelo sigue de manera más precisa las tendencias generales de las ventas reales, con pocas desviaciones. A lo largo del tiempo, ARIMA capturó de manera efectiva los patrones de crecimiento y las fluctuaciones, aunque en algunos puntos, especialmente en momentos de mayor crecimiento, las predicciones tienden a ser más conservadoras que las ventas reales. No obstante, el modelo logra converger correctamente hacia el final del período, lo que indica que ARIMA es más adecuado para este clúster que para otros en términos de predicción de ventas.

Ilustración 34 Ventas pronosticadas y reales para Clúster 1. Modelo ARIMA



En el clúster 2, el modelo ARIMA registró un MSE de 14,709,651, un MAE de 2,620.08 y una precisión del 73.99%. La ilustración 51 muestra que ARIMA fue capaz de seguir de manera efectiva las tendencias generales de las ventas reales, capturando correctamente los patrones de crecimiento a lo largo del tiempo. Sin embargo, en momentos clave, como en los picos y valles de las ventas, el modelo predijo de manera más conservadora, con una tendencia a subestimar los valores reales. Aunque ARIMA logró un buen rendimiento general en este clúster, no alcanza el nivel de precisión observado con el modelo LSTM, especialmente en la captura de las fluctuaciones más dinámicas.

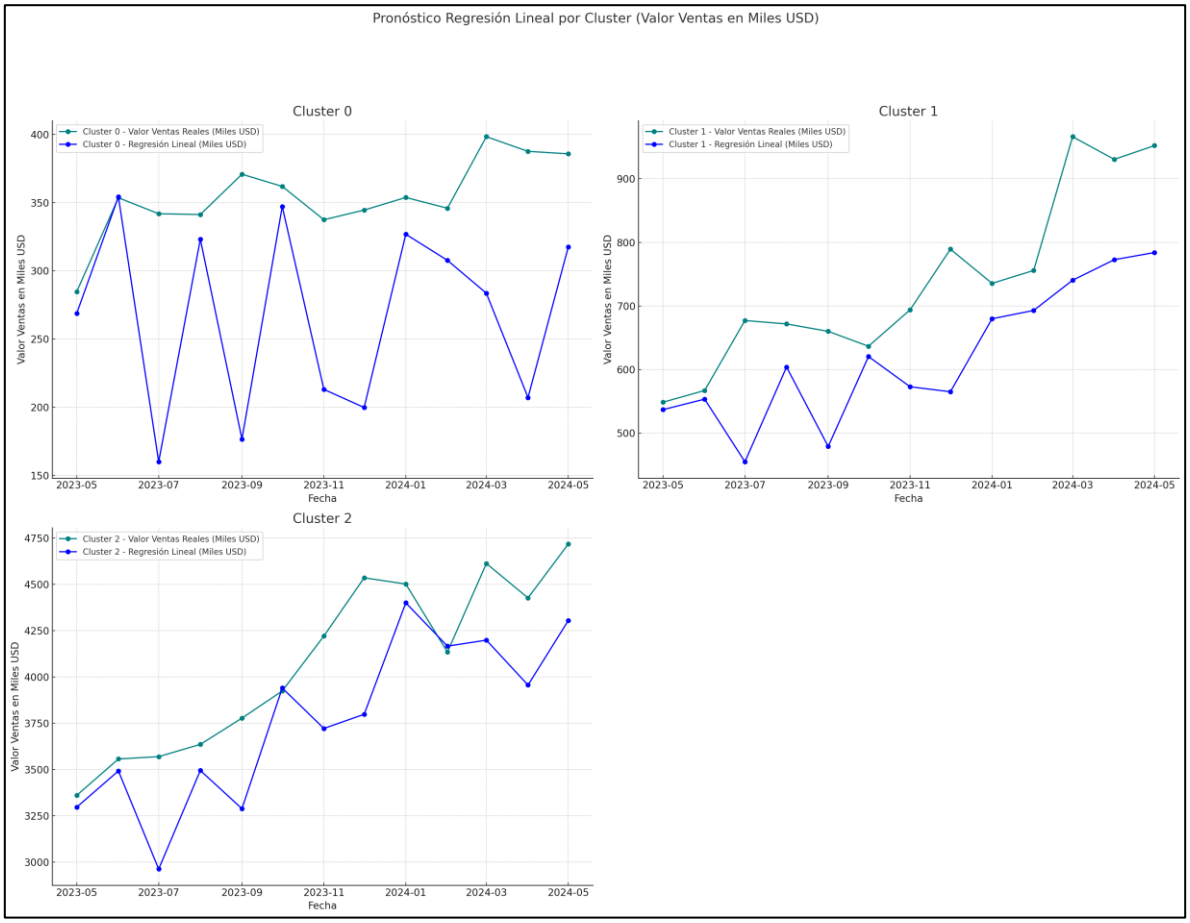
Ilustración 35 Ventas pronosticadas y reales para Clúster 2. Modelo ARIMA



4.4.3.2.3 Regresión Lineal

El modelo de Regresión Lineal no tuvo un buen rendimiento en los distintos clústeres. En el clúster 0, se observó un MSE de 316,663,458 y un MAE de 11,249.38, con una precisión del 30.56%. Lo que indica errores significativos en las predicciones. En el clúster 1, aunque la precisión mejoró a 44.46%, el MSE fue de 232,502,309 y el MAE de 8,652.52, mostrando que los errores seguían siendo elevados. En el clúster 2, el modelo registró un MSE de 81,805,034 y un MAE de 5,900.85, con una precisión del 45.37%. Estos resultados sugieren que la Regresión Lineal tiene dificultades para captar la complejidad de las ventas en este contexto, limitando su efectividad como herramienta de pronóstico en el sector farmacéutico. Los picos y valles no captados, tendencias de ventas son algunas de sus debilidades teóricas de modelo de regresión lineal. Las cuáles las podemos evidenciar visualmente en la ilustración 52.

Ilustración 36 Ventas pronosticadas y reales para los clústers. Modelo Regresión Lineal

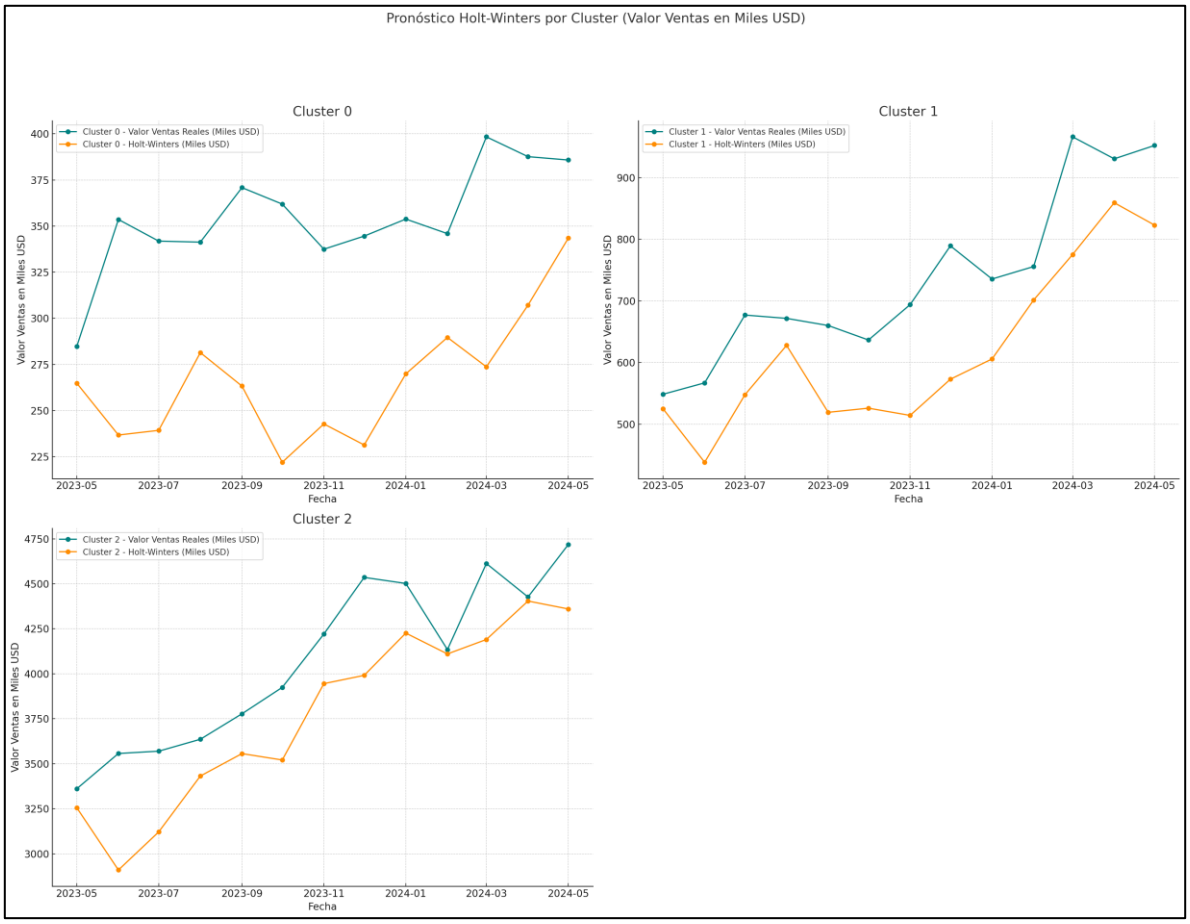


4.4.3.2.4 Holt-Winters

El modelo Holt-Winters, diseñado para capturar patrones estacionales, mostró un rendimiento moderado en todos los clústeres. En el clúster 0, registró un MSE de 244,926,998 y un MAE de 10,651.45, con una precisión del 20.94%. En el clúster 1, la precisión mejoró a 39.08%, con un MSE de 158,649,748 y un MAE de 7,863.58. En el clúster 2, Holt-Winters mostró un MSE de 73,555,620 y un MAE de 5,927.70, alcanzando una precisión del 41.11%. Estos resultados sugieren que Holt-Winters no

es eficaz en capturar patrones temporales para este conjunto de datos, sus diferencias con respecto a la venta real se aprecian en la ilustración 53.

Ilustración 37 Ventas pronosticadas y reales para los clústers. Modelo Holt Winters



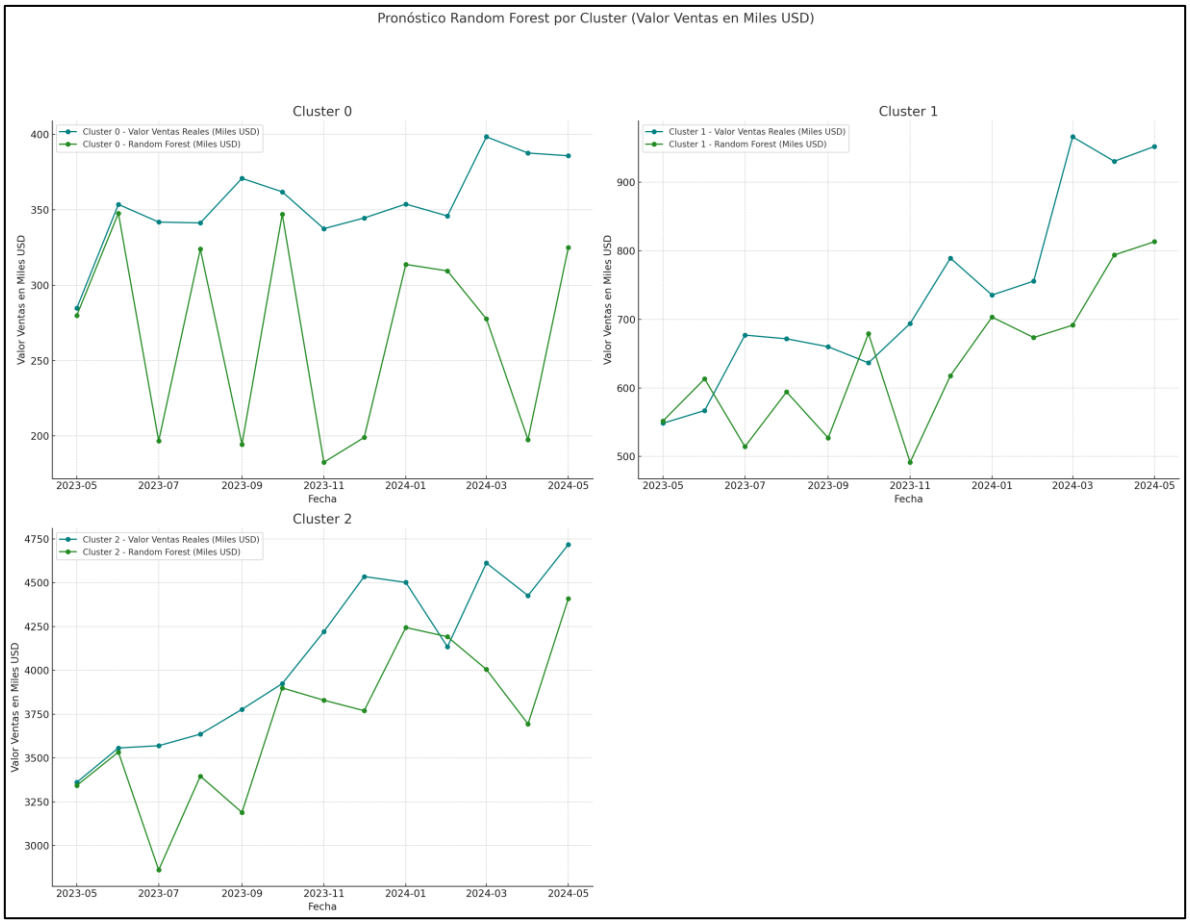
4.4.3.2.5 Random Forest

El modelo Random Forest mostró un rendimiento moderado en la predicción de ventas a lo largo de los diferentes clústeres. En el clúster 0, el modelo registró un MSE de 310,465,639 y un MAE de 11,135.39, con una precisión del 31.84%. La ilustración 54 revela fluctuaciones significativas entre las predicciones y las ventas reales, lo que indica que Random Forest tuvo dificultades para capturar adecuadamente la dinámica

de ventas en este clúster. En el clúster 1, el modelo mejoró su rendimiento con un MSE de 229,018,153 y un MAE de 8,920.09, alcanzando una precisión del 39.51%. Aunque hubo cierta mejora, las predicciones siguen mostrando grandes desviaciones en comparación con las ventas reales, como se observa en la ilustración.

En el clúster 2, Random Forest mostró un MSE de 93,480,753 y un MAE de 6,288.78, con una precisión del 42.85%. Aunque este clúster mostró una mejor alineación entre las predicciones y las ventas reales, el modelo aún presenta desafíos en la captura de las fluctuaciones más dinámicas, lo que limita su eficacia general.

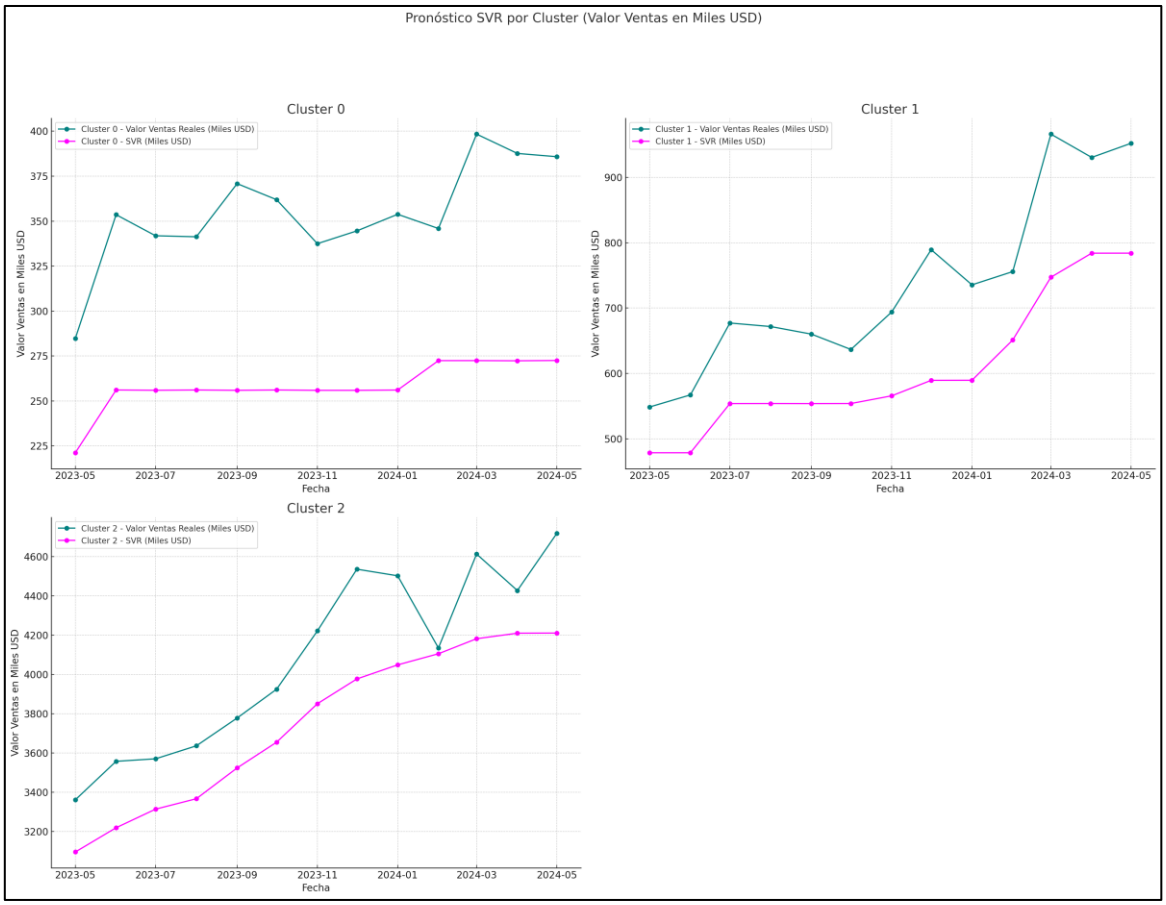
Ilustración 38 Ventas pronosticadas y reales para los clústers. Modelo Random Forest



4.4.3.2.6 Support Vector Regression (SVR)

El modelo Support Vector Regression mostró un desempeño bajo en todos los clústeres. En el clúster 0, registra un MSE de 284,033,249 y un MAE de 11,634.91, con una precisión del 16.67%. En el clúster 1, con un MSE de 162,781,119 y un MAE de 8,469.62, alcanzando una precisión del 34.82%. La ilustración 55 muestran las predicciones con una tendencia plana. Para el clúster 2, MSE de 63,617,158 y MAE de 5,587.21, y una precisión del 42.85%. Aunque hay una ligera mejora en la precisión no captura adecuadamente los picos y las caídas de las ventas, manteniendo una tendencia suavizada que no refleja la complejidad real de los datos.

Ilustración 39 Ventas pronosticadas y reales para los clústers. Modelo SVR



4.4.3.3 Comparación de Modelado de Pronósticos

Al evaluar los diferentes modelos de pronóstico aplicados a los clústeres de farmacias, se observan diferencias significativas en su desempeño, lo que ayuda a identificar el modelo más efectivo para predecir ventas en el entorno del retail farmacéutico.

LSTM sobresale como el modelo más preciso y confiable para la predicción de ventas en todos los clústeres. Con una precisión que varía entre el 60.04% y el 77.20%, y los valores más bajos de Error Cuadrático Medio (MSE) y Error Absoluto Medio (MAE), LSTM demuestra una capacidad superior para capturar patrones temporales complejos. En el clúster 2, que agrupa la mayoría de las tiendas de retail, LSTM alcanzó un MSE de 10,376,866 y un MAE de 2,314.68, con una precisión del 77.20%, subrayando su eficacia en la predicción de ventas en entornos con alta complejidad temporal. Este rendimiento indica que LSTM es especialmente adecuado para escenarios donde las dinámicas de ventas están influenciadas por múltiples factores temporales, consolidándose como el mejor modelo para este contexto.

ARIMA, por su parte, mostró un desempeño considerablemente sólido, especialmente cuando se considera su capacidad para capturar patrones temporales lineales y estacionales. Aunque no alcanzó los niveles de precisión de LSTM, ARIMA logró precisiones que varían entre el 57.92% y el 73.99%, con un MSE y MAE significativamente más bajos que algunos modelos tradicionales como la Regresión Lineal y SVR. En el clúster 2, ARIMA alcanzó un MSE de 14,709,651 y un MAE de 2,620.08, con una precisión del 73.99%, lo que lo convierte en una opción competitiva

para escenarios donde se necesita un balance entre simplicidad y efectividad en la modelización temporal.

Random Forest, aunque mostró un rendimiento aceptable, quedó por detrás de ARIMA y LSTM en términos de precisión y error. Este modelo logró precisiones moderadas, entre el 31.84% y el 42.85%, con MSE y MAE más altos en todos los clústeres. Si bien Random Forest es robusto en el manejo de la variabilidad de los datos, su capacidad para capturar la complejidad temporal es limitada en comparación con ARIMA y LSTM. En contextos menos complejos, Random Forest podría ser una opción válida, pero no alcanza el nivel de precisión necesario en este caso específico.

Holt-Winters, un modelo conocido por su capacidad para capturar patrones estacionales mostró un rendimiento inferior en comparación con ARIMA y LSTM. Su precisión osciló entre el 20.94% y el 41.11%, con errores más altos en comparación con LSTM y ARIMA. Aunque Holt-Winters es útil para capturar patrones estacionales, no logra predecir con precisión en clústeres con dinámicas temporales complejas, como las observadas en las farmacias analizadas.

Regresión Lineal y Support Vector Regression (SVR) fueron los modelos con el rendimiento más bajo en todos los clústeres. Ambos presentaron altos niveles de error (MSE y MAE) y bajas precisiones, con la Regresión Lineal variando entre el 30.56% y el 45.37%, y SVR entre el 16.67% y el 42.85%. Estos modelos no lograron captar

adecuadamente la complejidad de los datos de ventas, lo que los hace menos adecuados para este tipo de predicciones en el sector farmacéutico.

4.4.4 Despliegue

El plan para el despliegue del sistema tendrá en cuenta los resultados de la evaluación del modelado de pronósticos descritos anteriormente, y deberá ser presentado previamente a los stakeholders sugerir su implementación. El objetivo principal de este plan es integrar de manera efectiva los modelos de pronóstico más precisos en el entorno operativo de las farmacias, con el fin de mejorar la predicción de ventas y optimizar la toma de decisiones estratégicas.

Sugerencias para la Implementación del Modelo LSTM

Dado que el modelo LSTM ha demostrado ser el más eficaz en la predicción de ventas, con precisiones que alcanzan hasta un 77% en el clúster 2, se sugiere priorizar su implementación en el sistema. Este modelo debería integrarse en la plataforma de gestión de ventas para permitir el monitoreo y la previsión de las dinámicas de ventas en los diferentes clústeres, facilitando así una planificación más precisa de inventarios y decisiones estratégicas de marketing.

Preparación del Entorno

Recomendación: Establecer un entorno de producción robusto que permita la actualización continua de datos y la recalibración del modelo basado en nuevas tendencias de ventas.

Sugerencia: Ajustar el sistema de gestión existente para incorporar la nueva funcionalidad, asegurando que las predicciones generadas por el modelo LSTM sean accesibles en tiempo real para los equipos responsables de ventas y logística.

Capacitación del Personal

Recomendación: Proporcionar capacitación específica a los usuarios clave, incluidos los responsables de la planificación de inventarios y marketing, sobre cómo interpretar y utilizar las predicciones del modelo LSTM.

Sugerencia: Desarrollar manuales y guías detalladas que expliquen el uso del sistema, con un enfoque en cómo maximizar el valor de las predicciones para la toma de decisiones operativas.

Consideración de Modelos Alternativos:

Aunque se sugiere que el modelo LSTM sea el principal implementado, también se debería considerar la utilización del modelado ARIMA que ha mostrado un rendimiento aceptable. Este modelo puede servir como referencia en escenarios menos complejos o para análisis comparativos en situaciones donde la captura de patrones estacionales sea relevante.

Monitoreo y Ajustes Continuos:

Sugerencia: Implementar un sistema de monitoreo para evaluar la efectividad del modelo LSTM en comparación con otros modelos, realizando revisiones periódicas del

rendimiento del modelo y ajustes necesarios para mantener o mejorar la precisión de las predicciones.

Soporte Técnico y Actualizaciones

Recomendación: Designar un equipo de soporte técnico dedicado a asistir en la resolución de problemas relacionados con la implementación de los modelos y a realizar actualizaciones cuando sea necesario. Este equipo también debería encargarse de mejorar el modelo LSTM y otros modelos conforme se identifiquen nuevas necesidades o se disponga de más datos.

5 Conclusiones y Recomendaciones

5.1 Conclusiones

1. **Eficiencia y Precisión del Modelo LSTM en la Predicción de Ventas:** El modelo de Redes Neuronales de Memoria a Largo Plazo (LSTM, por sus siglas en inglés) ha demostrado ser una herramienta superior en la predicción de ventas para el sector retail farmacéutico. Su capacidad para captar la dinámica temporal y secuencial de los datos, que es crucial en este contexto, le ha permitido superar a otros modelos predictivos tradicionales, como el Random Forest y Holt-Winters. Esto ha sido particularmente evidente en la habilidad del LSTM para manejar tanto patrones de ventas cíclicos como eventuales, algo que los modelos convencionales suelen no captar de manera efectiva. La implementación de este modelo, por tanto, no solo mejora la exactitud de las previsiones, sino que también contribuye a una gestión más efectiva del inventario y a la optimización de los recursos en un mercado altamente competitivo y volátil como es el farmacéutico.
2. **Clustering como Estrategia para la Personalización de Predicciones:** La segmentación de puntos de venta mediante técnicas de clustering, específicamente con el algoritmo K-means, ha permitido crear grupos homogéneos de puntos de venta que comparten características similares. Este enfoque ha sido fundamental para personalizar las predicciones de ventas, adaptando los modelos predictivos a las particularidades de cada segmento.

Esto ha resultado en una mejora significativa en la precisión de las predicciones, evidenciando que una estrategia de segmentación bien implementada puede ser altamente eficaz en la gestión de inventarios y en la toma de decisiones operativas. Este hallazgo subraya la importancia de la segmentación de mercados en la planificación estratégica y en la optimización de recursos dentro del sector retail farmacéutico.

3. **Importancia de la Calidad de los Datos para la Precisión Predictiva:** La precisión de cualquier modelo predictivo depende en gran medida de la calidad y consistencia de los datos utilizados. Este estudio ha demostrado que un preprocesamiento riguroso de los datos, que incluye la gestión adecuada de valores faltantes, normalización de variables y la integración coherente de diversas fuentes de información, es esencial para el éxito de los modelos de aprendizaje automático. Los resultados obtenidos refuerzan la necesidad de mantener altos estándares en la recolección y procesamiento de datos, asegurando que cualquier análisis o predicción esté basado en información precisa y actualizada. La robustez del modelo LSTM, en este sentido, se ve potenciada cuando opera sobre un conjunto de datos bien estructurado y libre de inconsistencias.
4. **Limitaciones de los Modelos Predictivos Tradicionales:** Los enfoques tradicionales para la predicción de ventas, como la regresión lineal o el modelo de Holt-Winters, han mostrado limitaciones significativas en su capacidad para capturar la complejidad inherente a los datos de ventas en el sector

farmacéutico. Estos modelos, si bien pueden ser útiles en escenarios menos complejos o en series temporales con patrones más claros y menos fluctuantes, no logran captar de manera adecuada la variabilidad y los cambios abruptos en las tendencias de ventas. Este hallazgo refuerza la necesidad de adoptar enfoques más avanzados y flexibles, como las redes neuronales LSTM, que están mejor equipados para manejar la naturaleza dinámica y multifacética de los datos en este sector.

5.2 Recomendaciones

1. **Implementación del Modelo LSTM como Herramienta Principal:** Se recomienda la adopción del modelo LSTM como la herramienta principal para la predicción de ventas en el sistema de gestión de la empresa. Su capacidad para capturar patrones complejos de ventas, incluyendo estacionalidades y tendencias emergentes, lo convierte en una opción ideal para la planificación de inventarios y la estrategia comercial. Este modelo debe ser integrado de manera que permita un seguimiento continuo de las predicciones, proporcionando así una base sólida para la toma de decisiones en tiempo real.
2. **Capacitación y Desarrollo de Competencias del Personal:** Para maximizar el valor de las predicciones generadas por el modelo LSTM, es esencial que el personal encargado de la planificación de inventarios y la toma de decisiones estratégicas reciba una capacitación adecuada. Esta formación debe enfocarse no solo en la interpretación de los resultados, sino también en cómo utilizar

estos insights para optimizar operaciones y mejorar el servicio al cliente. Desarrollar manuales operativos y guías prácticas permitirá que el equipo esté bien equipado para integrar estos nuevos procesos en su rutina diaria.

3. **Monitoreo y Evaluación Continua del Desempeño del Modelo:** Dado que las condiciones del mercado y los patrones de consumo pueden cambiar con el tiempo, se recomienda establecer un sistema de monitoreo continuo para evaluar el desempeño del modelo LSTM. Esto incluye realizar revisiones periódicas y ajustar los parámetros del modelo según sea necesario para mantener su efectividad y precisión. Un enfoque de retroalimentación constante permitirá que el modelo evolucione en línea con las necesidades del negocio y los cambios en el entorno del mercado.
4. **Exploración de Modelos Complementarios en Escenarios Específicos:** Aunque el modelo LSTM ha mostrado ser el más efectivo en general, se sugiere no descartar completamente el uso de modelos tradicionales en ciertos escenarios específicos. Por ejemplo, en análisis comparativos o en situaciones donde los patrones estacionales sean más evidentes, modelos como Random Forest o Holt-Winters podrían ofrecer perspectivas valiosas. La combinación de diferentes enfoques puede enriquecer el proceso de toma de decisiones y proporcionar un respaldo adicional en el análisis de ventas.
5. **Creación de un Entorno de Producción Resiliente y Flexible:** Para garantizar que el modelo LSTM y otros modelos predictivos puedan operar de manera efectiva a largo plazo, se recomienda establecer un entorno de producción que

permita la actualización continua de datos y la recalibración del modelo. Esto incluye la implementación de procesos automáticos para la recolección de datos, así como mecanismos que permitan la fácil integración de nuevas fuentes de información. Un entorno de producción bien estructurado asegurará que las predicciones sigan siendo relevantes y ajustadas a las realidades del mercado.

6 Bibliografía

Aggarwal, C., & Reddy, C. (2016). *Data Clustering Algorithms and Applications*. New York: Chapman and Hall/CRC.

Atance, C. (25 de 04 de 2022). IQVIA: tendencias del mercado en Latam. *PHARMABIZ*. Obtenido de https://www.pharmabiz.net/iqvia-tendencias-del-mercado-en-latam/

Babai, M., & Nikolopoulos, K. (2013). Improving the performance of conditional comoment models for dinancial returns forecasting. *International Journal of Forecasting*, 644-654.

Berkhin, P. (2020). *A Survey of Clustering Data Mining Techniques*. Berlín: Springer.

Box, G., Jenkins, G., & Gregory, R. (2015). *Time Series Analysis*. John Wiley & Sons, Inc.

Brownlee, J. (2017). *Deep Learning for Time-Series Analysis*. Machine Learning Mastery.

Brownlee, J. (2018). *Deep Learning for Time Series Forecasting*. Machine Learning Mastery.

Fischer, T., & Krauss, C. (2018). Deep learning whit long short-term memory networks for financial market predictions. *Europen Jpurnal of Operational Research*, 654-669.

Gallardo, J. (2009). *Introducción al Análisis Cluster*. Granada.

Hyndman, R., & Athanasopoulos, G. (2018). *Forecasting: Principles and Practice*. Melbourne Australia: OTexts.

IQVIA. (2023). Latin America Market Review & Trends.

John, J., Shobayo, O., & Ogunleye, B. (2023). An Exploration of Clustering Algorithms for Customer Segmentation in the UK Retail Market. *analytics*, 809-823.

Luxburg, U. (2018). *A Tutorial on Spectral Clustering*. Tübingen-Germany: Springer.

Reynolds, D. (2020). *Gaussian Mixture Models: Recent Advances*. *ACM Computing Surveys*.

Van Engelen, J., & Hoos, H. (2019). *Machine Learning*. Springer.

Wang, Y., & Gunasekaran, A. (2019). Big Data Analytics in Logistics and Supply Chain Management: Certain Investigations of research and applications. *International Journal of Production Economics*, 278-291.

7 Anexos

7.1 Anexos Consultas SQL

```
registros_fc = run_query("""SELECT COUNT(*)

                FROM dwh.farmacomercial""")

temporalidad_fc = run_query("""

                SELECT min(fechafactura), max(fechafactura)

                FROM dwh.farmacomercial""")

# Ejecutar la consulta SQL para contar valores nulos y en blanco

nulos_blancos_fc = run_query("""

  SELECT

    SUM(CASE WHEN codfarmacia IS NULL OR TRIM(CAST(codfarmacia AS STRING))
= '' THEN 1 ELSE 0 END) AS null_blank_codfarmacia,

    SUM(CASE WHEN aniomes IS NULL OR TRIM(CAST(aniomes AS STRING)) = '' THEN
1 ELSE 0 END) AS null_blank_aniomes,

    SUM(CASE WHEN valorventa IS NULL OR TRIM(CAST(valorventa AS STRING)) = ''
THEN 1 ELSE 0 END) AS null_blank_valorventa

  FROM dwh.farmacomercial;""")

# Mostrar los resultados
```

```
print(nulos_blancos_fc)

#%% Describir Farmafarmacia

registros_ff = run_query("""SELECT COUNT(*)

                    FROM dwh.farmafarmacia""")

registros_ff

# Ejecutar la consulta SQL para contar valores nulos y en blanco

nulos_blancos_ff = run_query("""

  SELECT

    SUM(CASE WHEN farmacia IS NULL OR TRIM(CAST(farmacia AS STRING))

      = '' THEN 1 ELSE 0 END) AS farmacia,

    SUM(CASE WHEN sucursal IS NULL OR TRIM(CAST(sucursal AS STRING))

      = '' THEN 1 ELSE 0 END) AS sucursal,

    SUM(CASE WHEN tipofarmacia IS NULL OR TRIM(CAST(tipofarmacia AS STRING))

      = '' THEN 1 ELSE 0 END) AS tipofarmacia,

    SUM(CASE WHEN region IS NULL OR TRIM(CAST(region AS STRING))

      = '' THEN 1 ELSE 0 END) AS region,

    SUM(CASE WHEN provincia IS NULL OR TRIM(CAST(provincia AS STRING))

      = '' THEN 1 ELSE 0 END) AS provincia,
```

```
    SUM(CASE WHEN canton IS NULL OR TRIM(CAST(canton AS STRING))
      = '' THEN 1 ELSE 0 END) AS canton,
    SUM(CASE WHEN latitud IS NULL OR TRIM(CAST(latitud AS STRING))
      = '' THEN 1 ELSE 0 END) AS latitud,
    SUM(CASE WHEN longitud IS NULL OR TRIM(CAST(longitud AS STRING))
      = '' THEN 1 ELSE 0 END) AS longitud,
    SUM(CASE WHEN minfechafactura IS NULL OR TRIM(CAST(minfechafactura AS
STRING))
      = '' THEN 1 ELSE 0 END) AS minfechafactura,
    SUM(CASE WHEN estado IS NULL OR TRIM(CAST(estado AS STRING))
      = '' THEN 1 ELSE 0 END) AS estado
  FROM dwh.farmafarmacia""")
# Mostrar los resultados
print(nulos_blancos_ff)
#%% Describir farmaproductos
registros_fp = run_query("""SELECT COUNT(*)
                FROM dwh.farmaproductos""")
registros_fp
```

```
# Ejecutar la consulta SQL para contar valores nulos y en blanco
nulos_blancos_ff = run_query("""
  SELECT
    SUM(CASE WHEN farmacia IS NULL OR TRIM(CAST(farmacia AS STRING))
      = '' THEN 1 ELSE 0 END) AS farmacia,
    SUM(CASE WHEN sucursal IS NULL OR TRIM(CAST(sucursal AS STRING))
      = '' THEN 1 ELSE 0 END) AS sucursal,
    SUM(CASE WHEN tipofarmacia IS NULL OR TRIM(CAST(tipofarmacia AS STRING))
      = '' THEN 1 ELSE 0 END) AS tipofarmacia,
    SUM(CASE WHEN region IS NULL OR TRIM(CAST(region AS STRING))
      = '' THEN 1 ELSE 0 END) AS region,
    SUM(CASE WHEN provincia IS NULL OR TRIM(CAST(provincia AS STRING))
      = '' THEN 1 ELSE 0 END) AS provincia,
    SUM(CASE WHEN canton IS NULL OR TRIM(CAST(canton AS STRING))
      = '' THEN 1 ELSE 0 END) AS canton,
    SUM(CASE WHEN latitud IS NULL OR TRIM(CAST(latitud AS STRING))
      = '' THEN 1 ELSE 0 END) AS latitud,
    SUM(CASE WHEN longitud IS NULL OR TRIM(CAST(longitud AS STRING))
```

```
        = '' THEN 1 ELSE 0 END) AS longitud,

    SUM(CASE WHEN minfechafactura IS NULL OR TRIM(CAST(minfechafactura AS
STRING))

        = '' THEN 1 ELSE 0 END) AS minfechafactura,

    SUM(CASE WHEN estado IS NULL OR TRIM(CAST(estado AS STRING))

        = '' THEN 1 ELSE 0 END) AS estado

  FROM dwh.farmafarmacia""")

# Mostrar los resultados

print(nulos_blancos_ff)

#%% Selección de la data

# Información desde enero 2019

df=run_query("""

        SELECT fc.codfarmacia, ff.farmacia, ff.sucursal, ff.tipofarmacia,

        ff.region, ff.provincia, ff.canton, ff.latitud, ff.longitud,

        ff.minfechafactura, fc.aniomes, ROUND(sum(fc.valorventa),0) AS valorventa

        FROM dwh.farmacomercial AS fc

        INNER JOIN dwh.farmafarmacia as ff ON ff.codfarmacia = fc.codfarmacia

        WHERE fc.aniomes BETWEEN 201901 AND 202406
```

```
        AND ff.sucursal IN ('*****')
        AND ff.estado = 'ACTIVO'
        GROUP BY fc.codfarmacia, ff.farmacia, ff.sucursal, ff.tipofarmacia,
        ff.region, ff.provincia, ff.canton, ff.latitud, ff.longitud,
        ff.minfechafactura, fc.aniomes""")
df_pdv_cluster = run_query("""
        SELECT fc.codfarmacia, ff.farmacia, ff.sucursal, ff.tipofarmacia,
        ff.region, ff.provincia, ff.canton, ff.latitud, ff.longitud,
        ff.minfechafactura, fc.codproducto, fp.nombreproducto,
        ROUND((SUM(fc.valorventa)/6),2) AS venta_promedio
        FROM dwh.farmacomercial AS fc
        INNER JOIN dwh.farmafarmacia as ff ON ff.codfarmacia = fc.codfarmacia
        INNER JOIN dwh.farmaproductos as fp ON fp.codproducto = fc.codproducto
        WHERE fc.aniomes BETWEEN 202401 AND 202406
        AND ff.sucursal IN ('*****')
        AND ff.estado = 'ACTIVO'
        GROUP BY fc.codfarmacia, ff.farmacia, ff.sucursal, ff.tipofarmacia,
        ff.region, ff.provincia, ff.canton, ff.latitud, ff.longitud,
```

```
        ff.minfechafactura, fc.codproducto, fp.nombreproducto

        """)

df_pdv_cluster_atc1 = pd.merge(df_pdv_cluster, df_atc1, on=('codproducto'), how='inner')

df_pdv_cluster_atc1 = pd.pivot_table(df_pdv_cluster_atc1, values='venta_promedio',index=['codfarmacia', 'farmacia', 'sucursal', 'tipofarmacia', 'region',

    'provincia', 'canton', 'latitud', 'longitud', 'minfechafactura'], columns='atc_1', aggfunc='sum').fillna(0)

df_pdv_cluster_atc1 = df_pdv_cluster_atc1.reset_index()

# fecha de apertura anteriores a junio 2023, revisar si 12 meses de información son suficientes

df_pdv_cluster_atc1 = df_pdv_cluster_atc1[df_pdv_cluster_atc1['minfechafactura']<='2023-06-01 00:00:00']

#%% Crear variable meses desde la apertura

from datetime import datetime

df_pdv_cluster_atc1['minfechafactura'] = pd.to_datetime(df_pdv_cluster_atc1['minfechafactura'])

# Definir la fecha de corte
```

```
fecha_corte = datetime(2024, 6, 30)

# Calcular la diferencia en meses

def diferencia_meses(fecha1, fecha2):

  return (fecha1.year - fecha2.year) * 12 + fecha1.month - fecha2.month

df_pdv_cluster_atc1['meses_desde_apertura'] =

df_pdv_cluster_atc1['minfechafactura'].apply(lambda x:

diferencia_meses(fecha_corte, x))

# Calcular la suma total de ventas por farmacia

df_pdv_cluster_atc1['total_venta_ATC'] = df_pdv_cluster_atc1.iloc[:,

10:23].sum(axis=1)

# Calcular el porcentaje de cada categoría ATC1 respecto al total de la venta de la

farmacia

for column in df_pvd_cluster_atc1.columns[10:23]:

  df_pdv_cluster_atc1[f'{column}_Part'] = (df_pdv_cluster_atc1[column] /

df_pdv_cluster_atc1['total_venta_ATC']) * 100
```

7.2 Anexo Modelado Clúster

```
#%% K-means

# Crear una copia del DataFrame y aplicar dummies
```

```
df = df_pdv_cluster_atc1.copy()

df = pd.get_dummies(df, columns=['tipofarmacia'], drop_first=True)

df.columns

# Seleccionar las variables

variables = ['A - APARATO DIGEST.Y METABOL_Part',

        'B - SANGRE Y ORGANOS HEMATOP_Part',

        'C - APARATO CARDIOVASCULAR_Part',

        'D - DERMATOLOGICOS_Part',

        'G - PROD.GENITO URINARIOS_Part',

        'H - HORMONAS_Part',

        'J - ANTIINFECCIOSOS VIA GENE_Part',

        'L - ANTINEOPLAS Y AGENT INMUN_Part',

        'M - APARATO LOCOMOTOR_Part',

        'N - SISTEMA NERVIOSO_Part',

        'P - ANTIPARASITARIOS_Part',

        'R - APARATO RESPIRATORIO_Part',

        'S - ORGANOS DE LOS SENTIDOS_Part']

df_selected = df[variables]
```

```
# No es necesario normalizar los datos de participación porcentual ya que:

# Todos los valores están en un rango consistente y comparable.

# La suma de las participaciones ya está controlada para cada farmacia.

# Los porcentajes proporcionan una interpretación directa y útil.

#scaler = StandardScaler()

#df_normalized = scaler.fit_transform(df_selected)

#df_normalized= df_selected.copy()

# Aplicar el método del codo (Elbow Method)

wcss = []

for i in range(1, 11):

  kmeans = KMeans(n_clusters=i, init='k-means++', max_iter=300, n_init=10,
random_state=42)

  kmeans.fit(df_selected)

  wcss.append(kmeans.inertia_)

wcss

# Graficar el WCSS en función del número de clusters

plt.figure(figsize=(10, 6))

plt.plot(range(1, 11), wcss, marker='o', linestyle='--')
```

```
plt.xlabel('Número de clusters')

plt.ylabel('WCSS')

plt.title('Método del Codo (Elbow Method)')

plt.xticks(range(1, 11))

plt.grid(True)

plt.show()

# El gráfico del Método del Codo muestra cómo el WCSS (Within-Cluster Sum of Squares)

# Aplicar KMeans

kmeans = KMeans(n_clusters=3, random_state=42)  # Ajustar el número de clusters según sea necesario

df_pdv_cluster_atc1['cluster_kmeans'] = kmeans.fit_predict(df_selected)

# Agrupar por cluster y calcular estadísticas descriptivas

estadisticos_kmeans = df_pdv_cluster_atc1.groupby('cluster_kmeans').agg(['mean', 'std', 'min', 'max', 'median'])

# Mostrar las estadísticas por cada cluster

print(estadisticos_kmeans)

# Calcular estadísticas descriptivas para cada cluster
```

```
cluster_stats_kmeans = df_pdv_cluster_atc1.groupby('cluster_kmeans').agg({
    'A - APARATO DIGEST.Y METABOL_Part': ['mean', 'std'],
    'B - SANGRE Y ORGANOS HEMATOP_Part': ['mean', 'std'],
    'C - APARATO CARDIOVASCULAR_Part': ['mean', 'std'],
    'D - DERMATOLOGICOS_Part': ['mean', 'std'],
    'G - PROD.GENITO URINARIOS_Part': ['mean', 'std'],
    'H - HORMONAS_Part': ['mean', 'std'],
    'J - ANTIINFECCIOSOS VIA GENE_Part': ['mean', 'std'],
    'L - ANTINEOPLAS Y AGENT INMUN_Part': ['mean', 'std'],
    'M - APARATO LOCOMOTOR_Part': ['mean', 'std'],
    'N - SISTEMA NERVIOSO_Part': ['mean', 'std'],
    'P - ANTIPARASITARIOS_Part': ['mean', 'std'],
    'R - APARATO RESPIRATORIO_Part': ['mean', 'std'],
    'S - ORGANOS DE LOS SENTIDOS_Part': ['mean', 'std']})
cluster_stats_kmeans = cluster_stats_kmeans.stack(level=0).reset_index()
cluster_stats_kmeans.columns = ['ATC1 Category', 'Cluster', 'Mean', 'Std']
# Ordenar los resultados por Cluster para mejor visualización
```

```
cluster_stats_kmeans = cluster_stats_kmeans.sort_values(by=['Cluster', 'ATC1 Category'])

# Mostrar los resultados

print(cluster_stats_kmeans)

# Evaluar kmeans

from sklearn.metrics import silhouette_score

silhouette_avg = silhouette_score(df_selected, df_pdv_cluster_atc1['cluster_kmeans'])

print(f'Coeficiente de silueta promedio: {silhouette_avg}')

#%%

# Probar diferentes números de clusters

silhouette_scores = []

for n_clusters in range(2, 11):

    kmeans = KMeans(n_clusters=n_clusters, random_state=42)

    cluster_labels = kmeans.fit_predict(df_selected)

    silhouette_avg = silhouette_score(df_selected, cluster_labels)

    silhouette_scores.append(silhouette_avg)

# Graficar el coeficiente de silueta promedio para cada número de clusters
```

```
plt.figure(figsize=(10, 6))

plt.plot(range(2, 11), silhouette_scores, marker='o', linestyle='--')

plt.xlabel('Número de clusters')

plt.ylabel('Coeficiente de silueta promedio')

plt.title('Coeficiente de Silueta Promedio para Diferentes Números de Clusters')

plt.grid(True)

plt.show()

#%% DBSCAN

from sklearn.cluster import DBSCAN

# Aplicación de DBSCAN

# Definir diferentes valores para eps y min_samples

eps_values = [2.0, 2.5, 3.0, 3.5]

min_samples_values = [3, 4]

# Crear una tabla para almacenar los resultados

results = []

for eps in eps_values:

  for min_samples in min_samples_values:

    # Aplicar DBSCAN
```

```
        dbscan = DBSCAN(eps=eps, min_samples=min_samples)
        clusters_dbscan = dbscan.fit_predict(df_selected)
        # Calcular el coeficiente de silueta
        if len(set(clusters_dbscan)) > 1:  # Para evitar errores cuando todos los puntos están en un solo clúster
            silhouette_dbscan = silhouette_score(df_selected, clusters_dbscan)
        else:
            silhouette_dbscan = -1
        # Almacenar los resultados
        results.append((eps, min_samples, silhouette_dbscan))
# Convertir los resultados a un DataFrame para mejor visualización
results_df = pd.DataFrame(results, columns=['eps', 'min_samples', 'silhouette_score'])
print(results_df)
# Mejor combinación de eps y min_samples basada en el coeficiente de silueta
best_eps = results_df.loc[results_df['silhouette_score'].idxmax(), 'eps']
best_min_samples = results_df.loc[results_df['silhouette_score'].idxmax(), 'min_samples']
# Aplicar DBSCAN con la mejor combinación de parámetros
```

```
dbscan = DBSCAN(eps=best_eps, min_samples=best_min_samples)

clusters_dbscan = dbscan.fit_predict(df_selected)

df_pdv_cluster_atc1['cluster_dbscan'] = clusters_dbscan

#

cluster_stats_dbscan = df_pdv_cluster_atc1.groupby('cluster_dbscan').agg({

   'A - APARATO DIGEST.Y METABOL_Part': ['mean', 'std'],

   'B - SANGRE Y ORGANOS HEMATOP_Part': ['mean', 'std'],

   'C - APARATO CARDIOVASCULAR_Part': ['mean', 'std'],

   'D - DERMATOLOGICOS_Part': ['mean', 'std'],

   'G - PROD.GENITO URINARIOS_Part': ['mean', 'std'],

   'H - HORMONAS_Part': ['mean', 'std'],

   'J - ANTIINFECCIOSOS VIA GENE_Part': ['mean', 'std'],

   'L - ANTINEOPLAS Y AGENT INMUN_Part': ['mean', 'std'],

   'M - APARATO LOCOMOTOR_Part': ['mean', 'std'],

   'N - SISTEMA NERVIOSO_Part': ['mean', 'std'],

   'P - ANTIPARASITARIOS_Part': ['mean', 'std'],

   'R - APARATO RESPIRATORIO_Part': ['mean', 'std'],

   'S - ORGANOS DE LOS SENTIDOS_Part': ['mean', 'std']})
```

```
cluster_stats_dbscan = cluster_stats_dbscan.stack(level=0).reset_index()

cluster_stats_dbscan.columns = ['ATC1 Category', 'Cluster', 'Mean', 'Std']

# Ordenar los resultados por Cluster para mejor visualización

cluster_stats_dbscan = cluster_stats_dbscan.sort_values(by=['Cluster', 'ATC1 Category'])

# Calcular y mostrar el coeficiente de silueta para la mejor configuración

silhouette_dbscan = silhouette_score(df_selected, clusters_dbscan)

print(f'Coeficiente de Silueta para DBSCAN (Mejor Configuración): {silhouette_dbscan}')

#%% jerárquico

# Importar librerías necesarias

from scipy.cluster.hierarchy import dendrogram, linkage

from sklearn.cluster import AgglomerativeClustering

# Crear la matriz de enlace

Z = linkage(df_selected, method='ward')

# Dibujar el dendrograma

plt.figure(figsize=(10, 6))
```

```
dendrogram(Z)

plt.title('Dendrograma')

plt.xlabel('Puntos de Datos')

plt.ylabel('Distancia Euclidiana')

plt.show()

# Aplicación de Clustering Jerárquico

hclust = AgglomerativeClustering(n_clusters=3, affinity='euclidean', linkage='ward')

df_pdv_cluster_atc1['cluster_hierarchical'] = hclust.fit_predict(df_selected)

# Calcular estadísticas descriptivas para cada cluster

cluster_stats_hierarchical = df_pdv_cluster_atc1.groupby('cluster_hierarchical').agg({

   'A - APARATO DIGEST.Y METABOL_Part': ['mean', 'std'],

   'B - SANGRE Y ORGANOS HEMATOP_Part': ['mean', 'std'],

   'C - APARATO CARDIOVASCULAR_Part': ['mean', 'std'],

   'D - DERMATOLOGICOS_Part': ['mean', 'std'],

   'G - PROD.GENITO URINARIOS_Part': ['mean', 'std'],

   'H - HORMONAS_Part': ['mean', 'std'],

   'J - ANTIINFECCIOSOS VIA GENE_Part': ['mean', 'std'],

   'L - ANTINEOPLAS Y AGENT INMUN_Part': ['mean', 'std'],
```

```
    'M - APARATO LOCOMOTOR_Part': ['mean', 'std'],
    'N - SISTEMA NERVIOSO_Part': ['mean', 'std'],
    'P - ANTIPARASITARIOS_Part': ['mean', 'std'],
    'R - APARATO RESPIRATORIO_Part': ['mean', 'std'],
    'S - ORGANOS DE LOS SENTIDOS_Part': ['mean', 'std']})
cluster_stats_hierarchical = cluster_stats_hierarchical.stack(level=0).reset_index()
cluster_stats_hierarchical.columns = ['ATC1 Category', 'Cluster', 'Mean', 'Std']
# Ordenar los resultados por Cluster para mejor visualización
cluster_stats_hierarchical = cluster_stats_hierarchical.sort_values(by=['Cluster', 'ATC1 Category'])
#%%
# Calcular el coeficiente de silueta para KMeans
silhouette_kmeans = silhouette_score(df_selected, df_pdv_cluster_atc1['cluster_kmeans'])
print(f'Coeficiente de Silueta para KMeans: {silhouette_kmeans}')
# Calcular el coeficiente de silueta para DBSCAN
silhouette_dbscan = silhouette_score(df_selected, df_pdv_cluster_atc1['cluster_dbscan'])
```

```
print(f'Coeficiente de Silueta para DBSCAN: {silhouette_dbscan}')

# Calcular el coeficiente de silueta para Hierarchical Clustering

silhouette_hierarchical = silhouette_score(df_selected,

df_pdv_cluster_atc1['cluster_hierarchical'])

print(f'Coeficiente de Silueta para Hierarchical Clustering: {silhouette_hierarchical}')

#%%

from sklearn.decomposition import PCA

# Aplicar PCA para reducir la dimensionalidad a 2 componentes principales

pca = PCA(n_components=2)

df_pca = pca.fit_transform(df_selected)

df_pdv_cluster_atc1['PCA1'] = df_pca[:, 0]

df_pdv_cluster_atc1['PCA2'] = df_pca[:, 1]

# Visualizar los clusters KMeans

plt.figure(figsize=(10, 6))

sns.scatterplot(data=df_pdv_cluster_atc1, x='PCA1', y='PCA2', hue='cluster_kmeans',

palette='Set1')

plt.title('Clusters de Farmacias con KMeans')

plt.show()
```

```
# Visualizar los clusters DBSCAN

plt.figure(figsize=(10, 6))

sns.scatterplot(data=df_pdv_cluster_atc1, x='PCA1', y='PCA2', hue='cluster_dbscan',
palette='Set1')

plt.title('Clusters de Farmacias con DBSCAN')

plt.show()

# Visualizar los clusters Hierarchical Clustering

plt.figure(figsize=(10, 6))

sns.scatterplot(data=df_pdv_cluster_atc1, x='PCA1', y='PCA2',
hue='cluster_hierarchical', palette='Set1')

plt.title('Clusters de Farmacias con Hierarchical Clustering')

plt.show()

#%%

import matplotlib.pyplot as plt

from mpl_toolkits.mplot3d import Axes3D

# Aplicar PCA para reducir la dimensionalidad a 3 componentes principales

pca_3d = PCA(n_components=3)

df_pca_3d = pca_3d.fit_transform(df_selected)
```

```
df_pdv_cluster_atc1['PCA1'] = df_pca_3d[:, 0]

df_pdv_cluster_atc1['PCA2'] = df_pca_3d[:, 1]

df_pdv_cluster_atc1['PCA3'] = df_pca_3d[:, 2]

# Crear el gráfico tridimensional

fig = plt.figure(figsize=(10, 8))

ax = fig.add_subplot(111, projection='3d')

colors = ['r', 'g', 'b', 'm']

for cluster in df_pdv_cluster_atc1['cluster_kmeans'].unique():

  subset = df_pdv_cluster_atc1[df_pdv_cluster_atc1['cluster_kmeans'] == cluster]

  ax.scatter(subset['PCA1'], subset['PCA2'], subset['PCA3'], label=f'Cluster {cluster}',
alpha=0.6)

ax.set_xlabel('PCA1')

ax.set_ylabel('PCA2')

ax.set_zlabel('PCA3')

ax.set_title('Clusters de Farmacias en 3D con KMeans')

ax.legend()

plt.show()

#%% GMM
```

```
from sklearn.mixture import GaussianMixture

# Aplicar GMM

gmm = GaussianMixture(n_components=3)  # Ajusta el número de componentes según tu análisis

cluster_labels = gmm.fit_predict(df_selected)

# Evaluar el coeficiente de silueta

silhouette_avg_gmm = silhouette_score(df_selected, cluster_labels)

print(f'Coeficiente de Silueta para GMM: {silhouette_avg_gmm}')

# Añadir los labels al DataFrame

df_pdv_cluster_atc1['cluster_gmm'] = cluster_labels

# Visualizar en 2D utilizando PCA1 y PCA2

plt.figure(figsize=(10, 8))

scatter = plt.scatter(df_pdv_cluster_atc1['PCA1'], df_pdv_cluster_atc1['PCA2'], c=cluster_labels, cmap='viridis', alpha=0.7)

legend1 = plt.legend(*scatter.legend_elements(), title="Clusters GMM")

plt.gca().add_artist(legend1)

plt.title('Visualización 2D de Clusters GMM usando PCA')

plt.xlabel('PCA1')
```

```
plt.ylabel('PCA2')

plt.grid(True)

plt.show()
```

7.3 Anexo Modelado de Pronósticos

```
# Agrupar los productos por 'atc_1'

grouped_data = df_atc1.groupby('atc_1')['codproducto'].apply(list).reset_index()

# Lista para almacenar los resultados

resultados = []

# Agregar tqdm para mostrar el progreso

for index, row in tqdm(grouped_data.iterrows(), total=grouped_data.shape[0],
desc="Procesando grupos"):

  atc_1 = row['atc_1']

  codproducto_list = "','".join(row['codproducto'])

  # Realizar la consulta SQL para cada grupo

  df = run_query(f"""

        SELECT fc.codfarmacia, ff.farmacia, ff.sucursal, ff.tipofarmacia,

        ff.region, ff.provincia, ff.canton, ff.latitud, ff.longitud,

        ff.minfechafactura, fc.aniomes, ROUND(sum(fc.valorventa),0) AS valorventa
```

```
            FROM dwh.farmacomercial AS fc
            INNER JOIN dwh.farmafarmacia as ff ON ff.codfarmacia = fc.codfarmacia
            WHERE fc.aniomes BETWEEN 201901 AND 202405
            AND ff.sucursal IN ('*******')
            AND ff.estado = 'ACTIVO'
            AND fc.codproducto IN ('{codproducto_list}')
            GROUP BY fc.codfarmacia,ff.farmacia, ff.sucursal, ff.tipofarmacia,
            ff.region, ff.provincia, ff.canton, ff.latitud, ff.longitud,
            ff.minfechafactura, fc.aniomes
            """)
        # Agregar el nombre del grupo 'atc_1' al DataFrame resultante
    df['atc_1'] = atc_1
        # Almacenar el resultado en la lista
    resultados.append(df)
# Concatenar todos los resultados en un solo DataFrame
df = pd.concat(resultados, ignore_index=True)
# Filtrar los datos en función de los valores específicos de codfarmacia de la tabla de
cluster
```

```
codfarmacia_list = df_pdv_cluster_atc1['codfarmacia'].unique()

df = df[df['codfarmacia'].isin(codfarmacia_list)]

#%% Cluster

cluster=df_pdv_cluster_atc1[['codfarmacia','cluster_kmeans']]

#%% Merge

df = df.merge(cluster, how = "left")

#%%

df['minfechafactura'] = pd.to_datetime(df['minfechafactura'])

# Crear una nueva variable 'aniomes2' que convierta 'aniomes' en una fecha

df['aniomes2'] = pd.to_datetime(df['aniomes'].astype(str) + '01', format='%Y%m%d')

# Función para calcular la diferencia en meses

def calcular_meses_desde_apertura(row):

  delta = (row['aniomes2'].year - row['minfechafactura'].year) * 12 +
row['aniomes2'].month - row['minfechafactura'].month

  return delta

# Aplicar la función a cada fila del dataframe

df['meses_desde_apertura'] = df.apply(calcular_meses_desde_apertura, axis=1)

df = df.drop(columns=['minfechafactura', 'aniomes2'])
```

```
#%% Agrupamos el df

df= df.groupby(['codfarmacia', 'farmacia', 'sucursal', 'tipofarmacia', 'region',

    'provincia', 'canton', 'latitud', 'longitud', 'aniomes','cluster_kmeans',

    'meses_desde_apertura'],as_index=False)['valorventa'].sum()

# Ordenar el DataFrame por 'aniomes' y luego por 'codfarmacia'

df = df.sort_values(by=['codfarmacia','aniomes'])

df_pronosticos = df.copy()

#%% Limpiar el dataset

# Paso 1: Calcular el promedio de ventas por cada farmacia en df_pronosticos

promedios_ventas = df_pronosticos.groupby('codfarmacia')['valorventa'].mean()

# Paso 2: Iterar sobre cada farmacia y reemplazar valores atípicos en df_pronosticos

for codfarmacia in df_pronosticos['codfarmacia'].unique():

  # Filtrar el DataFrame para la farmacia actual

  farmacia_df = df_pronosticos[df_pronosticos['codfarmacia'] == codfarmacia]

  # Calcular IQR para la farmacia actual

  Q1 = farmacia_df['valorventa'].quantile(0.25)

  Q3 = farmacia_df['valorventa'].quantile(0.75)

  IQR = Q3 - Q1
```

```
    lower_bound = Q1 - 1.5 * IQR
    upper_bound = Q3 + 1.5 * IQR
    # Identificar valores atípicos
    outliers = farmacia_df[(farmacia_df['valorventa'] < lower_bound) | (farmacia_df['valorventa'] > upper_bound)]
    # Reemplazar los valores atípicos con el promedio de la farmacia solo en los índices correspondientes
    df_pronosticos.loc[outliers.index, 'valorventa'] = promedios_ventas[codfarmacia]
#%% Hiperparámetros para lstm
# Definir el espacio de hiperparámetros
param_dist = {
    'neurons': [20, 50, 100],
    'epochs': [50, 100, 150],
    'batch_size': [16, 32, 64],
    'learning_rate': [0.001, 0.01, 0.1]
}
# Número de combinaciones a probar
n_iter = 10
```

```
# Evaluar para cada cluster

mejores_parametros_por_cluster = {}

for cluster in df_pronosticos['cluster_kmeans'].unique():

  print(f"Optimización para el cluster {cluster}")

  # Filtrar las farmacias en el cluster actual

  cluster_farmacias = df_pronosticos[df_pronosticos['cluster_kmeans'] ==
cluster]['codfarmacia'].unique()

  mejor_mse_cluster = np.inf

  mejores_params_cluster = None

  # Random Search para el cluster actual

  param_sampler = ParameterSampler(param_dist, n_iter=n_iter, random_state=42)

  for params in param_sampler:

    print(f"Evaluando con parámetros: {params}")

    # Evaluar las farmacias del cluster con los parámetros actuales

    total_mse = 0

    count = 0

    for codfarmacia in cluster_farmacias:

      # Filtrar los datos para la farmacia actual
```

```
        farmacia_df = df_pronosticos[df_pronosticos['codfarmacia'] == codfarmacia]
        # Preparar los datos para LSTM
        scaler = MinMaxScaler(feature_range=(0, 1))
        valorventa_scaled                                                    =
scaler.fit_transform(farmacia_df['valorventa'].values.reshape(-1, 1))
        # Crear secuencias para LSTM
        def create_sequences(data, time_steps):
          sequences = []
          labels = []
          for i in range(len(data) - time_steps):
            seq = data[i:(i + time_steps)]
            label = data[i + time_steps]
            sequences.append(seq)
            labels.append(label)
          return np.array(sequences), np.array(labels)
        time_steps = 3  # Ejemplo de 3 pasos de tiempo
        X, y = create_sequences(valorventa_scaled, time_steps)
        # Dividir en conjuntos de entrenamiento y prueba
```

```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42, shuffle=False)

# Construir el modelo LSTM con los hiperparámetros actuales

model = Sequential()

model.add(Input(shape=(X_train.shape[1], 1)))  # Usar Input para definir la forma de entrada

model.add(LSTM(units=params['neurons'], return_sequences=False))

model.add(Dense(1))

optimizer = Adam(learning_rate=params['learning_rate'])

model.compile(optimizer=optimizer, loss='mean_squared_error')

# Entrenar el modelo

model.fit(X_train, y_train, epochs=params['epochs'], batch_size=params['batch_size'], verbose=0)

# Realizar predicciones en el conjunto de prueba

y_pred = model.predict(X_test)

# Desescalar los valores predichos y reales

y_pred_descaled = scaler.inverse_transform(y_pred)

y_test_descaled = scaler.inverse_transform(y_test.reshape(-1, 1))
```

```
            # Evaluar el rendimiento del modelo
            mse = mean_squared_error(y_test_descaled, y_pred_descaled)
            total_mse += mse
            count += 1
        # Promediar el MSE para todas las farmacias en el cluster
        avg_mse = total_mse / count
        if avg_mse < mejor_mse_cluster:
            mejor_mse_cluster = avg_mse
            mejores_params_cluster = params
    print(f"Mejores hiperparámetros para el cluster {cluster}: {mejores_params_cluster}
con MSE: {mejor_mse_cluster}")
    mejores_parametros_por_cluster[cluster] = mejores_params_cluster
print("Optimización completada.")
mejores_parametros_por_cluster
#%%
# Hiperparámetros óptimos obtenidos para cada cluster
hiperparametros_optimos = {
    2: {'neurons': 50, 'learning_rate': 0.01, 'epochs': 50, 'batch_size': 32},
```

```
    0: {'neurons': 50, 'learning_rate': 0.01, 'epochs': 50, 'batch_size': 32},
    1: {'neurons': 20, 'learning_rate': 0.01, 'epochs': 100, 'batch_size': 16}}
# Crear una nueva columna en df_pronosticos para almacenar las predicciones
df_pronosticos['valorventa_pronosticado'] = np.nan
# Iterar sobre los clusters para entrenar los modelos
for cluster in df_pronosticos['cluster_kmeans'].unique():
    print(f"Entrenando modelos finales para cluster {cluster}")
    # Filtrar las farmacias en el cluster actual
    cluster_farmacias = df_pronosticos[df_pronosticos['cluster_kmeans'] ==
cluster]['codfarmacia'].unique()
    # Obtener los hiperparámetros óptimos para el cluster actual
    params = hiperparametros_optimos[cluster]
    # Iterar sobre cada farmacia en el cluster
    for codfarmacia in cluster_farmacias:
        print(f"Entrenando modelo LSTM final para farmacia {codfarmacia}")
        # Filtrar los datos para la farmacia actual
        farmacia_df = df_pronosticos[df_pronosticos['codfarmacia'] == codfarmacia]
        # Escalar los datos de ventas
```

```
    scaler = MinMaxScaler(feature_range=(0, 1))
    valorventa_scaled = scaler.fit_transform(farmacia_df['valorventa'].values.reshape(-1, 1))
    # Función para crear secuencias de datos para el LSTM
    def create_sequences(data, time_steps):
        sequences = []
        labels = []
        for i in range(len(data) - time_steps):
            seq = data[i:(i + time_steps)]
            label = data[i + time_steps]
            sequences.append(seq)
            labels.append(label)
        return np.array(sequences), np.array(labels)
    # Crear secuencias con una ventana de tiempo de 3
    X, y = create_sequences(valorventa_scaled, time_steps=3)
    # Dividir los datos en conjuntos de entrenamiento y prueba
    X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42, shuffle=False)
```

```
    # Construir el modelo LSTM con Dropout para evitar sobreajuste
    model = Sequential()
    model.add(Input(shape=(X_train.shape[1], 1)))  # Definir la forma de entrada
    model.add(LSTM(units=params['neurons'], return_sequences=False))
    model.add(Dropout(0.2))  # Añadir Dropout del 20% para mitigar el sobreajuste
    model.add(Dense(1))  # Capa densa de salida
    model.compile(optimizer=Adam(learning_rate=params['learning_rate']),
loss='mean_squared_error')
    # Configurar Early Stopping para detener el entrenamiento si no mejora
    early_stopping = EarlyStopping(monitor='val_loss', patience=10,
restore_best_weights=True)
    # Entrenar el modelo LSTM con los hiperparámetros óptimos y Early Stopping
    history = model.fit(X_train, y_train, validation_data=(X_test, y_test),
                        epochs=params['epochs'], batch_size=params['batch_size'],
                        callbacks=[early_stopping], verbose=0)

    # Realizar predicciones en el conjunto de prueba
    y_pred = model.predict(X_test)
```

```
    # Desescalar las predicciones y los valores reales para interpretarlos correctamente
    y_pred_descaled = scaler.inverse_transform(y_pred)
    y_test_descaled = scaler.inverse_transform(y_test.reshape(-1, 1))
    # Evaluar el rendimiento del modelo en conjunto de entrenamiento
    y_train_pred = model.predict(X_train)
    y_train_pred_descaled = scaler.inverse_transform(y_train_pred)
    y_train_descaled = scaler.inverse_transform(y_train.reshape(-1, 1))
    mse_train = mean_squared_error(y_train_descaled, y_train_pred_descaled)
    mae_train = mean_absolute_error(y_train_descaled, y_train_pred_descaled)
    print(f"Entrenamiento - MSE: {mse_train}, MAE: {mae_train}")
    # Evaluar el rendimiento del modelo en conjunto de prueba
    mse_test = mean_squared_error(y_test_descaled, y_pred_descaled)
    mae_test = mean_absolute_error(y_test_descaled, y_pred_descaled)
    print(f"Prueba - MSE: {mse_test}, MAE: {mae_test}")
    # Identificar los índices correspondientes a las predicciones en el DataFrame original
    pred_indices = farmacia_df.index[-len(y_pred_descaled):]
```

```
    # Insertar las predicciones en la nueva columna del DataFrame
    df_pronosticos.loc[pred_indices, 'valorventa_pronosticado'] = y_pred_descaled.flatten()
print("Entrenamiento y evaluación completados para todos los clusters.")
#%% 1. Error promedio por Cluster:
# Inicializar diccionario para almacenar los errores por cluster
errores_por_cluster = {}
for cluster in df_pronosticos['cluster_kmeans'].unique():
  # Filtrar las farmacias en el cluster actual
  cluster_farmacias = df_pronosticos[df_pronosticos['cluster_kmeans'] == cluster]['codfarmacia'].unique()
  mse_cluster = []
  mae_cluster = []
  # Calcular el MSE y MAE para cada farmacia en el cluster
  for codfarmacia in cluster_farmacias:
    # Filtrar los datos de la farmacia actual
    farmacia_df = df_pronosticos[df_pronosticos['codfarmacia'] == codfarmacia]
    # Obtener los valores reales y pronosticados
```

```
    real_values = farmacia_df['valorventa'].dropna()
    predicted_values = farmacia_df['valorventa_pronosticado'].dropna()
    # Asegurar que solo se comparen los índices que tienen ambos valores
    common_index = real_values.index.intersection(predicted_values.index)
    real_values = real_values.loc[common_index]
    predicted_values = predicted_values.loc[common_index]
    # Verificar si hay valores para comparar
    if len(real_values) > 0 and len(predicted_values) > 0:
        # Calcular el MSE y MAE
        mse = mean_squared_error(real_values, predicted_values)
        mae = mean_absolute_error(real_values, predicted_values)
        mse_cluster.append(mse)
        mae_cluster.append(mae)
# Calcular el MSE y MAE promedio para el cluster si hay datos disponibles
if mse_cluster and mae_cluster:
    mse_cluster_promedio = np.mean(mse_cluster)
    mae_cluster_promedio = np.mean(mae_cluster)
```

```
        errores_por_cluster[cluster] = {
            'MSE Promedio': mse_cluster_promedio,
            'MAE Promedio': mae_cluster_promedio
        }
print("Errores por Cluster:", errores_por_cluster)
#%% Arima
from statsmodels.tsa.arima.model import ARIMA
from sklearn.metrics import mean_squared_error, mean_absolute_error
from statsmodels.stats.stattools import durbin_watson
from statsmodels.graphics.tsaplots import plot_acf
import warnings
# Ignorar warnings para ARIMA
warnings.filterwarnings("ignore")
# Crear una nueva columna en df_pronosticos para almacenar las predicciones ARIMA
df_pronosticos['valorventa_arima'] = np.nan
# Inicializar diccionario para almacenar los errores por cluster y modelo
errores_arima_por_cluster = {}
# Iterar sobre los clusters para entrenar los modelos ARIMA
```

```
for cluster in df_pronosticos['cluster_kmeans'].unique():
    print(f"Entrenando modelos ARIMA para cluster {cluster}")
    # Filtrar las farmacias en el cluster actual
    cluster_farmacias = df_pronosticos[df_pronosticos['cluster_kmeans'] == cluster]['codfarmacia'].unique()
    mse_cluster_arima = []
    mae_cluster_arima = []
    # Iterar sobre cada farmacia en el cluster
    for codfarmacia in cluster_farmacias:
        print(f"Entrenando modelo ARIMA para farmacia {codfarmacia}")
        # Filtrar los datos para la farmacia actual
        farmacia_df = df_pronosticos[df_pronosticos['codfarmacia'] == codfarmacia]
        # Usar solo la columna 'valorventa' como serie temporal
        valorventa_series = farmacia_df['valorventa'].values
        # Definir los valores de p, d, q para la búsqueda de parámetros
        p_values = [0, 1, 2]
        d_values = [0, 1]
        q_values = [0, 1, 2]
```

```
    # Inicializar variables para almacenar los mejores parámetros y el mejor error
    best_score, best_cfg = float("inf"), None
    # Búsqueda de los mejores parámetros p, d, q
    for p in p_values:
      for d in d_values:
        for q in q_values:
          try:
            model = ARIMA(valorventa_series, order=(p,d,q))
            model_fit = model.fit()
            y_pred = model_fit.predict(start=len(valorventa_series)//2,
end=len(valorventa_series)-1)
            y_true = valorventa_series[len(valorventa_series)//2:]
            mse = mean_squared_error(y_true, y_pred)
            if mse < best_score:
              best_score, best_cfg = mse, (p,d,q)
          except:
            continue
```

```
    print(f"Mejores parámetros ARIMA para farmacia {codfarmacia}: {best_cfg} con MSE: {best_score}")
    # Entrenar el modelo ARIMA final con los mejores parámetros
    model = ARIMA(valorventa_series, order=best_cfg)
    model_fit = model.fit()
    y_pred = model_fit.predict(start=len(valorventa_series)//2, end=len(valorventa_series)-1)
    y_true = valorventa_series[len(valorventa_series)//2:]
    # Calcular el MSE y MAE para la farmacia actual
    mse_arima = mean_squared_error(y_true, y_pred)
    mae_arima = mean_absolute_error(y_true, y_pred)
    mse_cluster_arima.append(mse_arima)
    mae_cluster_arima.append(mae_arima)
    # Verificar autocorrelación en los residuos
    residuos = y_true - y_pred
    dw_stat = durbin_watson(residuos)
    print(f"Durbin-Watson stat para farmacia {codfarmacia} (ARIMA): {dw_stat}")
    plot_acf(residuos)
```

```
        # Identificar los índices correspondientes a las predicciones en el DataFrame original
        pred_indices = farmacia_df.index[len(valorventa_series)//2:]
        # Insertar las predicciones en la columna 'valorventa_arima' del DataFrame
        df_pronosticos.loc[pred_indices, 'valorventa_arima'] = y_pred
    # Calcular el MSE y MAE promedio para el cluster
    if mse_cluster_arima and mae_cluster_arima:
        mse_cluster_arima_promedio = np.mean(mse_cluster_arima)
        mae_cluster_arima_promedio = np.mean(mae_cluster_arima)
        errores_arima_por_cluster[cluster] = {
            'MSE Promedio': mse_cluster_arima_promedio,
            'MAE Promedio': mae_cluster_arima_promedio
        }
print("Errores ARIMA por Cluster:", errores_arima_por_cluster)
# Ejemplo de cálculo del promedio de Durbin-Watson
dw_stats = []  # Lista para almacenar los valores de DW
for cluster in df_pronosticos['cluster_kmeans'].unique():
    # Suponiendo que ya has calculado dw_stat para cada clúster y modelo
```

```
    dw_stats.append(dw_stat)  # Agregar cada estadístico DW a la lista

dw_promedio = np.mean(dw_stats)

print(f"Promedio Durbin-Watson para todos los clústers: {dw_promedio}")

#%% Otros Modelos

from sklearn.linear_model import LinearRegression

from statsmodels.tsa.holtwinters import ExponentialSmoothing

from sklearn.ensemble import RandomForestRegressor

from sklearn.svm import SVR

from statsmodels.tsa.api import SimpleExpSmoothing

from sklearn.metrics import mean_squared_error, mean_absolute_error

from statsmodels.stats.stattools import durbin_watson

from sklearn.model_selection import train_test_split

import warnings

import numpy as np

warnings.filterwarnings("ignore")

# Crear columnas en df_pronosticos para almacenar las predicciones

df_pronosticos['valorventa_regresion_lineal'] = np.nan

df_pronosticos['valorventa_holt_winters'] = np.nan
```

```
df_pronosticos['valorventa_rf'] = np.nan

df_pronosticos['valorventa_svr'] = np.nan

# Inicializar un diccionario para almacenar los errores por cluster y modelo

errores_modelos_clasicos = {

  'Regresión Lineal': {},

  'Holt-Winters': {},

  'Random Forest': {},

  'SVR': {}

}

# Inicializar listas para almacenar los valores de Durbin-Watson para cada modelo

dw_stats_rl = []

dw_stats_hw = []

dw_stats_rf = []

dw_stats_svr = []

# Iterar sobre los clusters para calcular los errores y generar predicciones

for cluster in df_pronosticos['cluster_kmeans'].unique():

  cluster_farmacias = df_pronosticos[df_pronosticos['cluster_kmeans'] ==

cluster]['codfarmacia'].unique()
```

```
    # Inicializar listas para almacenar errores por cluster
    mse_rl, mae_rl = [], []
    mse_hw, mae_hw = [], []
    mse_rf, mae_rf = [], []
    mse_svr, mae_svr = [], []
    for codfarmacia in cluster_farmacias:
        farmacia_df = df_pronosticos[df_pronosticos['codfarmacia'] == codfarmacia]
        valorventa_series = farmacia_df['valorventa'].values.reshape(-1, 1)
        X = np.arange(len(valorventa_series)).reshape(-1, 1)
        # Separar los datos en conjuntos de entrenamiento y prueba
        X_train, X_test, y_train, y_test = train_test_split(X, valorventa_series, test_size=0.2,
random_state=42)
        # Regresión Lineal
        if len(valorventa_series) > 0:
            model_lr = LinearRegression()
            model_lr.fit(X_train, y_train)
            pred_rl = model_lr.predict(X_test)
```

```
        df_pronosticos.loc[farmacia_df.index[-len(X_test):],
'valorventa_regresion_lineal'] = pred_rl.flatten()

        # Calcular Durbin-Watson para Regresión Lineal en el conjunto de prueba

        residuos_rl = y_test.flatten() - pred_rl.flatten()

        dw_stat_rl = durbin_watson(residuos_rl)

        dw_stats_rl.append(dw_stat_rl)

        mse_rl.append(mean_squared_error(y_test, pred_rl))

        mae_rl.append(mean_absolute_error(y_test, pred_rl))

    # Holt-Winters

    try:

      if len(y_train) >= 24:  # Asegurar que hay suficientes datos para ciclos
estacionales

        model_hw = ExponentialSmoothing(y_train, seasonal='add',
seasonal_periods=12).fit()

      else:

        model_hw = SimpleExpSmoothing(y_train).fit()

      pred_hw = model_hw.predict(start=len(y_train), end=len(y_train) + len(y_test) - 1)
```

```
        df_pronosticos.loc[farmacia_df.index[-len(y_test):], 'valorventa_holt_winters'] =
pred_hw.flatten()
        # Calcular Durbin-Watson para Holt-Winters en el conjunto de prueba
        residuos_hw = y_test.flatten() - pred_hw.flatten()
        dw_stat_hw = durbin_watson(residuos_hw)
        dw_stats_hw.append(dw_stat_hw)
        mse_hw.append(mean_squared_error(y_test, pred_hw))
        mae_hw.append(mean_absolute_error(y_test, pred_hw))
    except ValueError as e:
        print(f"Error en Holt-Winters para farmacia {codfarmacia}: {e}")
    # Random Forest
    model_rf = RandomForestRegressor(n_estimators=100, random_state=42)
    model_rf.fit(X_train, y_train.flatten())
    pred_rf = model_rf.predict(X_test)
    df_pronosticos.loc[farmacia_df.index[-len(X_test):], 'valorventa_rf'] =
pred_rf.flatten()
    # Calcular Durbin-Watson para Random Forest en el conjunto de prueba
    residuos_rf = y_test.flatten() - pred_rf.flatten()
```

```
        dw_stat_rf = durbin_watson(residuos_rf)
        dw_stats_rf.append(dw_stat_rf)
        mse_rf.append(mean_squared_error(y_test, pred_rf))
        mae_rf.append(mean_absolute_error(y_test, pred_rf))
        # SVR
        model_svr = SVR()
        model_svr.fit(X_train, y_train.flatten())
        pred_svr = model_svr.predict(X_test)
        df_pronosticos.loc[farmacia_df.index[-len(X_test):], 'valorventa_svr'] =
pred_svr.flatten()
        # Calcular Durbin-Watson para SVR en el conjunto de prueba
        residuos_svr = y_test.flatten() - pred_svr.flatten()
        dw_stat_svr = durbin_watson(residuos_svr)
        dw_stats_svr.append(dw_stat_svr)
        mse_svr.append(mean_squared_error(y_test, pred_svr))
        mae_svr.append(mean_absolute_error(y_test, pred_svr))
    # Calcular errores promedio para el cluster
    errores_modelos_clasicos['Regresión Lineal'][cluster] = {
```

```
    'MSE Promedio': np.mean(mse_rl) if mse_rl else np.nan,
    'MAE Promedio': np.mean(mae_rl) if mae_rl else np.nan,
    'DW Promedio': np.mean(dw_stats_rl) if dw_stats_rl else np.nan
}
errores_modelos_clasicos['Holt-Winters'][cluster] = {
    'MSE Promedio': np.mean(mse_hw) if mse_hw else np.nan,
    'MAE Promedio': np.mean(mae_hw) if mae_hw else np.nan,
    'DW Promedio': np.mean(dw_stats_hw) if dw_stats_hw else np.nan
}
errores_modelos_clasicos['Random Forest'][cluster] = {
    'MSE Promedio': np.mean(mse_rf) if mse_rf else np.nan,
    'MAE Promedio': np.mean(mae_rf) if mae_rf else np.nan,
    'DW Promedio': np.mean(dw_stats_rf) if dw_stats_rf else np.nan
}
errores_modelos_clasicos['SVR'][cluster] = {
    'MSE Promedio': np.mean(mse_svr) if mse_svr else np.nan,
    'MAE Promedio': np.mean(mae_svr) if mae_svr else np.nan,
    'DW Promedio': np.mean(dw_stats_svr) if dw_stats_svr else np.nan
```

```
    }

# Calcular los promedios de Durbin-Watson para cada modelo

dw_promedio_rl = np.mean(dw_stats_rl)

dw_promedio_hw = np.mean(dw_stats_hw)

dw_promedio_rf = np.mean(dw_stats_rf)

dw_promedio_svr = np.mean(dw_stats_svr)

# Imprimir los resultados

print(f"Promedio Durbin-Watson para todos los clústers (Regresión Lineal): {dw_promedio_rl}")

print(f"Promedio Durbin-Watson para todos los clústers (Holt-Winters): {dw_promedio_hw}")

print(f"Promedio Durbin-Watson para todos los clústers (Random Forest): {dw_promedio_rf}")

print(f"Promedio Durbin-Watson para todos los clústers (SVR): {dw_promedio_svr}")

# Calcular y mostrar errores MSE y MAE para cada modelo y clúster

# Inicializar un diccionario para almacenar los errores por clúster y modelo

errores_modelos = {

    'Regresión Lineal': {},
```

```
    'Holt-Winters': {},
    'Random Forest': {},
    'SVR': {},
    'LSTM': {},
    'ARIMA': {}
}
# Iterar sobre los clusters para calcular los errores
for cluster in df_pronosticos['cluster_kmeans'].unique():
    cluster_farmacias = df_pronosticos[df_pronosticos['cluster_kmeans'] ==
cluster]['codfarmacia'].unique()
    # Inicializar listas para almacenar errores por clúster
    mse_rl, mae_rl = [], []
    mse_hw, mae_hw = [], []
    mse_rf, mae_rf = [], []
    mse_svr, mae_svr = [], []
    mse_lstm, mae_lstm = [], []
    mse_arima, mae_arima = [], []
    for codfarmacia in cluster_farmacias:
```

```
    farmacia_df = df_pronosticos[df_pronosticos['codfarmacia'] == codfarmacia]
    # Obtener valores reales
    real_values = farmacia_df['valorventa'].dropna()
    # Regresión Lineal
    if 'valorventa_regresion_lineal' in farmacia_df.columns:
        pred_rl = farmacia_df['valorventa_regresion_lineal'].dropna()
        common_index = real_values.index.intersection(pred_rl.index)
        if len(common_index) > 0:
            mse_rl.append(mean_squared_error(real_values.loc[common_index],
pred_rl.loc[common_index]))
            mae_rl.append(mean_absolute_error(real_values.loc[common_index],
pred_rl.loc[common_index]))
    # Holt-Winters
    if 'valorventa_holt_winters' in farmacia_df.columns:
        pred_hw = farmacia_df['valorventa_holt_winters'].dropna()
        common_index = real_values.index.intersection(pred_hw.index)
        if len(common_index) > 0:
```

```
            mse_hw.append(mean_squared_error(real_values.loc[common_index],
pred_hw.loc[common_index]))
            mae_hw.append(mean_absolute_error(real_values.loc[common_index],
pred_hw.loc[common_index]))
        # Random Forest
        if 'valorventa_rf' in farmacia_df.columns:
            pred_rf = farmacia_df['valorventa_rf'].dropna()
            common_index = real_values.index.intersection(pred_rf.index)
            if len(common_index) > 0:
                mse_rf.append(mean_squared_error(real_values.loc[common_index],
pred_rf.loc[common_index]))
                mae_rf.append(mean_absolute_error(real_values.loc[common_index],
pred_rf.loc[common_index]))
        # SVR
        if 'valorventa_svr' in farmacia_df.columns:
            pred_svr = farmacia_df['valorventa_svr'].dropna()
            common_index = real_values.index.intersection(pred_svr.index)
            if len(common_index) > 0:
```

```
            mse_svr.append(mean_squared_error(real_values.loc[common_index],
pred_svr.loc[common_index]))
            mae_svr.append(mean_absolute_error(real_values.loc[common_index],
pred_svr.loc[common_index]))
    # LSTM
    if 'valorventa_pronosticado' in farmacia_df.columns:
        pred_lstm = farmacia_df['valorventa_pronosticado'].dropna()
        common_index = real_values.index.intersection(pred_lstm.index)
        if len(common_index) > 0:
            mse_lstm.append(mean_squared_error(real_values.loc[common_index],
pred_lstm.loc[common_index]))
            mae_lstm.append(mean_absolute_error(real_values.loc[common_index],
pred_lstm.loc[common_index]))
    # ARIMA
    if 'valorventa_arima' in farmacia_df.columns:
        pred_arima = farmacia_df['valorventa_arima'].dropna()
        common_index = real_values.index.intersection(pred_arima.index)
        if len(common_index) > 0:
```

```
            mse_arima.append(mean_squared_error(real_values.loc[common_index],
pred_arima.loc[common_index]))
            mae_arima.append(mean_absolute_error(real_values.loc[common_index],
pred_arima.loc[common_index]))
    # Calcular errores promedio para el clúster
    errores_modelos['Regresión Lineal'][cluster] = {
        'MSE Promedio': np.mean(mse_rl) if mse_rl else np.nan,
        'MAE Promedio': np.mean(mae_rl) if mae_rl else np.nan
    }
    errores_modelos['Holt-Winters'][cluster] = {
        'MSE Promedio': np.mean(mse_hw) if mse_hw else np.nan,
        'MAE Promedio': np.mean(mae_hw) if mae_hw else np.nan
    }
    errores_modelos['Random Forest'][cluster] = {
        'MSE Promedio': np.mean(mse_rf) if mse_rf else np.nan,
        'MAE Promedio': np.mean(mae_rf) if mae_rf else np.nan
    }
    errores_modelos['SVR'][cluster] = {
```

```
        'MSE Promedio': np.mean(mse_svr) if mse_svr else np.nan,
        'MAE Promedio': np.mean(mae_svr) if mae_svr else np.nan
    }
    errores_modelos['LSTM'][cluster] = {
        'MSE Promedio': np.mean(mse_lstm) if mse_lstm else np.nan,
        'MAE Promedio': np.mean(mae_lstm) if mae_lstm else np.nan
    }
    errores_modelos['ARIMA'][cluster] = {
        'MSE Promedio': np.mean(mse_arima) if mse_arima else np.nan,
        'MAE Promedio': np.mean(mae_arima) if mae_arima else np.nan
    }
# Imprimir los errores calculados por clúster y modelo
for modelo, errores in errores_modelos.items():
    print(f"Errores para {modelo}:")
    for cluster, metrics in errores.items():
        print(f"    Cluster {cluster} - MSE Promedio: {metrics['MSE Promedio']}, MAE Promedio: {metrics['MAE Promedio']}")
#%% Precisión del modelo
```

```
# Definir el rango de tolerancia

tolerancia = 0.1

# Inicializar un diccionario para almacenar la precisión por clúster y modelo

precision_modelos = {

  'Regresión Lineal': {},

  'Holt-Winters': {},

  'Random Forest': {},

  'SVR': {},

  'LSTM': {},

  'ARIMA': {}

}

# Iterar sobre los clusters para calcular la precisión

for cluster in df_pronosticos['cluster_kmeans'].unique():

  cluster_farmacias = df_pronosticos[df_pronosticos['cluster_kmeans'] == cluster]['codfarmacia'].unique()

  # Inicializar listas para almacenar precisión por clúster

  precision_rl, precision_hw, precision_rf, precision_svr, precision_lstm, precision_arima = [], [], [], [], [], []
```

```
    for codfarmacia in cluster_farmacias:
        farmacia_df = df_pronosticos[df_pronosticos['codfarmacia'] == codfarmacia]
        # Obtener valores reales
        real_values = farmacia_df['valorventa'].dropna()
        # Regresión Lineal
        if 'valorventa_regresion_lineal' in farmacia_df.columns:
            pred_rl = farmacia_df['valorventa_regresion_lineal'].dropna()
            common_index = real_values.index.intersection(pred_rl.index)
            if len(common_index) > 0:
                correct_preds = np.abs(real_values.loc[common_index] -
pred_rl.loc[common_index]) <= tolerancia * real_values.loc[common_index]
                precision_rl.append(correct_preds.mean())
        # Holt-Winters
        if 'valorventa_holt_winters' in farmacia_df.columns:
            pred_hw = farmacia_df['valorventa_holt_winters'].dropna()
            common_index = real_values.index.intersection(pred_hw.index)
            if len(common_index) > 0:
```

```
            correct_preds = np.abs(real_values.loc[common_index] -
pred_hw.loc[common_index]) <= tolerancia * real_values.loc[common_index]
            precision_hw.append(correct_preds.mean())
        # Random Forest
        if 'valorventa_rf' in farmacia_df.columns:
            pred_rf = farmacia_df['valorventa_rf'].dropna()
            common_index = real_values.index.intersection(pred_rf.index)
            if len(common_index) > 0:
                correct_preds = np.abs(real_values.loc[common_index] -
pred_rf.loc[common_index]) <= tolerancia * real_values.loc[common_index]
                precision_rf.append(correct_preds.mean())
        # SVR
        if 'valorventa_svr' in farmacia_df.columns:
            pred_svr = farmacia_df['valorventa_svr'].dropna()
            common_index = real_values.index.intersection(pred_svr.index)
            if len(common_index) > 0:
                correct_preds = np.abs(real_values.loc[common_index] -
pred_svr.loc[common_index]) <= tolerancia * real_values.loc[common_index]
```

```
            precision_svr.append(correct_preds.mean())

        # LSTM

        if 'valorventa_pronosticado' in farmacia_df.columns:

            pred_lstm = farmacia_df['valorventa_pronosticado'].dropna()

            common_index = real_values.index.intersection(pred_lstm.index)

            if len(common_index) > 0:

                correct_preds = np.abs(real_values.loc[common_index] -
pred_lstm.loc[common_index]) <= tolerancia * real_values.loc[common_index]

                precision_lstm.append(correct_preds.mean())

        # ARIMA

        if 'valorventa_arima' in farmacia_df.columns:

            pred_arima = farmacia_df['valorventa_arima'].dropna()

            common_index = real_values.index.intersection(pred_arima.index)

            if len(common_index) > 0:

                correct_preds = np.abs(real_values.loc[common_index] -
pred_arima.loc[common_index]) <= tolerancia * real_values.loc[common_index]

                precision_arima.append(correct_preds.mean())

    # Calcular precisión promedio para el clúster
```

```
    precision_modelos['Regresión Lineal'][cluster] = np.mean(precision_rl) if precision_rl else np.nan
    precision_modelos['Holt-Winters'][cluster] = np.mean(precision_hw) if precision_hw else np.nan
    precision_modelos['Random Forest'][cluster] = np.mean(precision_rf) if precision_rf else np.nan
    precision_modelos['SVR'][cluster] = np.mean(precision_svr) if precision_svr else np.nan
    precision_modelos['LSTM'][cluster] = np.mean(precision_lstm) if precision_lstm else np.nan
    precision_modelos['ARIMA'][cluster] = np.mean(precision_arima) if precision_arima else np.nan
# Imprimir la precisión calculada por clúster y modelo
for modelo, precisions in precision_modelos.items():
    print(f"Precisión para {modelo}:")
    for cluster, precision in precisions.items():
        print(f"  Cluster {cluster} - Precisión Promedio: {precision * 100:.2f}%")
# FIn
```