Evaluación de modelos de Machine Learning en polígonos irregulares en la ciudad de Medellín para la predicción de delitos

Gabriel Antonio Lopera Madrid

Informe de Práctica Académica para Optar el Título de Ingeniero de Telecomunicaciones

Modalidad de Práctica
Trabajo de Grado

Asesor Interno
Jaime Alberto Vergara Tejada, Candidato a Doctor (PhDc) en Ingeniería Electrónica

Universidad de Antioquia
Facultad de Ingeniería
Pregrado en Ingeniería de Telecomunicaciones
Medellín, Antioquia, Colombia
2024

Cita	Lopera Madrid [1]
Referencia Estilo IEEE (2020)	[1] G. A Lopera Madrid, “Evaluación de modelos de Machine Learning en polígonos irregulares en la ciudad de Medellín para la predicción de delitos”, Trabajo de grado profesional, Ingeniería de Telecomunicaciones, Universidad de Antioquia Medellín, Antioquia, Colombia, 2024.

Dedicatoria

A mi hogar, son mi fuerza para salir adelante.

TABLA DE CONTENIDO

LISTA DE TABLAS

LISTA DE FIGURAS

SIGLAS, ACRÓNIMOS Y ABREVIATURAS

IEEE	Institute of Electrical and Electronics Engineers
ML	Machine Learning
AR	Autoregressive
MA	Moving Average
ACF	Autocorrelation Function
PACF	Partial Autocorrelation Function
ARIMA	Autoregressive Integrated Moving Average
SARIMA	Seasonal Autoregressive Integrated Moving Average
ARMAX	Autoregressive Moving Average with Exogenous Variables
SIATA	Sistema de Alerta Temprana de Antioquia
LightGBM	Light Gradient Boosting Machine
MAE	Mean Absolute Error
RMSE	Root Mean Square Error
UdeA	Universidad de Antioquia

RESUMEN

Este trabajo de grado se enfoca en implementar y comparar varios modelos de Machine Learning tales como ARIMA, ARMAX, SARIMA y RNN LSTM para pronosticar la criminalidad en diferentes áreas de la ciudad de Medellín. Estos modelos predictivos se evalúan utilizando métricas como RMSE, MAE y R^2. Por otro lado, estos se alimentan mediante diferentes grupos de datos los cuales se definen en variedad de polígonos irregulares, esto como resultado de la aplicación de varios modelos de agrupamiento como K-MEANS, DBSCAN y BIRCH.

Los resultados para los modelos de agrupamiento sugieren que dos de los mencionados fueron los que presentaron un mejor desempeño, logrando una segmentación adecuada de las zonas de interés. Por consiguiente, como los datos de agrupamiento varían, la incorporación de variables exógenas y de control, así como datos socioeconómicos y meteorológicos fueron importantes para mejorar los modelos predictivos.

En síntesis, estos resultados brindan análisis y herramientas para incentivar una gestión inteligente de la seguridad ciudadana, buscando siempre optar por medidas efectivas en la predicción de delitos.

Palabras clave — K-MEANS, DBSCAN, BIRCH, ARIMA, ARMAX, SARIMA, RNN, LSTM, ML,

1. INTRODUCCIÓN

La seguridad ciudadana es un componente fundamental para el bienestar y la estabilidad de cualquier sociedad. Su importancia radica en su capacidad para garantizar que los ciudadanos puedan vivir y desarrollarse en un entorno libre de violencia y criminalidad, lo cual es esencial para el progreso económico, social y cultural de una nación. La percepción de seguridad afecta directamente la calidad de vida de las personas, ya que un ambiente seguro permite a los ciudadanos participar plenamente en actividades económicas, educativas y recreativas sin el temor constante de ser víctimas de delitos [1].

La ciudad de Medellín ha experimentado una notable transformación en términos de seguridad ciudadana en las últimas décadas. Anteriormente conocida por altos niveles de violencia y criminalidad, la ciudad ha implementado una serie de estrategias integrales que han contribuido a mejorar la seguridad y la calidad de vida de sus habitantes [2]. Sin embargo, a pesar de los avances, la ciudad aún enfrenta importantes desafíos. Se debe continuar trabajando en la implementación de políticas de seguridad sostenibles y adaptativas que respondan a las nuevas dinámicas de criminalidad y a las necesidades cambiantes de su población.

La relación entre la cantidad de delitos cometidos en Medellín y los que son efectivamente denunciados es un tema complejo. Existe un fenómeno conocido como “cifra negra”, que se refiere a los delitos que no son reportados a las autoridades y, por lo tanto, no figuran en las estadísticas oficiales. Para el año 2022, sólo se reportaron 1.028 denuncias por extorsión [3], de la misma forma para el 2023, la ciudad presentó un incremento del 9% en hurto a personas respecto al año anterior, cifras que no necesariamente reflejan la totalidad de los casos que ocurren en la ciudad [4]. Este fenómeno puede ser resultado de diversos factores, como el miedo a represalias, la desconfianza en las autoridades o la creencia de que la denuncia no llevará a una solución efectiva.

Además, los datos existentes de diversas fuentes, son analizados manualmente por los entes de vigilancia y control, prolongando el tiempo de atención oportuna, esto ralentiza significativamente el análisis de la información. Dicho de otra manera, esto subraya la importancia de la información y la participación activa de la ciudadanía, la empresa privada y la colaboración

entre diferentes niveles de gobierno. Asimismo, es crucial buscar alternativas y mecanismos que faciliten a las personas el reporte de incidentes de seguridad. La incorporación de las TICs aporta a la interacción ciudadana y de los entes de vigilancia para la gestión de la información, buscando garantizar la credibilidad de la misma [5].

Con este marco y todos estos factores a tener en cuenta, es fundamental que estos mecanismos, faciliten la creación de herramientas eficientes y fiables para que las autoridades puedan realizar un análisis, que permitan tener acción y predicción en aquellas zonas con mayor porcentaje de delitos y que la ciudadanía en general pueda también tomar prevenciones en dichas zonas. Ante esto, este trabajo busca plantear una solución de manera integral utilizando datos históricos de delitos almacenados en la plataforma Medata, complementados con información meteorológica y diversos factores socioeconómicos que aporten al problema. Con esto también se aplicará una metodología con polígonos irregulares definidos por la distribución real de delitos, analizando cuál método aporta más significativamente.

Así pues, el objetivo de este trabajo de grado es agrupar los datos utilizando algoritmos de aprendizaje no supervisado, y , evaluar los diferentes modelos de Machine Learning en dichos polígonos irregulares, utilizando el Error Cuadrático Medio (RMSE), Error Absoluto Medio (MAE) y el Coeficiente de Determinación (R^2) cómo métrica de desempeño. Estos modelos se basan en técnicas estadísticas y algorítmicas avanzadas, que permiten predecir la probabilidad de ocurrencia de delitos en ubicaciones y fechas específicas. Utilizando patrones y tendencias identificadas en los datos históricos, estos modelos pueden estimar con precisión cuándo y dónde es más probable que ocurran incidentes delictivos. Esto es adecuado para entregar herramientas a la ciudadanía y a las autoridades para la gestión de la seguridad, permitiendo la implementación de estrategias preventivas basadas en análisis predictivos, buscando también mejorar la capacidad de respuesta ante incidentes delictivos y fortalecer la seguridad en la ciudad de Medellín.

2. OBJETIVOS

2.1. Objetivo general

Evaluar el desempeño de modelos de Machine Learning en polígonos irregulares generados a partir de la distribución geográfica de diferentes variables y con diversos algoritmos de aprendizaje no supervisado, para predecir la cantidad de delitos en la ciudad de Medellín.

2.2. Objetivos específicos

- Seleccionar e integrar datos históricos de la plataforma pública Medata con variables socioeconómicas y meteorológicas que aporten información al análisis predictivo.
- Definir polígonos irregulares a partir de la distribución de delitos utilizando diferentes modelos de agrupamiento para definir las zonas sobre las cuales se realizarán las predicciones.
- Implementar y ajustar diferentes algoritmos de ML utilizados para la predicción de delitos.
- Evaluar la precisión de los modelos desarrollados por medio de las métricas: Error Cuadrático Medio (RMSE), MAE (Error Absoluto Medio), R^2 (Coeficiente de Determinación).

3. MARCO TEÓRICO

La complejidad de los datos recolectados para la predicción de delitos en la ciudad de Medellín radica en la diversidad y volumen de información necesaria, que incluye datos geoespaciales, temporales, socioeconómicos y comportamentales. Este amplio conjunto de variables refleja la naturaleza de los fenómenos delictivos, donde múltiples factores interactúan entre sí, lo que demanda enfoques avanzados para su análisis. Para abordar esta complejidad, el marco teórico integra diversas herramientas, y una de las más destacadas es el Clustering, una técnica de aprendizaje no supervisado que permite agrupar los datos en conjuntos homogéneos basados en características compartidas. Estos grupos pueden representar áreas geográficas con características delictivas similares o perfiles de comportamiento delictivo. Una vez formados los grupos, se pueden aplicar modelos de Machine Learning (ML) supervisado que tenga el componente, para predecir la ocurrencia de delitos futuros. Esta combinación de técnicas permite a las autoridades no solo entender mejor los patrones delictivos, sino también anticipar y prevenir delitos de manera más efectiva, optimizando la asignación de recursos y mejorando la seguridad pública.

3.1. Modelos para la agrupación de datos

El Clustering es una técnica de aprendizaje automático no supervisado utilizada para dividir un conjunto de datos en grupos, de tal manera que los puntos de datos en el mismo grupo son más similares entre sí que con los de otros grupos [1]. En el contexto de la predicción de delitos, se puede aplicar esta técnica utilizando las coordenadas geográficas, como longitud y latitud, para agrupar delitos en áreas específicas. De esta manera, es posible identificar patrones espaciales de criminalidad, lo que facilita la segmentación geográfica de zonas de alto riesgo y permite a las autoridades enfocar mejor sus esfuerzos de prevención y seguridad.

3.1.1. K-means

Este modelo de clustering es ampliamente utilizado para agrupar grandes conjuntos de datos. Es un método numérico, no supervisado, no determinista e iterativo que clasifica los objetos de datos en k clusters a través de iteraciones que convergen a un mínimo local. Aunque los resultados son compactos e independientes, el algoritmo tiene una alta complejidad computacional

debido a la necesidad de reasignar los puntos de datos varias veces en cada iteración, afectando su eficiencia. Funciona seleccionando k centros (o centroides) aleatoriamente y asignando cada objeto de datos al centro más cercano usando la distancia euclidiana. Este proceso se repite hasta minimizar la función de criterio, definida como la suma del error cuadrático de todos los objetos. En cada iteración, se recalculan los promedios de los clusters, actualizando los centroides según los objetos asignados, utilizando la distancia euclidiana para determinar la cercanía entre los objetos y los centros de los clusters [2].

La función a minimizar se expresa como:

$$E = \sum_{i=1}^{k} \sum_{x \in C_i} |x - x_i|^2$$

Donde E es la función a minimizar, k el número total de clusters, Ci representa el i-ésimo cluster, x es un objeto de datos que pertenece al cluster Ci, Xi es el promedio o centroide del cluster Ci.

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

Este modelo está basado en la densidad, que se destaca por su capacidad para identificar clusters de formas arbitrarias en un conjunto de datos. Este modelo clasifica las observaciones en tres tipos: puntos núcleo, puntos frontera y ruido. Un punto núcleo tiene un número mínimo de otros puntos dentro de un radio predefinido, un punto frontera es alcanzable desde un punto núcleo pero no tiene suficientes vecinos para ser un punto núcleo por sí mismo, y el ruido son puntos que no son alcanzables desde ningún punto núcleo. El modelo cuenta con un parámetro ‘eps’, que es el radio máximo entre dos puntos para que uno sea considerado en el vecindario del otro, y el parámetro ‘MinSamples’, es el número mínimo de puntos requeridos para formar un cluster denso. DBSCAN comienza con un punto arbitrario, y si este tiene suficientes vecinos dentro de eps, se forma un cluster. Luego, se expande el cluster agregando todos los puntos densamente alcanzables [3].

3.1.3. Balanced Iterative Reducing and Clustering using Hierarchies (BIRCH)

Su principal ventaja es que puede encontrar una buena agrupación con un solo escaneo de los datos, y mejorar la calidad con algunos escaneos adicionales. El funcionamiento de BIRCH se basa en la construcción de un árbol CF (Clustering Feature Tree) en memoria, que permite organizar y reducir los datos de manera eficiente. Este árbol se construye en la primera fase del algoritmo, donde se escanean todos los datos y se agrupan en subclusters utilizando la memoria disponible. BIRCH es local en su enfoque, lo que significa que toma decisiones de agrupamiento sin necesidad de escanear todos los puntos de datos o clusters existentes, lo que lo hace más eficiente en comparación con métodos globales. Además, BIRCH utiliza medidas que reflejan la cercanía natural de los puntos, y puede mantener estas medidas de forma incremental durante el proceso de agrupamiento. Esto permite que el algoritmo trate regiones densas de puntos como un solo cluster, mientras que los puntos en regiones dispersas pueden ser considerados como outliers y eliminados opcionalmente [4].

3.1.4. Silhouette Method

Es una técnica utilizada para evaluar la calidad de un modelo de clustering, midiendo qué tan bien están separados los clusters. Se basa en calcular un valor para cada punto que refleja la similitud con los puntos de su propio cluster en comparación con otros clusters. Este valor varía entre -1 y 1, donde un valor cercano a 1 indica que el punto está bien agrupado, 0 sugiere que el punto está en el límite entre dos clusters, y un valor negativo indica que el punto podría estar en el cluster incorrecto. El valor promedio de la silueta para todos los puntos proporciona una medida general de la calidad del clustering, siendo especialmente útil para determinar el número óptimo de clusters [5].

3.1.5. Elbow Method

Es otra técnica utilizada para determinar el número óptimo de clusters en un algoritmo de clustering, como K-means. Funciona midiendo la suma de las distancias cuadradas dentro de cada cluster (inercia) para diferentes cantidades de clusters. A medida que se incrementa el número de clusters, la inercia disminuye porque los puntos se agrupan en grupos más pequeños. Sin embargo, llega un punto donde añadir más clusters no mejora significativamente la compactación dentro de los clusters. Este punto se visualiza como un "codo" en la gráfica que relaciona el número de

clusters con la inercia. El "codo" indica el número de clusters óptimo, donde se logra un buen equilibrio entre precisión y simplicidad [6].

3.2. Propiedades de series temporales

3.2.1. Estacionariedad

La estacionariedad es una propiedad de una serie temporal que indica que sus características estadísticas, como la media, la varianza y la autocovarianza, son constantes en el tiempo. Esto significa que los patrones de la serie no cambian a lo largo del tiempo, lo que facilita la predicción y el análisis. Si los datos no son estacionarios, pueden producirse sesgos en las predicciones y los resultados pueden no ser fiables. Para abordar esta cuestión, es común aplicar técnicas como la diferenciación o la transformación logarítmica para estabilizar la media y la varianza antes de modelar, asegurando así que los supuestos del modelo se cumplan y mejorando la precisión de las predicciones [7].

3.2.2. Autocorrelación

Evalúa la dependencia entre los valores de una serie temporal y sus propios valores en diferentes instantes de tiempo. Esta técnica permite detectar patrones repetitivos, estacionalidad y tendencias inherentes en los datos. La autocorrelación es importante para identificar la estructura subyacente de la serie, lo que facilita la selección de modelos adecuados como ARIMA o SARIMA, que requieren conocer la correlación de un valor con sus precedentes. Esto mejora la precisión de las predicciones al capturar la dinámica temporal, permitiendo un modelado más robusto de fenómenos como fluctuaciones periódicas o dependencias a largo plazo en los datos [8].

3.3. Modelos para predicción de series temporales

3.3.3. ARIMA

Autoregressive Integrated Moving Averag, este modelo combina tres componentes principales: autoregresión (AR), media móvil (MA) e integración (I). El componente AR describe la relación entre una observación y sus valores pasados, mientras que el componente MA modela el error de predicción como una combinación lineal de errores pasados. La integración (I) se utiliza

para hacer que una serie temporal no estacionaria se convierta en estacionaria, mediante la diferenciación de los datos, lo que permite trabajar con series temporales que tienen tendencias o variaciones estacionales. Este modelo es particularmente efectivo para datos de series temporales que contengan tendencias y estacionalidades [9].

ARIMA se representa con tres parámetros fundamentales que configuran el modelo: p, d, q. Cada uno de estos parámetros representa una parte específica del proceso que sigue la serie temporal para hacer predicciones. En un modelo de primer orden, el parámetro *p =1* significa que la predicción en el tiempo actual depende de la observación inmediatamente anterior, lo que introduce una componente autoregresiva de primer orden. El parámetro *d=1* indica que la serie ha sido diferenciada una vez para hacerla estacionaria, eliminando las tendencias a largo plazo. Finalmente, *q=1* en un modelo de primer orden significa que el componente de media móvil utiliza el error de predicción del paso anterior para ajustar el valor estimado. Es decir, un modelo ARIMA (1, 1, 1) utiliza una observación previa, una diferenciación de primer orden y un error de predicción inmediato para hacer las estimaciones. Este tipo de modelo es particularmente útil cuando se observan dependencias temporales inmediatas entre los datos, como en series financieras o de criminalidad [10].

3.3.4. SARIMA

También conocido como Seasonal ARIMA, se presenta como una extensión de ARIMA para incluir componentes estacionales para capturar patrones repetitivos a lo largo de períodos específicos de tiempo. Este introduce parámetros adicionales, representados por *P, D y Q* que son las contrapartes estacionales de los parámetros autoregresivos *(p)*, de diferenciación *(d)*, y de media móvil *(q)* en el modelo ARIMA estándar. Ahora bien, en SARIMA la estacionalidad se modela mediante la inclusión de estos parámetros a intervalos regulares, lo que lo convierte en una opción ideal para series temporales donde se observan fluctuaciones recurrentes, como datos trimestrales o anuales. Además, el modelo incluye un parámetro *(S)* que indica la duración del ciclo estacional, como ejemplo 12 meses para datos mensuales. Esto lo hace ideal para manejar tendencias a largo plazo como las fluctuaciones estacionales, como predicción de ventas, análisis de temperaturas y la predicción de delitos con componentes cíclicos [10] [11].

3.3.5. ARMAX

Autoregressive Moving Average with Exogenous Inputs, este modelo combina dos componentes, un modelo autoregresivo (AR) y un modelo de media móvil (MA), pero incluye un tercer componente con variables adicionales que pueden afectar la variable de interés. Estas variables exógenas pueden ser factores externos que influyen en la serie temporal, pero que no forman parte de la serie misma. Es importante que las variables exógenas estén correlacionadas entre si o con la variable de interés, ya que puede presentar dificultades al momento de interpretar el coeficiente de la variable exógena [11].

3.3.6. Red Neuronal Recurrente

Conocida como (RNN) es una alternativa muy importante para los modelos de aprendizaje profundo, mejorando el procesamiento de datos secuenciales y temporales. Estas redes incorporan conexiones recurrentes que les permiten mantener una memoria interna, facilitando el procesamiento de secuencias de longitud variable y la captura de dependencias temporales en los datos.

La arquitectura fundamental de una RNN se distingue por la capacidad que presenta para mantener un estado oculto que funciona como una forma de memoria, permitiendo que la red conserve información de entradas previas para influir en las predicciones actuales tal como se evidencia en la ilustración 1. Esta característica es muy valiosa para aplicaciones que dependen en gran medida del contexto temporal, como el procesamiento del lenguaje natural, el análisis de series temporales y el reconocimiento de voz. En cada paso temporal, la red procesa una nueva entrada en conjunto con su estado oculto anterior, actualizando este estado y generando una predicción basada en toda la información procesada hasta ese momento.

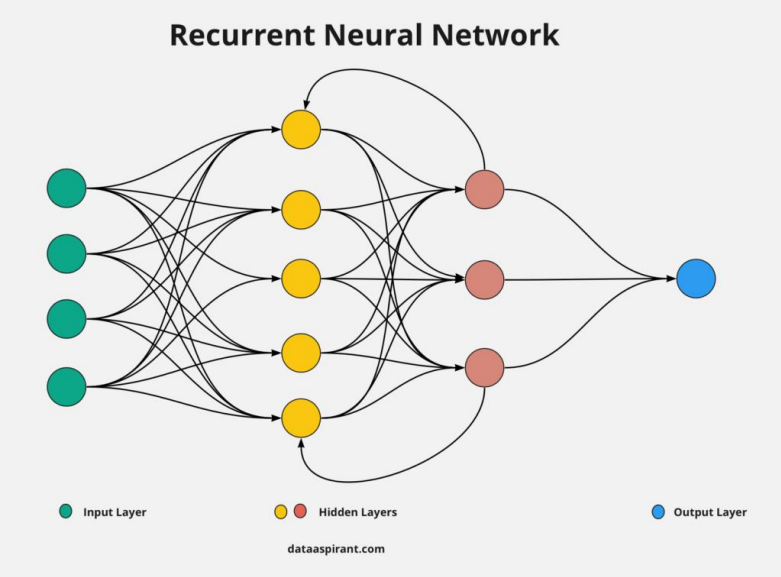


Ilustración 1. Diagrama Red Neuronal Recurrente [12].

Matemáticamente, una RNN básica, implica una transformación no lineal del estado oculto previo y la entrada actual, normalmente mediante la función tangente hiperbólica, seguida de una transformación lineal para generar la salida. Sin embargo, esta arquitectura básica enfrenta desafíos, particularmente el problema del desvanecimiento del gradiente, que dificulta el aprendizaje de dependencias a largo plazo en las secuencias.

3.3.7. Red Neuronal Recurrente con LSTM

Para abordar las limitantes que ofrecen las RNN, se ha adoptado una variante como lo es la Long Short-Term Memory (LSTM). Estas introducen un mecanismo de compuertas que permite a la red aprender selectivamente qué información debe recordar y qué debe olvidar, facilitando la captura de dependencias a largo plazo. La compuerta de olvido decide qué información del estado de la celda anterior debe ser descartada. La función sigmoide produce valores entre 0 y 1, donde 0 significa "olvidar completamente" y 1 significa "mantener completamente". La compuerta de entrada determina que nueva información se almacenará en el estado de la celda, con una función sigmoide que decide que valores actualizar y una capa tangente hiperbólica que crea un vector de nuevos valores candidatos. El estado de la celda se actualiza combinando la información filtrada del estado anterior con la nueva información candidata. Por último, la compuerta de salida, que controla qué partes del estado de celda se transmitirá a la salida.

Estas redes con su control de información, gradientes estables, memoria selectiva y la robustez que presentan, son una excelente alternativa para las predicciones en series temporales.

4. METODOLOGÍA

La presente metodología adopta un enfoque sistemático que comienza en la recopilación, preparación, selección y limpieza de los datos hasta la implementación de modelos de agrupación para la definición de polígonos irregulares y con ello también en el desarrollo de modelos de predicción de delitos, estos dos últimos con la particularidad de que se ajustarán y evaluarán buscando los mejores resultados.

4.1. Recopilación de datos, meteorológicos y socioeconómicos

Inicialmente, es importante analizar y comprender los factores que pueden influir en la incidencia de delitos, como los socioeconómicos y meteorológicos. Factores socioeconómicos, como el desempleo, la pobreza y la desigualdad, pueden aumentar la presión social y económica, llevando a un mayor riesgo de actividades delictivas. Esto conlleva a analizar días feriados, día de pago y fines de semana entre las variables a tener en cuenta para las predicciones. Por otro lado, las condiciones meteorológicas, como la temperatura y las precipitaciones, también pueden afectar la frecuencia y el tipo de delitos cometidos, ya que ciertos climas pueden facilitar o dificultar la comisión de delitos. Estas variables se conocen como variables de control, permiten enriquecer la capacidad del modelo y desarrollar modelos predictivos más precisos y efectivos, proporcionando una base sólida para la implementación de estrategias de prevención del crimen.

El Sistema de Alerta Temprana de Antioquia (SIATA), ofrece datos históricos meteorológicos. Además de ello, la plataforma MEData de la ciudad de Medellín busca la apropiación, apertura y uso de los datos como herramienta de gobierno, acción ciudadana y toma de decisiones. Esta proporciona los datos históricos de delitos ocurridos en la ciudad.

4.2. Distribución estaciones meteorológicas

El SIATA proporcionó un conjunto robusto de datos históricos que incluyeron no solo las medidas de temperatura y precipitación, sino también información detallada sobre la ubicación y el periodo de funcionamiento de cada estación meteorológica. Esta rica fuente de datos permitió ubicarlas en el mapa tal como se muestra en la ilustración 2, y asociarlas a los datos del histórico

de delitos, las cuales fueron fundamentales para explorar posibles asociaciones entre las condiciones climáticas y los patrones de criminalidad en la ciudad de Medellín.

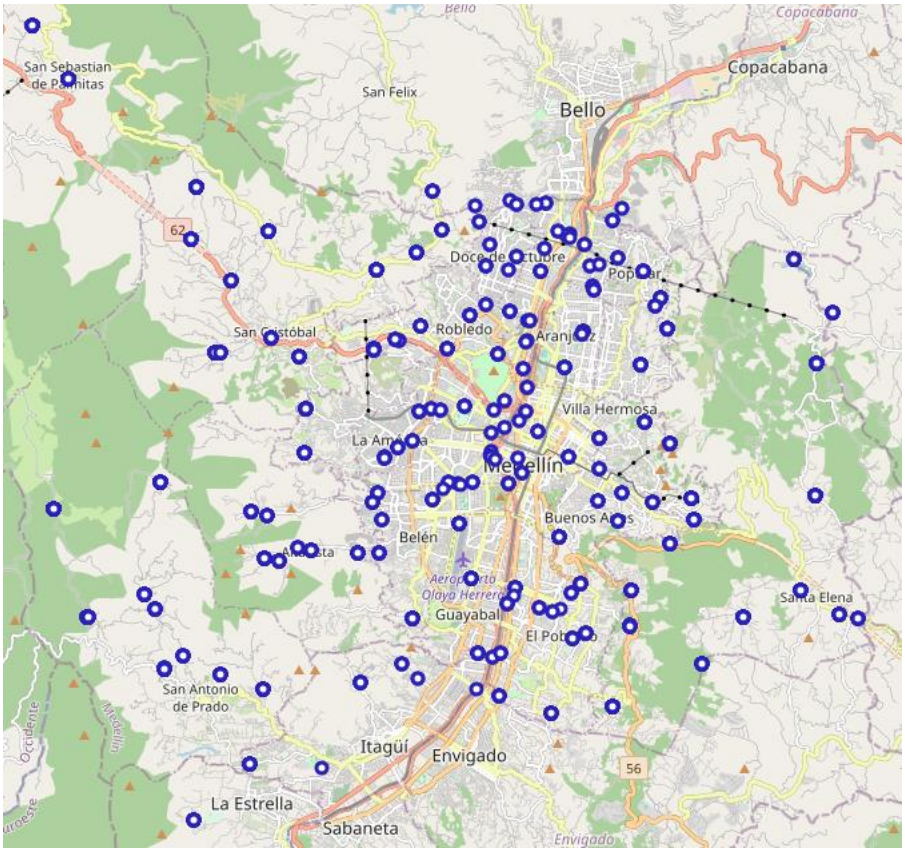


Ilustración 2. Distribución de estaciones meteorológicas.

4.3. Histórico de delitos

Los datos históricos de delitos que brinda la plataforma MEData, se extrajeron por medio del back-end del proyecto de investigación "Administración inteligente de problemas de seguridad ciudadana a través de modelos y herramientas generadas a partir de plataformas para territorios inteligentes apoyadas por estrategias de participación ciudadana en la ciudad de Medellín".

Estos datos vienen procesados de manera que puedan ser tratados fácilmente. En el proyecto de investigación anteriormente mencionado, se optó por dividir la ciudad de Medellín en hexágonos, cada uno con un identificador único [13]. Esto permitió una asociación eficiente entre las estaciones meteorológicas y los delitos registrados en cada área. Sin embargo, en este proyecto se ha decidido utilizar una división distinta, basada en modelos de clustering. lo que indicará

divisiones irregulares de acuerdo a las propiedades de esto datos. Este enfoque busca capturar la heterogeneidad espacial de una manera más ajustada al fenómeno que se desea analizar.

Si bien se tienen los delitos desde el año 2012 hasta la fecha, el análisis en este trabajo se hará con los delitos desde el año 2019 hasta el 2022. Esta elección es aleatoria y no se fundamenta en ningún criterio.

4.4. Entorno del desarrollo del trabajo

A continuación, se detallan las librerías y herramientas utilizadas para el desarrollo del trabajo que fueron vitales para ejecutar los diversos modelos y análisis del proyecto. La integración de estas tecnologías facilitó una exploración detallada y la obtención de resultados diversos para entender la dinámica de los datos.

4.4.1. Lenguaje de programación y características del sistema

Se utilizó la CPU en la nube de Google Colab con el backend de Google Compute Engine. Esta emplea Python 3, específicamente la versión 3.10.12 y con las siguientes características del sistema:

- System: Linux
- Release: 6.1.85+
- Version: #1 SMP PREEMPT_DYNAMIC Thu Jun 27 21:05:47 UTC 2024
- Processor: x86_64
- Total RAM: 12,67 GB
- Total Disk Space: 107,72 GB

4.4.2. Librerías utilizadas

5.4.2.1. Librerías para agrupación

- KMeans (from sklearn.cluster import KMeans): Método de agrupación basado en centroides que divide los datos en un número específico de clusters.
- DBSCAN (from sklearn.cluster import DBSCAN): Algoritmo de clustering basado en densidad que forma clusters de puntos densamente conectados.
- Birch (from sklearn.cluster import Birch): Algoritmo de clustering jerárquico que forma una estructura de árbol para agrupar grandes conjuntos de datos.

- StandardScaler (from sklearn.preprocessing import StandardScaler): Escala los datos para tener media 0 y desviación estándar 1, esencial en clustering para evitar sesgos en los cálculos de distancias.
- SilhouetteVisualizer (from yellowbrick.cluster import SilhouetteVisualizer): Visualización de la calidad de los clusters mediante la métrica de silueta.
- silhouette_score (from sklearn.metrics import silhouette_score): Métrica que evalúa la cohesión y separación de los clusters.
- ConvexHull (from scipy.spatial import ConvexHull): Calcula el "envolvente convexa" para visualizar el perímetro de los clusters en el mapa.
- davies_bouldin_score (from sklearn.metrics import davies_bouldin_score): Métrica que evalúa la separación de los clusters, siendo valores bajos mejores.
- calinski_harabasz_score (from sklearn.metrics import calinski_harabasz_score): Métrica que mide la dispersión dentro de los clusters en relación con la dispersión entre clusters.

4.4.2.2. Librerías para predicción

- ARIMA (from statsmodels.tsa.arima.model import ARIMA): Modelo autorregresivo integrado para predicciones basadas en series temporales.
- SARIMAX (from statsmodels.tsa.statespace.sarimax import SARIMAX): Extensión del modelo ARIMA que incluye factores estacionales y otras variables exógenas.
- acf, pacf (from statsmodels.tsa.stattools import acf, pacf): Funciones de autocorrelación y autocorrelación parcial, usadas para identificar la estructura de dependencia en series temporales.
- adfuller (from statsmodels.tsa.stattools import adfuller): Prueba de Dickey-Fuller para comprobar la estacionariedad de una serie temporal.
- mean_squared_error, mean_absolute_error, r2_score (from sklearn.metrics import mean_squared_error, mean_absolute_error, r2_score): Métricas usadas para evaluar el rendimiento de los modelos predictivos, como el error cuadrático medio, error absoluto medio y el coeficiente de determinación (R^2).

- LSTM (from keras.layers import LSTM): Tipo de red neuronal recurrente utilizada para predecir secuencias temporales, útil para predicción de delitos a lo largo del tiempo.
- Sequential (from keras.models import Sequential): Arquitectura secuencial de modelos en Keras, utilizada para construir redes neuronales.
- EarlyStopping (from keras.callbacks import EarlyStopping): Detiene el entrenamiento de la red neuronal cuando no hay mejoras significativas, ayudando a evitar el sobreajuste.
- Statsmodels.api (import statsmodels.api as sm): Biblioteca para análisis estadísticos, incluyendo modelos de series temporales como ARIMA y SARIMAX.
- Tensorflow, keras (import tensorflow as tf, from keras.models import Sequential, from keras.layers import LSTM, SimpleRNN, Dense, Dropout): Librerías para construir y entrenar redes neuronales, utilizadas para predicciones de series temporales con modelos LSTM y RNN.

4.4.2.3. Librerías de visualización

- Matplotlib.pyplot (import matplotlib.pyplot as plt): Librería para crear gráficos y visualizaciones 2D, como líneas, dispersión, y diagramas de barras.
- Seaborn (import seaborn as sns): Librería para visualización de datos que extiende matplotlib con gráficos estadísticos de mayor nivel.
- Folium (import folium): Herramienta para crear mapas interactivos, en este caso usada para visualizar clusters o patrones geográficos sobre un mapa.
- Folium.plugins (import folium.plugins as plugins): Extiende folium con funcionalidades adicionales, como agrupación de marcadores, líneas de tiempo, entre otros.
- Matplotlib (import matplotlib): Librerías para estética de los gráficos.
- Natsort (from natsort import natsorted): Ordena de manera natural (alfanuméricamente) listas de elementos, útil cuando las etiquetas de los datos no siguen un orden numérico simple.

4.4.2.4. Librerías para manejo de datos y utilidades adicionales

- Pandas (import pandas as pd): Biblioteca fundamental para la manipulación y análisis de datos estructurados. Proporciona estructuras de datos como DataFrame para trabajar con datos tabulares.
- Numpy (import numpy as np): Biblioteca clave para cálculos numéricos, utilizada para manejar arreglos y realizar operaciones matemáticas de manera eficiente.
- Holidays (import holidays): Permite crear listas de días festivos, útil para agregar información relevante de días no laborales o festivos en el análisis de series temporales.
- Pandas.tseries.holiday (from pandas.tseries.holiday import AbstractHolidayCalendar, Holiday, nearest_workday, Easter, EasterMonday): Proporciona herramientas para agregar y manejar calendarios de días festivos personalizados en los análisis de series temporales.
- Pandas.tseries.offsets (from pandas.tseries.offsets import DateOffset, Week, Day): Herramientas para crear desplazamientos de tiempo en series temporales, como añadir semanas, días, etc.
- Concurrent.futures (import concurrent.futures): Permite la ejecución de código de manera concurrente usando hilos o procesos, útil para paralelizar operaciones que consumen mucho tiempo.
- Pickle (import pickle): Se utiliza para serializar y deserializar objetos de Python, permitiendo guardar y cargar modelos o datos de manera persistente.
- Gdown (import gdown): Herramienta para descargar archivos desde Google Drive, especialmente útil en entornos colaborativos como Google Colab.

4.5. Análisis de Agrupamiento

Antes de realizar un análisis de características y posteriormente aplicar a modelos de predicción, es importante realizar una tarea de agrupación, más conocida como Clustering, esto con el fin de reducir la dimensionalidad, detectar anomalías, segmentar la población en grupos homogéneos y demás patrones subyacentes que se puedan encontrar en los datos.

Para ello, se inició con el algoritmo K-Means, recordando que se está haciendo el análisis con los datos históricos de los delitos de Medellín entre los años 2019 a 2022, donde se buscaba agrupar los datos en conjuntos de clusters basados en sus similitudes. En este caso específico se hizo el análisis con la latitud y longitud, buscando agrupar cada delito al centroide más cercano y luego recalculando los cretroides hasta que se alcance la convergencia. En la creación del modelo se fijó una semilla aleatoria de 42 para garantizar que los resultados sean reproducibles.

Sin embargo, este modelo de entrada, tiene la necesidad de especificar el número de clusters para realizar la agrupación, lo cual supone una limitación si se asigna un número aleatorio y se crea la suposición de que los clusters serán convexos y de tamaño similar.

El método de la silueta es una técnica que evalúa la calidad de un agrupamiento midiendo qué tan bien se ajusta cada punto de datos a su propio clúster en comparación con los demás clústeres. Para cada punto, se calcula un valor de silueta que oscila entre -1 y 1. Un valor cercano a 1 indica que el punto está correctamente asignado a su clúster, mientras que un valor cercano a -1 sugiere que el punto podría pertenecer a otro clúster. El valor promedio de la silueta para todos los puntos proporciona una medida global de la calidad de la agrupación.

En este caso, se implementó el método de la silueta comenzando con un número mínimo de 2 clústeres, ya que es necesario al menos esa cantidad para calcular el coeficiente, y se extendió hasta 5 clústeres, tal como se muestra en la ilustración 3. Esto se debió a la complejidad computacional de evaluar una mayor cantidad de clústeres. Cabe mencionar que en pruebas adicionales realizadas con 10, 20 e incluso 50 clústeres, el promedio más alto del valor de silueta se obtuvo con 4 clústeres. Esta consistencia indica que los patrones en los datos son robustos, y que este número de clústeres maximiza tanto la cohesión interna dentro de cada clúster como la separación entre clústeres, lo que significa que los puntos dentro de cada clúster son muy similares y los clústeres están bien diferenciados entre sí.

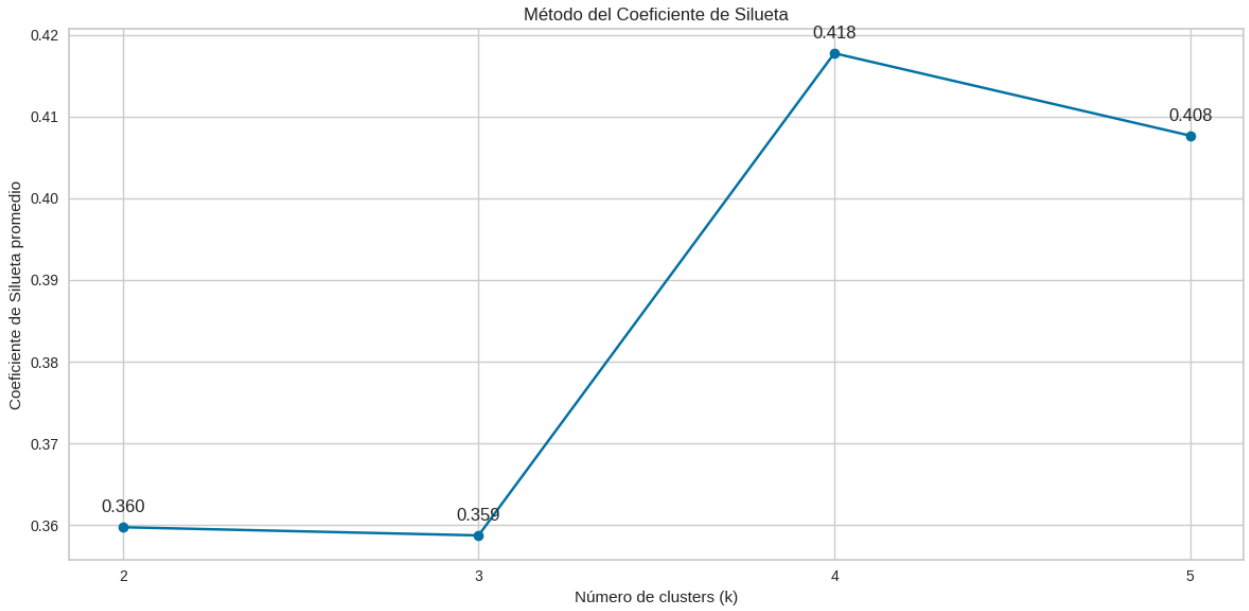


Ilustración 3. Método de Silueta con datos 2019 a 2022 para K-Means.

Complementario a esto, es ideal probar con otro método que permita confirmar el anterior resultado. Para ello se utilizó el método del codo que proporciona otra estimación del número óptimo de clusters. Al combinar ambas técnicas, se pudo obtener una evaluación más precisa y confiable de los resultados del análisis de agrupamiento.

Este método calcula la inercia que mide la suma de las distancias cuadradas de cada punto a su centroide más cercano, y es una medida de la compacidad de los clusters. Esta se detecta visualmente en la gráfica donde la inercia deja de disminuir significativamente al aumentar el número de clusters, formando un codo en la gráfica.

Tal y como se observa en la ilustración 4, los cambios en la inercia se encuentran bien acoplados y casi que indistinguibles. Sólo se detecta un pequeño cambio en el cluster 4, lo que puede dar indicios de que este número es el indicado para la segmentación de clusters.

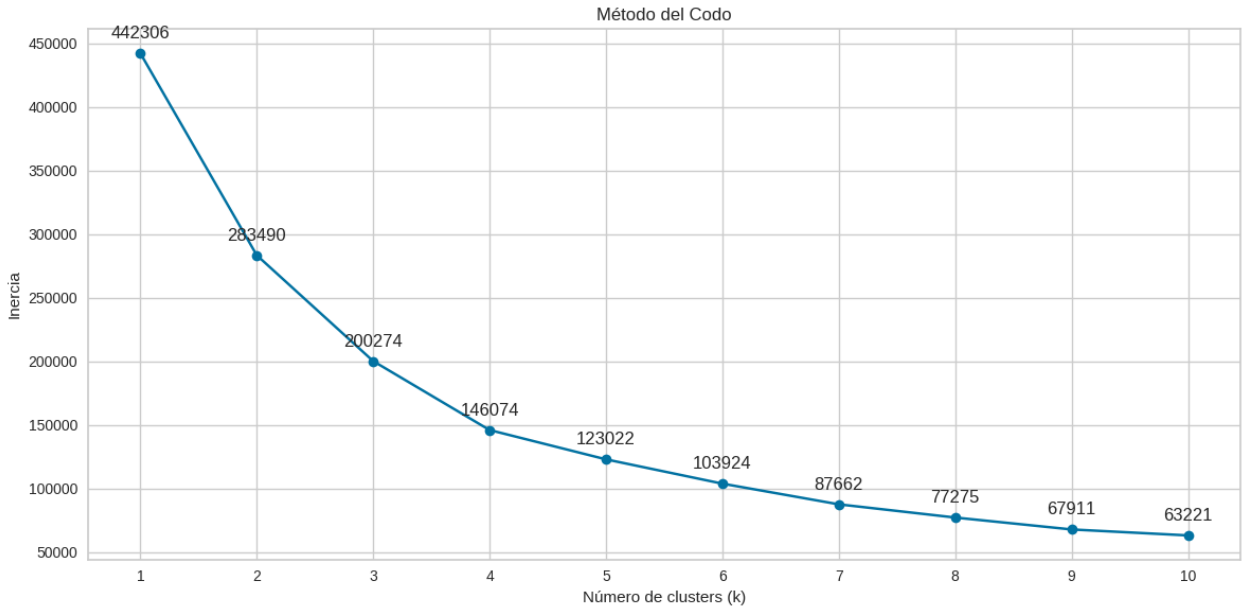


Ilustración 4. Método del codo con datos 2019 a 2022 para K-Means.

El BIRCH es menos costoso computacionalmente, utiliza la estrategia de construcción de un árbol de clustering de forma incremental, lo que lo hace mucho más eficiente que el K-Means. Este se configuró con una threshold igual a 0,99 para medir los dos puntos de datos de un sub-cluster. Este es un proceso iterativo que requiere una combinación de técnicas estadísticas y conocimiento del dominio del problema.

Posteriormente, se procedió con la creación y configuración del modelo DBSCAN. Este indica que es especialmente útil cuando los datos presentan agrupaciones de diferentes formas y tamaños, o cuando hay ruido (outliers). En este se intenta hallar la distancia a K-Vecinos, pero este modelo cuenta con dos hiperparámetros, ‘minsamples’, ‘pes’, para definir todos los puntos desde ese radio, debe ser definidos al iniciarlo. Afortunadamente, también se cuenta con una técnica llamada K-Distance Graph, que es útil para hallar las dos variables de acuerdo necesarias para el clustering, como se muestra en la ilustración 5. Con estos resultados se determina el k-esimo vecino más cercano, es decir, este método se aplicará con un ‘eps’ aproximado de 0,3 que es el valor en el eje ‘Y’ donde se observa un cambio fuerte en las distancias a los vecinos, que indica la distancia máxima desde el núcleo entre dos muestras para considerarse vecinas, y con un ‘min_samples’ de 100 que será el número mínimo de puntos requeridos para formar un cluster.

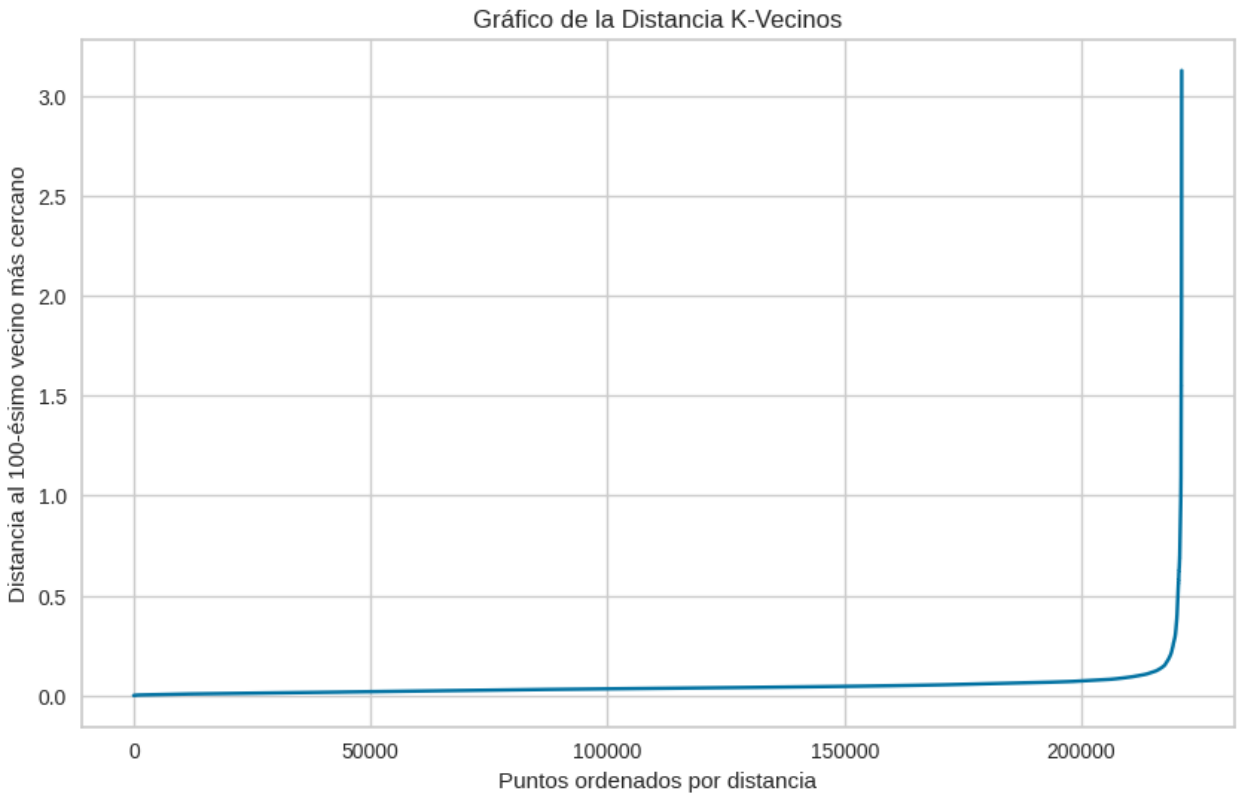


Ilustración 5. Distancia 100-ésimo vecino cercano para método DBScan

Este resultado que se llevará al modelo de agrupación, puede indicar que hay una clase dominante o una distribución de datos sesgada. Es decir, existe algún sesgo en los datos o se están enviando parámetros incorrectos.

4.5. Análisis de características

Antes de comenzar a realizar los ajustes necesarios para los modelos de predicción, es necesario realizar un análisis de las características que alimentarán el modelo. Esto ayuda a comprender la naturaleza de los datos y que diferencias hay entre ellas y sus respectivas variables. Para esto ya se tiene un trabajo relacionado donde se utilizó la misma fuente de información, pero con diferente planteamiento de agrupación, en el cual realizaron un análisis exhaustivo, con resultados de gran interés que aportan a las predicciones. Se creó un modelo de clasificación, considerando tasa alta y baja de delitos, representados como ‘1’ y ‘0’ respectivamente. Dicha clasificación se hizo promediando la cantidad de delitos permitiendo tener un modelo inicial. Este

enfoque simplifica el análisis al dividir las zonas o momentos en dos clases y permite abordar el problema de manera escalable, facilitando la creación de un modelo inicial de predicción.

Con esto, se procedió a la implementación de dos modelos principales de aprendizaje automático, la regresión logística con regularizaciones L1 (Lasso) y L2 (Ridge), y LightGBM. Esta aproximación dual permitió una evaluación comparativa de las capacidades predictivas de diferentes enfoques algorítmicos [13].

Un aspecto fundamental del procedimiento implementado, fue el manejo del desequilibrio inherente en las clases de datos. Inicialmente, el modelo de regresión logística mostró limitaciones significativas en la predicción de la clase minoritaria (en este caso es la tasa alta de delitos). Para abordar esta problemática, se implementaron técnicas de balanceo de clases, lo que resultó en una mejora en la detección de la clase minoritaria, aunque con cierto deterioro en la predicción de la clase mayoritaria [13].

En las métricas Split y Gain de LightGBM, se evidenció una estrecha relación entre las condiciones meteorológicas y la ocurrencia de delitos. De manera particular, la temperatura se comportó como el factor climático más influyente en la predicción de delitos. Los niveles de precipitación, medidos a través de los pluviómetros P1 y P2, también mostraron una correlación significativa con la tasa de criminalidad [13]. Estos hallazgos indican que las variaciones en el clima podrían desencadenar cambios en los patrones delictivos, posiblemente debido a factores como el aumento de la actividad al aire libre en días cálidos o la disminución de la visibilidad durante las lluvias.

En contraste, las variables temporales como los fines de semana, días festivos o fechas cercanas a pagos, mostraron una influencia más modesta en el modelo predictivo [13]. No obstante, se logró identificar cierto impacto de los eventos especiales y ciclos económicos en la dinámica delictiva, aunque en menor medida que las variables meteorológicas.

Es importante señalar que la efectividad y la relevancia de las variables identificadas en este estudio pueden variar significativamente según el contexto geográfico, socioeconómico y temporal de los datos analizados. Los patrones delictivos y sus correlaciones con variables meteorológicas y temporales pueden manifestarse de manera diferente en distintas regiones urbanas o rurales, así como en diferentes contextos culturales y socioeconómicos. Por lo tanto, mientras que esta investigación proporciona hallazgos significativos, la aplicación de estos modelos en otros contextos requeriría una cuidadosa validación y posible adaptación a las

características específicas de cada entorno. Esto subraya la importancia de considerar las particularidades locales y temporales al implementar modelos predictivos de delitos, como también la necesidad de una constante actualización y refinamiento de los modelos según las características específicas de cada conjunto de datos y contexto de aplicación.

4.6. Análisis de estacionariedad

Para asegurar la validez de los modelos de predicción, se llevó a cabo un análisis de estacionariedad de los datos mediante la prueba de Dickey-Fuller. Esta prueba estadística se aplicó a múltiples regiones geográficas para verificar si las series de tiempo presentaban patrones estacionales o tendencias que pudieran afectar la precisión de los modelos. Los resultados obtenidos respaldan la hipótesis de estacionariedad, lo que significa que las características estadísticas de los datos (media, varianza y autocorrelación) se mantienen constantes a lo largo del tiempo. Es importante aclarar que esta condición no es estrictamente necesaria para todas las técnicas de aprendizaje automático. En particular, las redes neuronales recurrentes (RNN) tienen la capacidad de capturar patrones en los datos, independientemente de si estos presentan estacionariedad o no. No obstante, en métodos tradicionales de series de tiempo, la estacionariedad sigue siendo un requisito fundamental para asegurar la precisión y consistencia de los modelos.

4.7. Análisis de modelos

Con los datos históricos entre 2019 y 2022, se generó el punto de partida para la configuración de los diferentes modelos a probar. La ilustración 6 muestra la serie de tiempo distribuida en meses. Es importante comprender que estos se segmentarán de acuerdo a los resultados de la agrupación y de esta forma comenzar con el uso de los modelos de predicción.

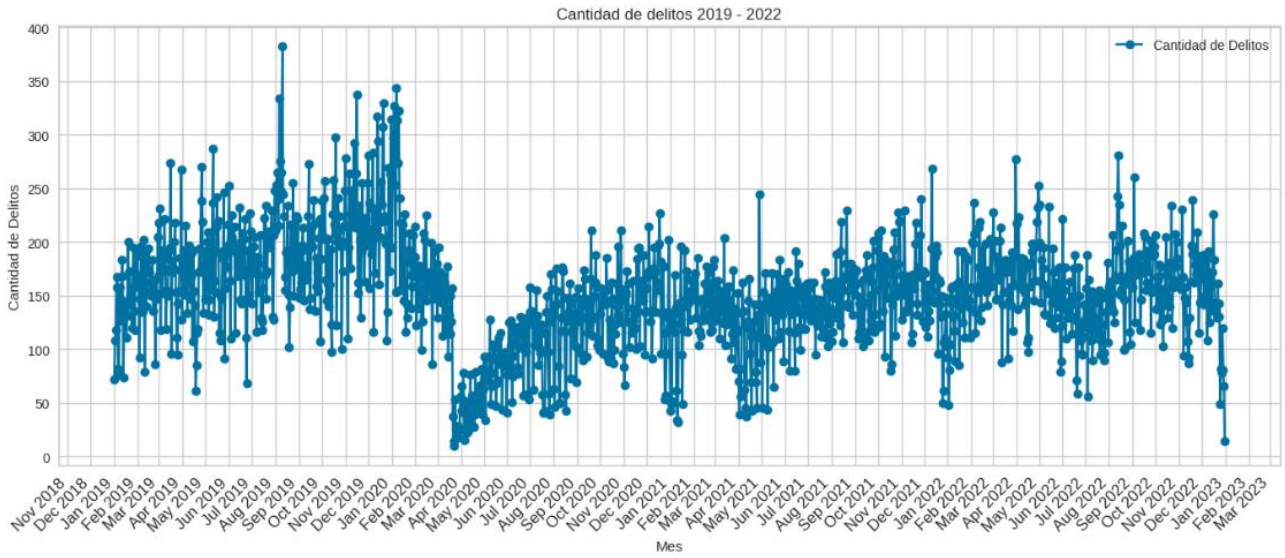


Ilustración 6. Serie temporal cantidad de delitos año 2019 a 2022.

4.7.1. Aplicación ARIMA

La selección de los parámetros p, d y q en el modelo ARIMA es un paso decisivo para garantizar su precisión y capacidad predictiva. Para llevar a cabo esta selección, se recurre al análisis de las funciones de autocorrelación (ACF) y autocorrelación parcial (PACF) de cada cluster que se está realizando el análisis. Estas funciones proporcionan información valiosa sobre la estructura de dependencia de una serie temporal.

De la misma forma, en la prueba de estacionariedad que se mencionó anteriormente, los resultados indicaron que se puede rechazar la hipótesis nula y asegurar que la serie es estacionaria. Esto es importante ya que el factor ‘d’ de diferenciación comenzaría en 0.

4.7.2. Aplicación SARIMA

Posterior al modelo ARIMA y con el orden establecido, se procedió a mejorar la capacidad predictiva del modelo con el modelo SARIMA. Este modelo tiene una propiedad adicional y son los componentes estacionales que se le pueden aplicar, esto con el fin de abordar patrones recurrentes en la serie temporal. Teniendo en cuenta las fechas de la serie temporal escogida, se configuró con una presencia de 12, que indica estacionalidad anual. En los resultados se analizará si esto presentó una ventaja significativa.

4.7.3. Aplicación ARMAX

Este modelo se enfocó en incluir variables externas, como lo son variables meteorológicas que agrupan temperaturas y precipitaciones, y por otro lado variables de control, cómo días de pago, fines de semana, días feriados. Estas variables en el modelo, permiten una mejor comprensión de los mecanismos subyacentes que genera la criminalidad y facilitan la identificación de posibles puntos de intervención para reducirla.

4.7.4. Redes Neuronales Recurrentes con LSTM

Para abordar de otra manera la predicción de delitos con estas series temporales, se configuró una red neuronal recurrente (RNN) con capas LSTM y dropout. Esta arquitectura se seleccionó debido a su capacidad para capturar patrones temporales complejos y a largo plazo, características inherentes a los datos de series temporales de delitos. Las capas LSTM fueron fundamentales para mitigar el problema del desvanecimiento del gradiente y permitir que la red "memorizara" información relevante a lo largo de la secuencia.

La configuración del modelo implicó una detallada selección de hiperparámetros. Se experimentó con diferentes combinaciones de unidades LSTM, tasas de dropout, tamaños de lote y número de épocas para encontrar la configuración óptima. El tamaño del lote determinó la cantidad de muestras procesadas en cada iteración del entrenamiento, mientras que el número de épocas controlaba el número total de iteraciones. La tasa de dropout se ajustó para prevenir el sobreajuste, evitando que el modelo se especializara demasiado en los datos de entrenamiento y perdiera capacidad de generalización.

Para evitar el sobreentrenamiento, se implementó la técnica de early stopping. Esta estrategia permitió detener el entrenamiento cuando el rendimiento del modelo en un conjunto de validación dejó de mejorar, evitando así que el modelo continuara aprendiendo ruido en los datos de entrenamiento. De esta manera, se garantizó que el modelo finalizado tuviera un buen desempeño en datos no vistos previamente.

Con el objetivo de aprovechar al máximo los datos disponibles y mejorar la eficiencia del proceso de modelado, se decidió entrenar modelos separados para cada cluster de delitos. Los pesos de estos modelos se guardaron para su posterior uso en otros cluster con características similares. Esta estrategia de transferencia de aprendizaje permitió aprovechar el conocimiento adquirido en cada subconjunto de datos y acelerar el proceso de entrenamiento de nuevos modelos.

5. RESULTADOS

Se presentan los resultados obtenidos a partir de los métodos de agrupamiento realizados y los modelos de predicción de delitos creados, estos evaluados con métricas específicas como RMSE, MAE y R2.

5.1. Agrupamiento con K-Means

Los métodos del codo y silueta resultaron ser herramientas fundamentales para determinar el número óptimo de clusters en el algoritmo K-means, estos fueron alimentados con la longitud y latitud de cada registro del histórico de delitos analizado en este trabajo. Estas técnicas proporcionaron criterios objetivos para la selección del parámetro k, lo cual fue esencial para obtener agrupaciones significativas.

El método del codo demostró ser particularmente útil al graficar la variación dentro de los clusters (WCSS) en función del número de clusters, es decir la variación de la suma de cuadrados dentro del grupo. Al observar la ilustración 4, se identificó un "codo" o punto de inflexión donde la disminución de la varianza se volvió levemente marginal. Este punto representó el equilibrio óptimo entre el número de clusters y la compacidad de los mismos, evitando así el sobreajuste del modelo.

Por su parte, el coeficiente de silueta proporcionó una medida cuantitativa de qué tan similar fue un objeto a su propio cluster en comparación con otros clusters. Los valores del coeficiente oscilaron entre -1 y 1, donde los valores cercanos a 1 indicaron que los objetos estuvieron bien agrupados, los valores cercanos a 0 sugirieron que los objetos pudieron pertenecer a clusters vecinos, y los valores negativos indicaron posibles asignaciones incorrectas.

De esta manera, se evitó la selección arbitraria del número de clusters y se optimizó la coherencia interna de los grupos, maximizando la separación entre clusters diferentes. Esta doble validación aseguró que los resultados obtenidos fueran confiables y representativos de la estructura subyacente de los datos.

Una vez determinado el valor óptimo, que se estableció en k=4, se procedió a implementar el algoritmo K-means. El valor k seleccionado se utilizó para inicializar el número exacto de centroides que actuarían como puntos de referencia para la formación de los clusters. Este

parámetro fue esencial ya que definió el agrupamiento final tal como se observa en la ilustración 7.

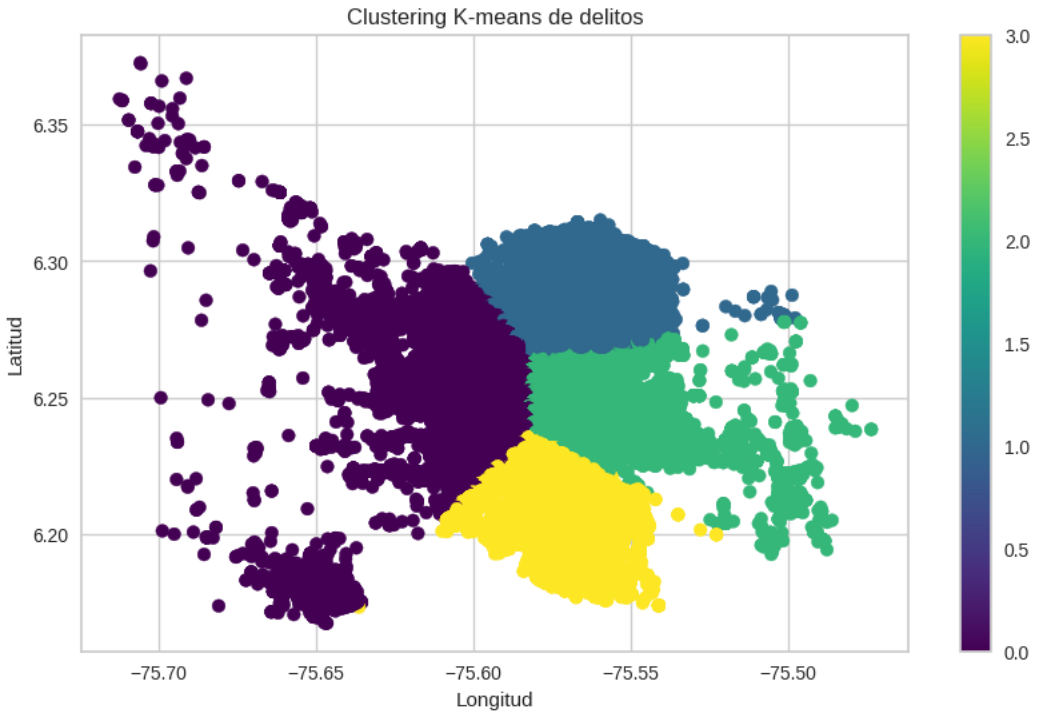


Ilustración 7. Clustering K-means para k=4.

Durante la fase de entrenamiento, el algoritmo K-means utilizó este valor k para distribuir aleatoriamente k centroides en el espacio de características. En cada iteración, el algoritmo asignó cada punto de datos al centroide más cercano y recalculó la posición de los centroides basándose en la media de todos los puntos asignados a cada cluster. Este proceso iterativo continuó hasta que los centroides se estabilizaron o se alcanzó el número máximo de iteraciones establecidas.

Este proceso es particularmente costoso computacionalmente, especialmente porque se está trabajando con un gran conjunto de datos y esto ocasiona que el algoritmo deba realizar más cálculos para asignar cada punto al cluster más cercano y actualizar los centroides.

5.2. Agrupamiento con DBSCAN

La implementación del algoritmo DBSCAN presentó un reto en hacer una precisa selección de sus dos parámetros fundamentales: 'eps' (epsilon), que definió el radio de vecindad para cada punto, y 'min_samples', que estableció el número mínimo de puntos necesarios para formar un

cluster denso. La determinación de estos valores se realizó mediante un análisis exploratorio de los datos y la utilización del método del vecino más cercano.

Para encontrar un valor apropiado de 'eps', se calcularon las distancias al vecino más cercano para cada punto y se generó un gráfico de distancias ordenadas. El punto de inflexión en este gráfico proporcionó una primera aproximación del valor óptimo de 'eps'. Por su parte, el parámetro 'min_samples' se estableció considerando la dimensionalidad de los datos y el tamaño mínimo significativo que debería tener un cluster para el contexto específico del problema, dando como resultado un eps = 0,3 y min_samples = 100.

En el uso de estos valores previamente calculados, se evidenció que el algoritmo no lograba realizar una correctamente separación de los datos, además de una gran cantidad de puntos clasificados como ruido (-1) tal como se evidencia en la ilustración 8.

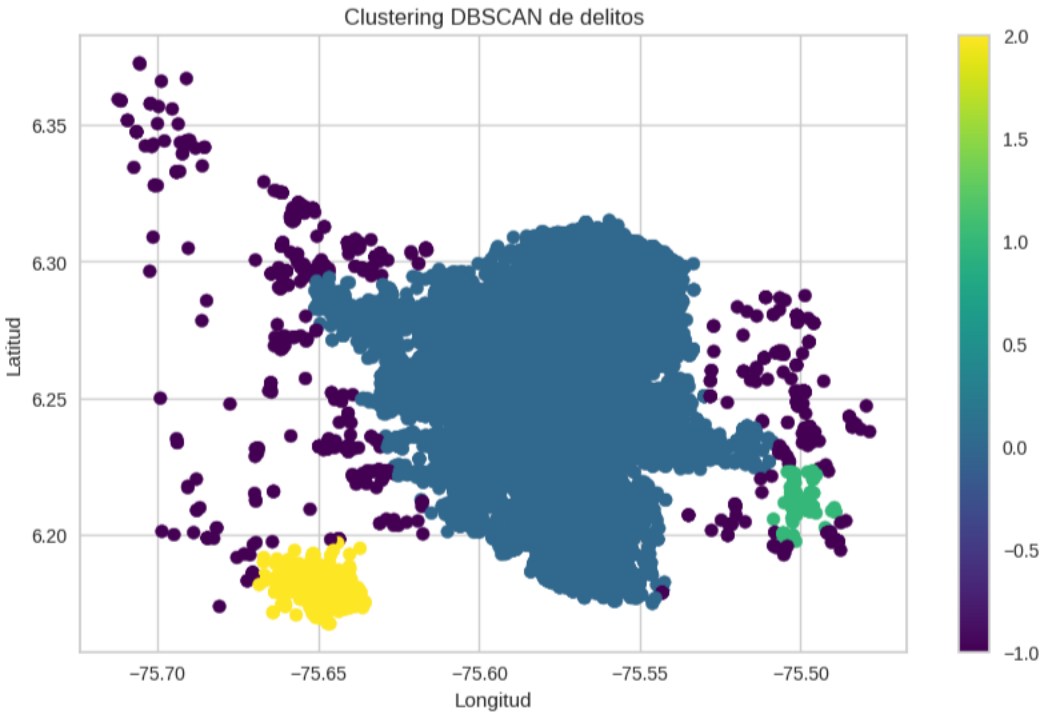


Ilustración 8. Clustering DBSCAN con eps=0,3 y min_samples = 100

Esto sugiere que estos valores no capturaron adecuadamente la densidad y distribución de los datos. Se realizaron más cálculos disminuyendo el mínimo de puntos para formar un cluster,

evidenciando la aparición de más clusters pero inmensamente desproporcionales, estos se evidencia en la ilustración 9, concluyendo que este agrupamiento no es adecuado para estos datos.

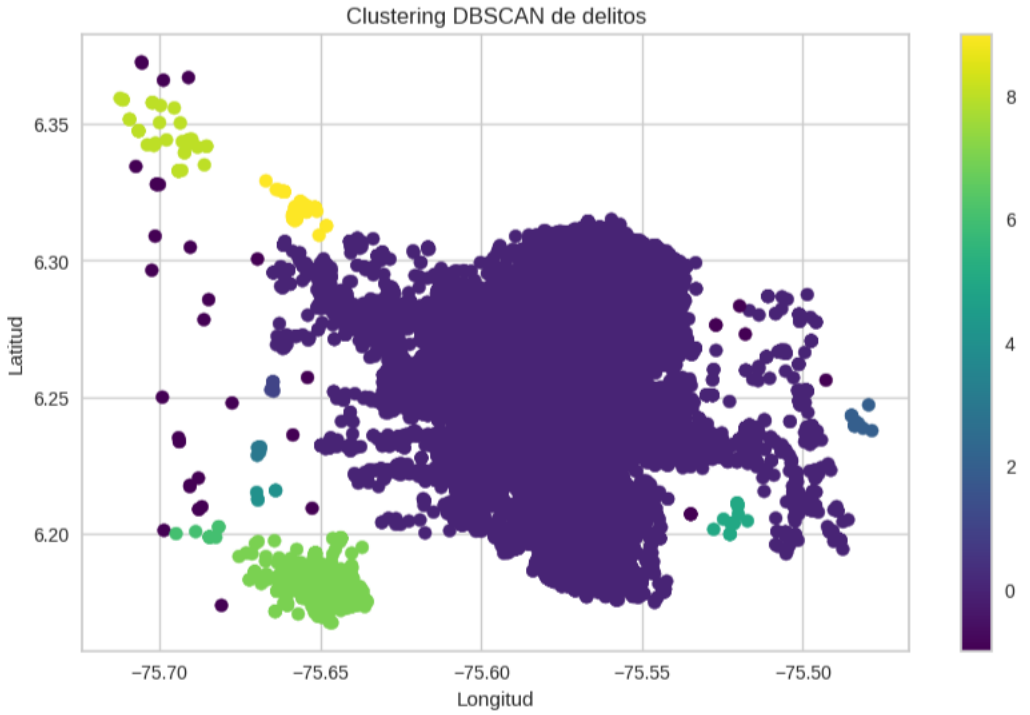


Ilustración 9. Clustering DBSCAN con eps=0,3 y min_samples = 5

5.2. Agrupamiento con BIRCH

Este modelo demostró ser particularmente efectivo para el manejo de nuestro conjunto de datos a gran escala. La característica más notable fue su capacidad para procesar los datos de manera incremental y su eficiente uso de la memoria, lo cual resultó ser crucial dado el volumen sustancial de información que se necesitaba analizar.

Una ventaja significativa que se observó fue la capacidad del algoritmo para manejar valores atípicos de manera natural. Los outliers, es decir, aquellos puntos muy extremos, fueron eficientemente identificados y aislados en clusters pequeños. Además, se observó una notable variación en la densidad y tamaño de los clusters, dando como resultado un valor k=11 cómo se evidencia en la ilustración 10.

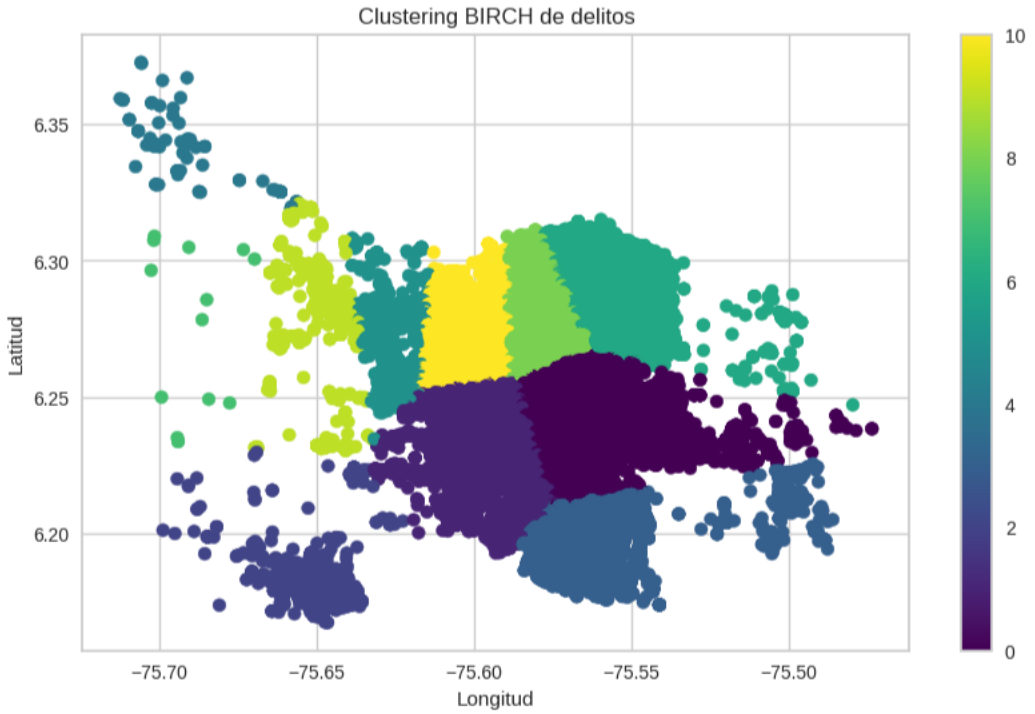


Ilustración 10. Clustering BIRCH

Esta configuración de agrupamiento, tiene como parámetros importantes el threshold y el n_clusters, definidos como 0,99 y 'None' respectivamente, donde el primero indica el máximo radio permitido para los subclusters, que significó tolerante a la hora de agrupar puntos de datos en un mismo subcluster, y el segundo permitiendo que el propio algoritmo determine automáticamente el número de clusters.

5.2. Selección de agrupamiento y clustering para predicciones

Con los algoritmos K-Means y BIRCH, que presentaron los mejores resultados, se procedió a realizar una comparación entre ambos revelando resultados similares en la distribución de algunos clusters, particularmente en las zonas de mayor densidad de datos, con una distribución espacial similar. Ambos métodos demostraron una capacidad sobresaliente para identificar y agrupar los patrones delictivos en las áreas más críticas de la ciudad, presentando una concordancia significativa en la delimitación del cluster principal.

La implementación de K-means, con el valor k óptimo determinado mediante los métodos del codo y silueta, generó una distribución de clusters que coincidió con los patrones identificados

por BIRCH. Esta similitud fue especialmente evidente en el cluster central, que en ambos casos agrupó la mayor concentración de incidentes delictivos, sugiriendo una robustez en la identificación de patrones independientemente del método utilizado.

A continuación, en las ilustraciones 11 y 12, se evidencian gráficamente la delimitación de todos los clusters K-Means y BIRCH respectivamente.

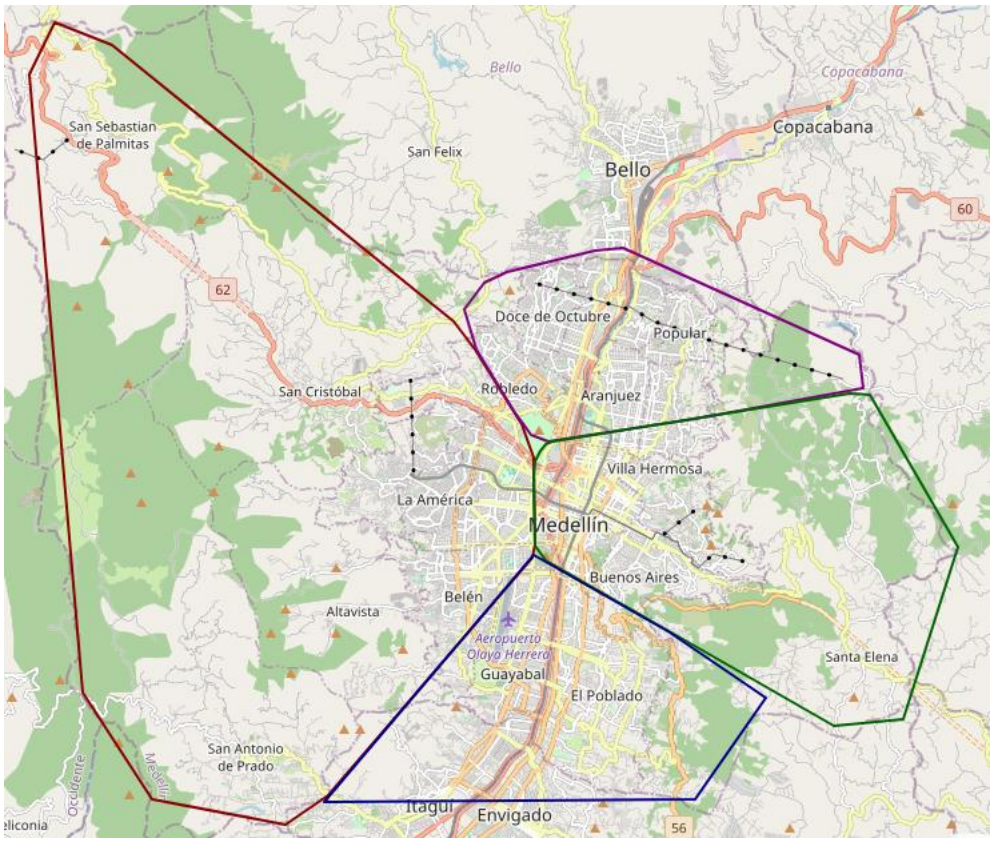


Ilustración 11. Delimitación de clusters para K-Means.

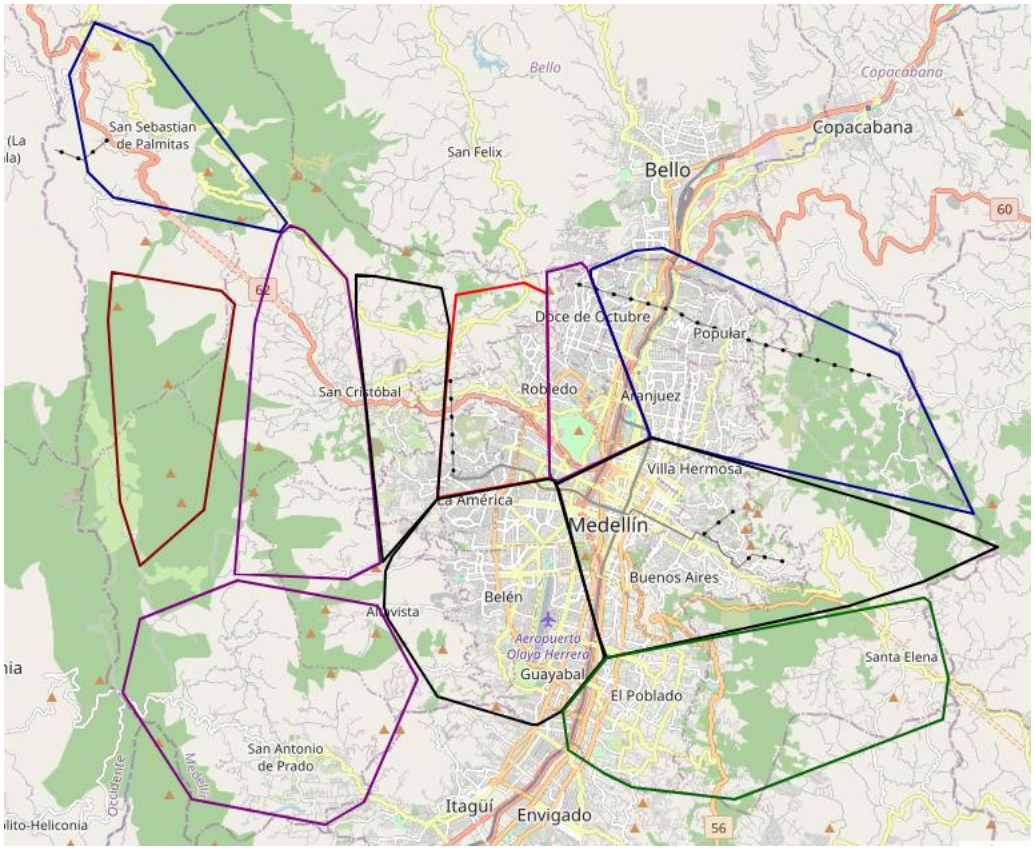


Ilustración 12. Delimitación de clusters para BIRCH.

A pesar de utilizar enfoques matemáticos diferentes, tanto K-means como BIRCH identificaron patrones espaciales consistentes con el cluster de mayor tamaño, ubicado en la zona central del área estudiada. Este mostró características similares en términos de densidad y distribución espacial en ambos métodos, validando la robustez de los patrones identificados, tal como se evidencia en la ilustración 13 de K-Means con el cluster 2 y en la ilustración 14 con el cluster 0.

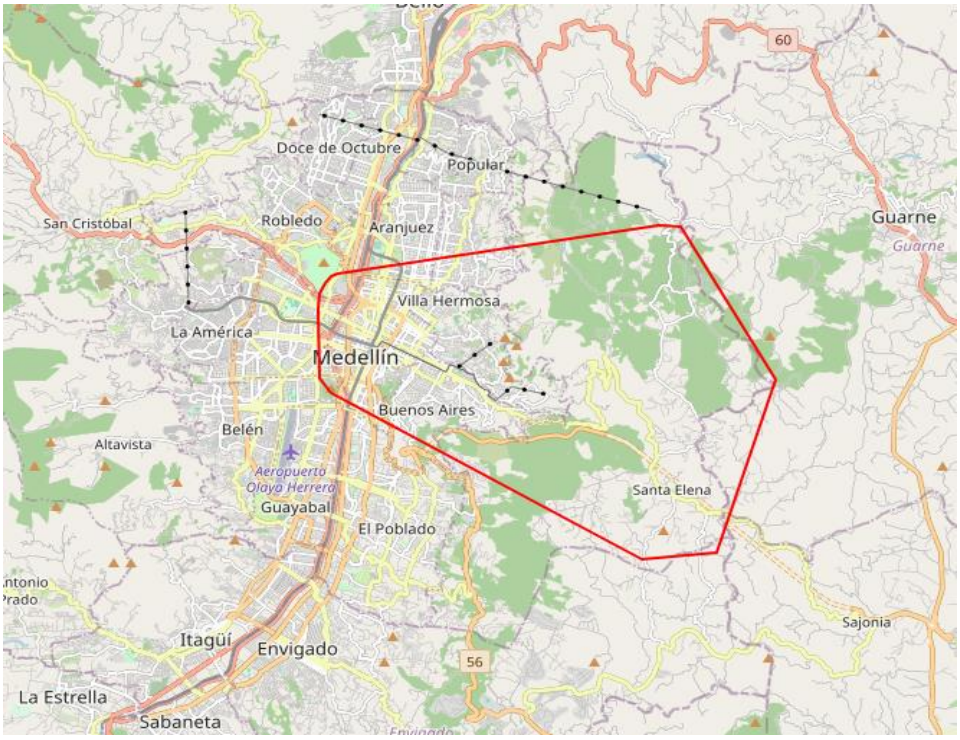


Ilustración 13. Delimitación cluster 2 de K-Means con mayor densidad de datos.

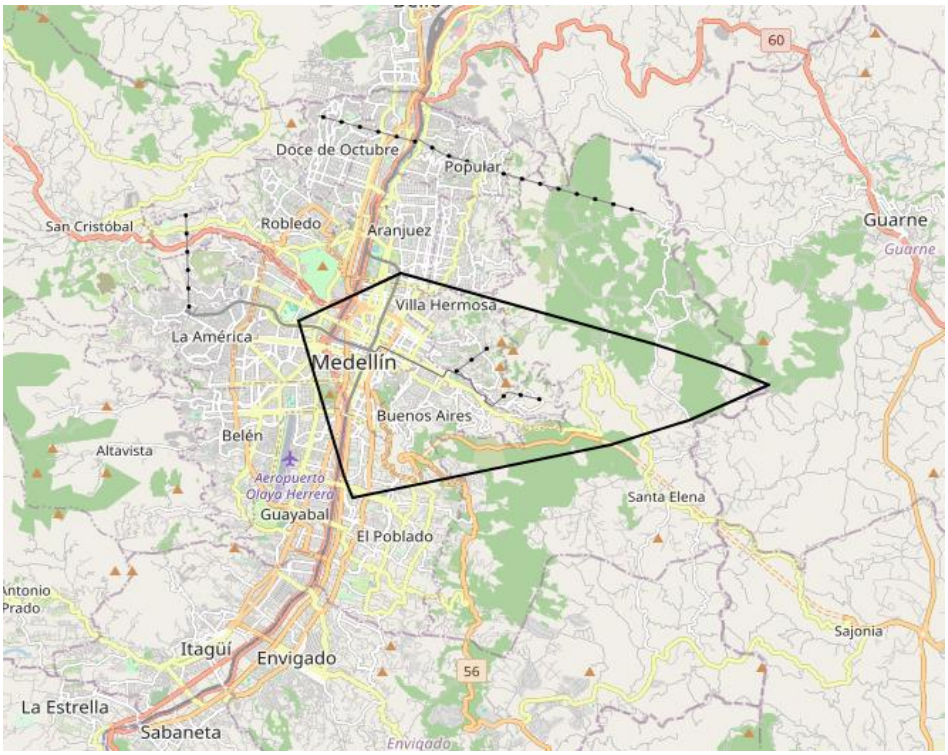


Ilustración 14. Delimitación cluster 0 de BIRCH con mayor densidad de datos.

Ahora bien, fue fundamental la prueba Dickey-Fuller realizada a ambos clusters para conocer si se rechazaba la hipótesis nula y verificar si la serie es estacionaria. Esto aseguraba la estacionariedad de los datos y permitía aplicar correctamente los modelos de forecasting y Machine Learning. Los resultados en la tabla 1, revelaron un valor de p estadísticamente significativo ($p<0,05$), indicando una fuerte evidencia en contra de la hipótesis nula de no estacionariedad para el cluster 2 de K-Means. Esto indica que es estacionaria y con diferenciación 0.

Algoritmo	Cluster	p-value	Data lenght
K-Means	2	0,0321828	81609
BIRCH	0	0,0587194	86832

Tabla 1. Resultados prueba Dicker-Fuller para cluster con mayor densidad de cada algoritmo.

En el caso de K-means, la naturaleza del algoritmo forzó una distribución más uniforme de los datos dentro de cada cluster debido a su método de centroide y distancia euclidiana. Esto resultó en una estacionariedad, ya que K-means tendió a crear clusters de tamaños similares y formas esféricas, incluso cuando los patrones subyacentes de los delitos podrían haber sido más irregulares o variables en el tiempo, es decir con una posible tendencia, esto es importante tenerlo en cuenta para los análisis en las series temporales de datos que se analicen.

Por otro lado, BIRCH mostró una distribución más realista de los datos debido a su estructura de árbol CF (Clustering Feature) y su capacidad para adaptarse a la naturaleza temporal de los delitos. El algoritmo fue capaz de capturar las variaciones naturales en la frecuencia de delitos a lo largo del tiempo, permitiendo que los clusters reflejaran los patrones temporales reales sin forzar una distribución uniforme.

Con los anteriores resultados, se pudo evidenciar que los clusters realizados por el método K-Means son los indicados para los modelos de forecasting y Machine Learning. Esto también se pudo confirmar realizando nuevamente la prueba Dickey-Fuller a todos los 4 clusters resultantes de este método, junto con sus respectivos valores críticos.

Cluster	ADF Statistician	p-value	Critical Value (1%)	Critical Value (5%)	Critical Value (10%)
0	-3,48319194	0,0084294	-3,4349056	-2,863552	-2,5678412
1	-4,539718473	0,00016658	-3,4348773	-2,8635395	-2,5678345
2	-3,030038013	0,03218286	-3,4349056	-2,863552	-2,5678412
3	-3,738497331	0,00360311	-3,434912	-2,8635548	-2,5678427

Tabla 2. Resultados prueba Dicker-Fuller para los 4 clusters de K-Means

En comparación con la anterior serie temporal donde se evidenciaron todos los delitos de toda la ciudad, se observó que presentan la misma forma, pero con una densidad menor, evidentemente por el filtrado a los registros asociados al cluster 2 tal como se evidencia en la ilustración 15.

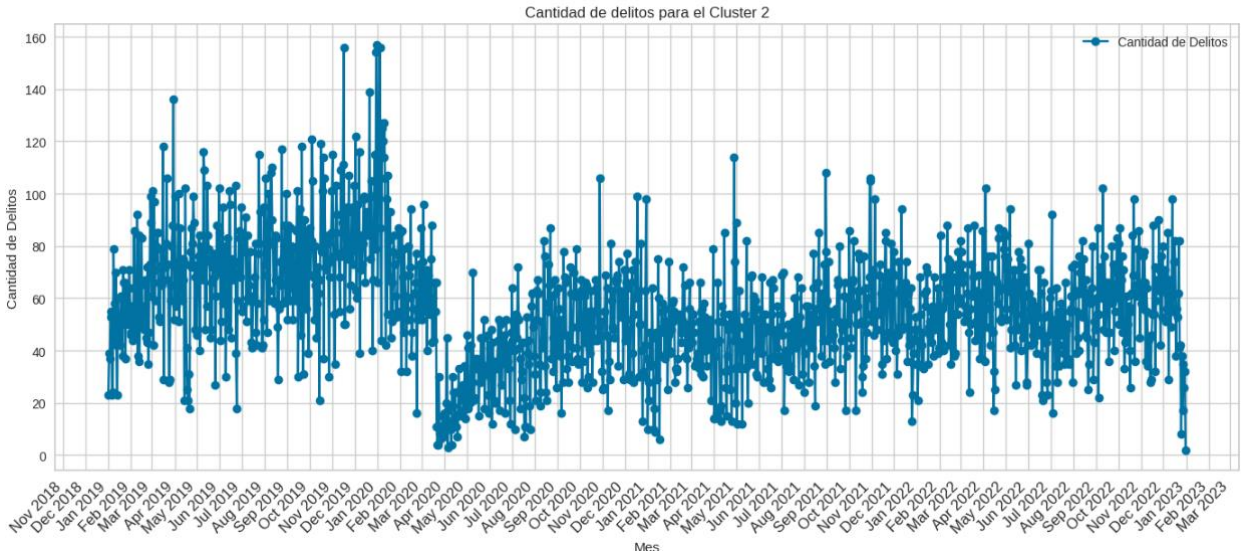


Ilustración 15. Serie temporal cantidad de delitos 2019 a 2022 para cluster 2.

Por último, en la ilustración 16 se evidencia el resultado de la validación de la Autocorrelación (ACF) y Autocorrelación Parcial (PACF) a la serie temporal, observándose importantes resultados para escoger el orden en el modelo ARIMA. En la gráfica PACF se analizó una disminución pronunciada después del primer o segundo rezago, lo que indicaba que el componente autorregresivo podría ser de orden p=1 o cómo máximo p=2. La gráfica ACF decae gradualmente, lo cual sugiere un componente de promedio móvil de orden q=1 o superior, dado

que no decae inmediatamente se puede considerar un q=2. La serie no presenta diferenciación por lo que se utilizó un valor d=0.

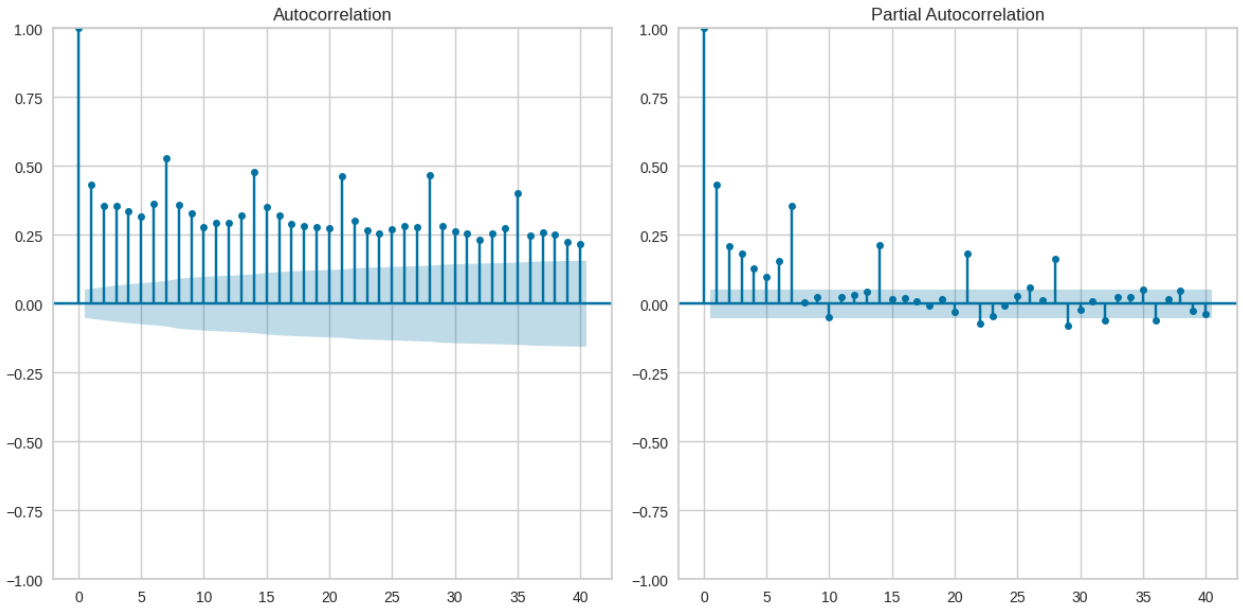


Ilustración 16. Gráficos ACF y PACF para ayudar a determinar el orden de ARIMA.

5.3. Resultados modelo ARIMA

Para este modelo se evaluaron varios órdenes, comenzando con el ARIMA (1,0,1) donde se refleja un proceso con un componente autorregresivo de primer orden y un componente de media móvil también de primer orden, que es adecuado para capturar la estructura temporal observada en los datos. Los resultados del RMSE, MAE y R2 indicaron que este modelo no lograba capturar bien la dinámica subyacente de los datos. Por otro lado, en validaciones con otros ordenes, el que mejor presentó resultados fue el ARIMA (1,1,0). Esto se evidenció por medio del análisis de residuos y el histograma de residuos que, entre todos, es el que mejor presenta una distribución normal. En las ilustraciones 19, 22, 25 y 28 se puede observar los histogramas de los residuos de cada uno de los clusters.

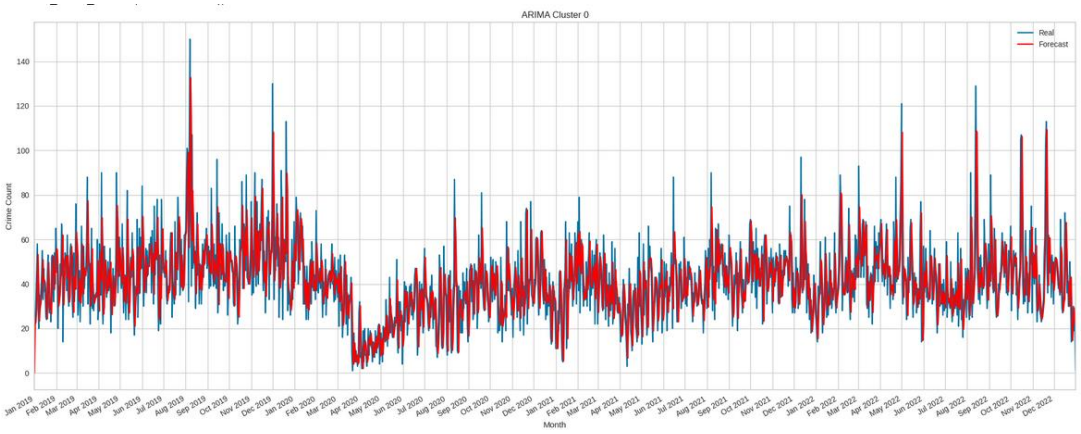


Ilustración 17. Forecasting ARIMA para cluster 0

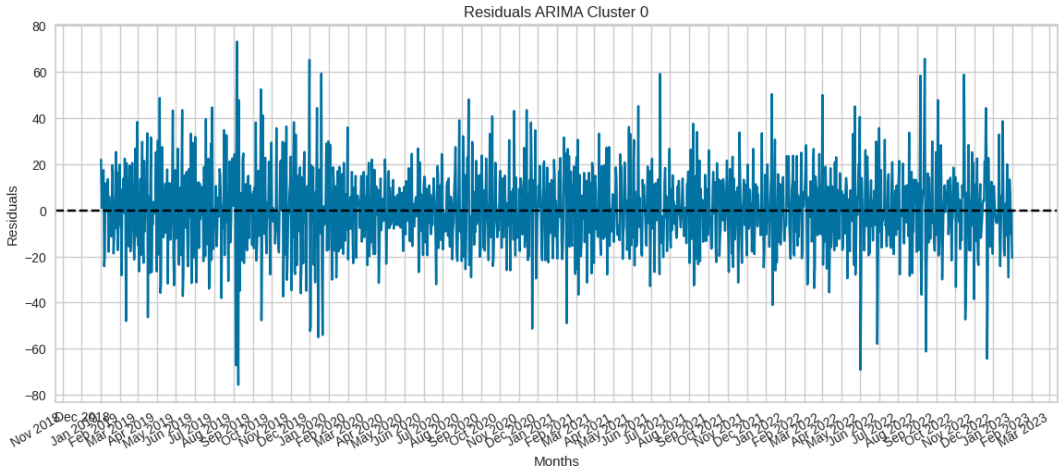


Ilustración 18. Residuos ARIMA para el cluster 2

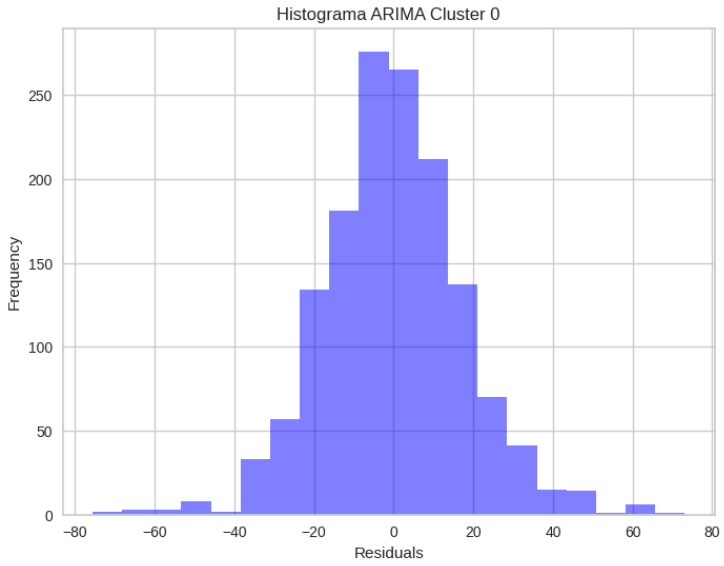


Ilustración 19. Histograma de residuos ARIMA para el cluster 2

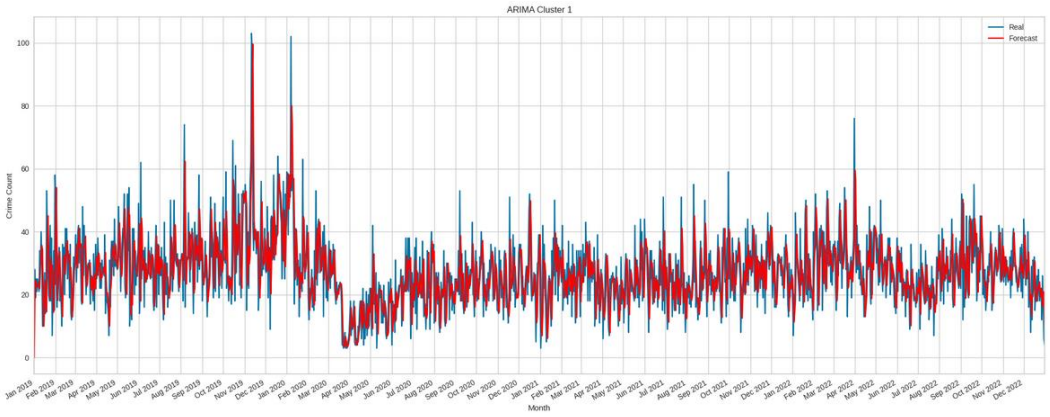


Ilustración 20.Forecasting ARIMA para cluster 1

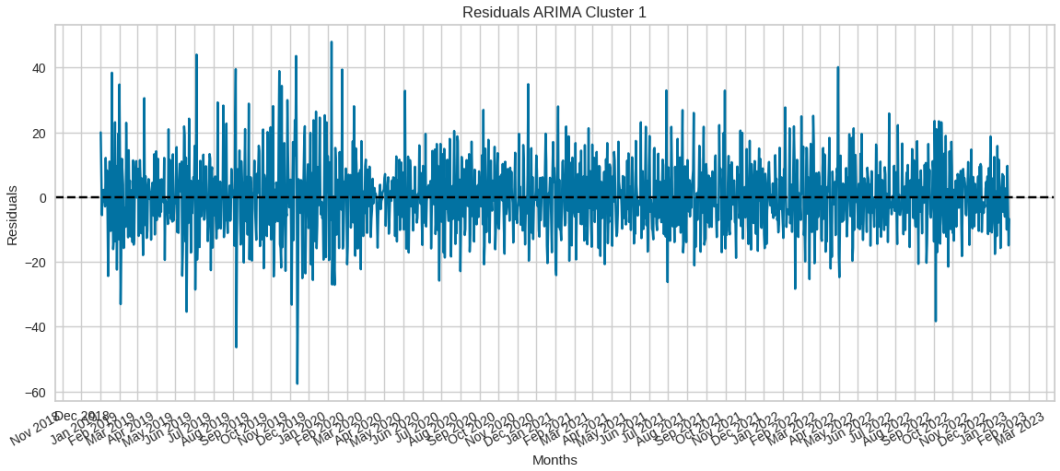


Ilustración 21. Residuos ARIMA para el cluster 1

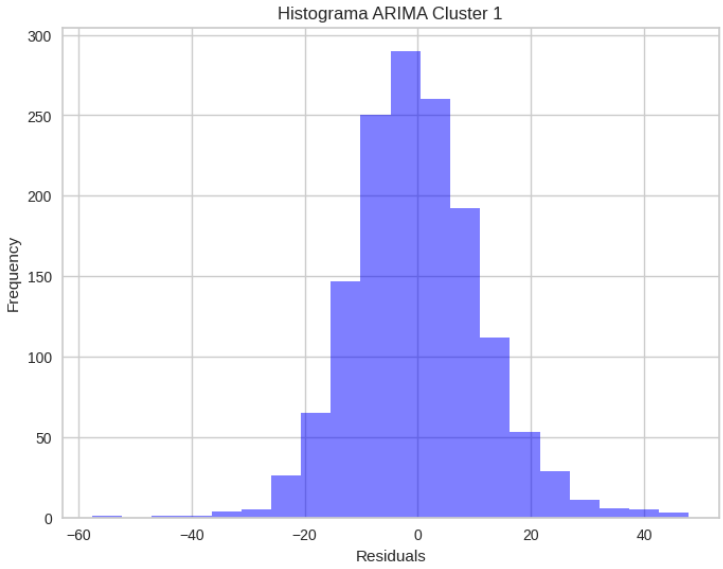


Ilustración 22. Histograma de residuos ARIMA para el cluster 1

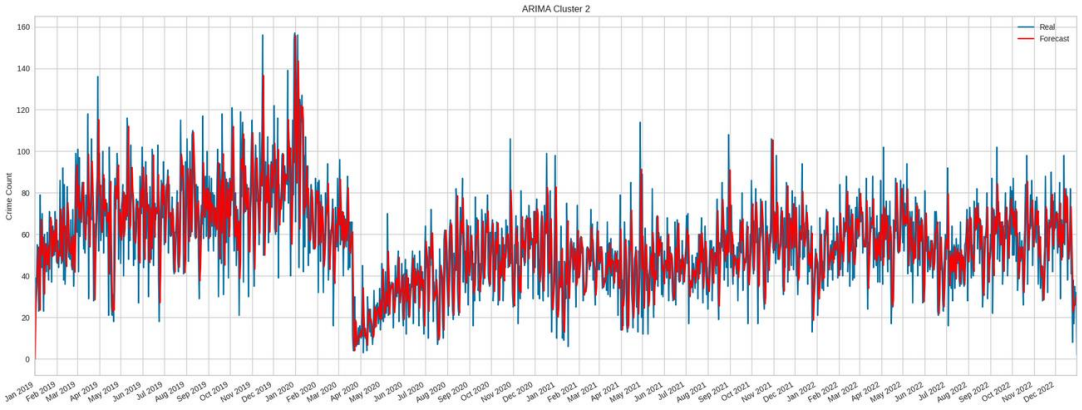


Ilustración 23. Forecasting ARIMA para cluster 2

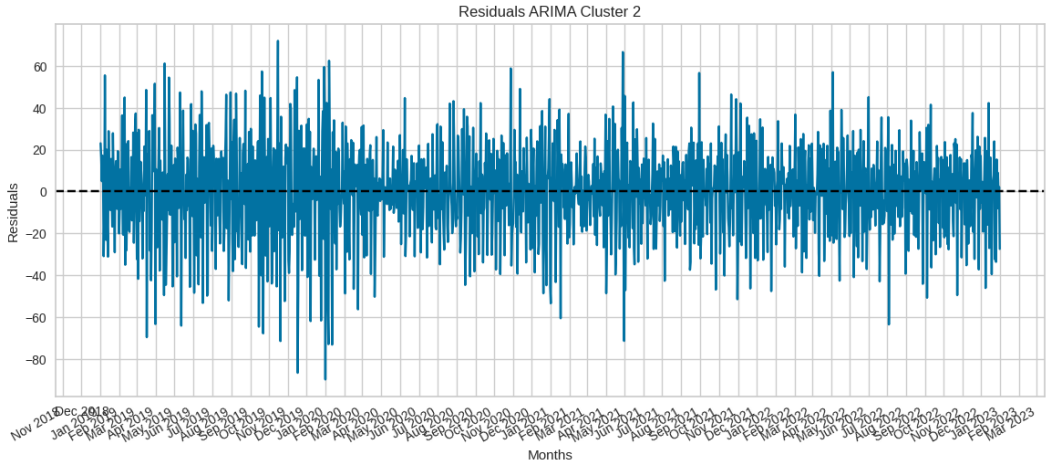


Ilustración 24. Residuos ARIMA para el cluster 2

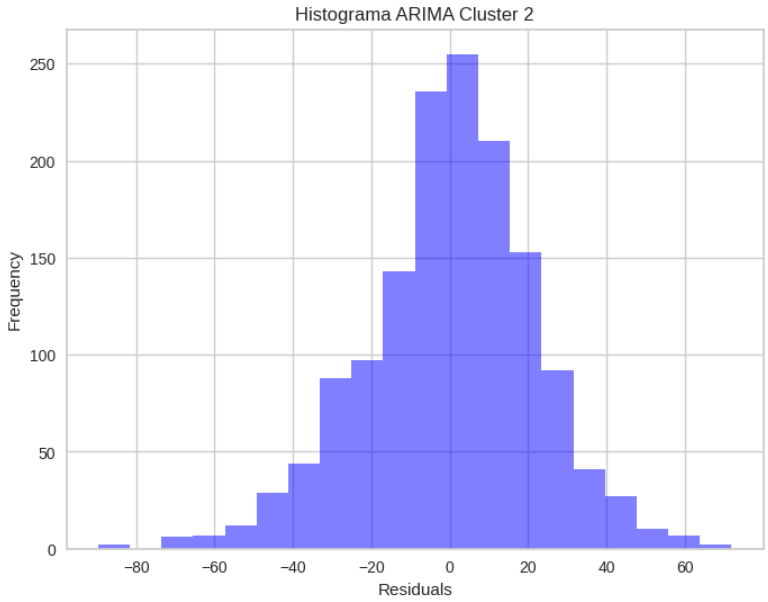


Ilustración 25. Histograma de residuos ARIMA para el cluster 2

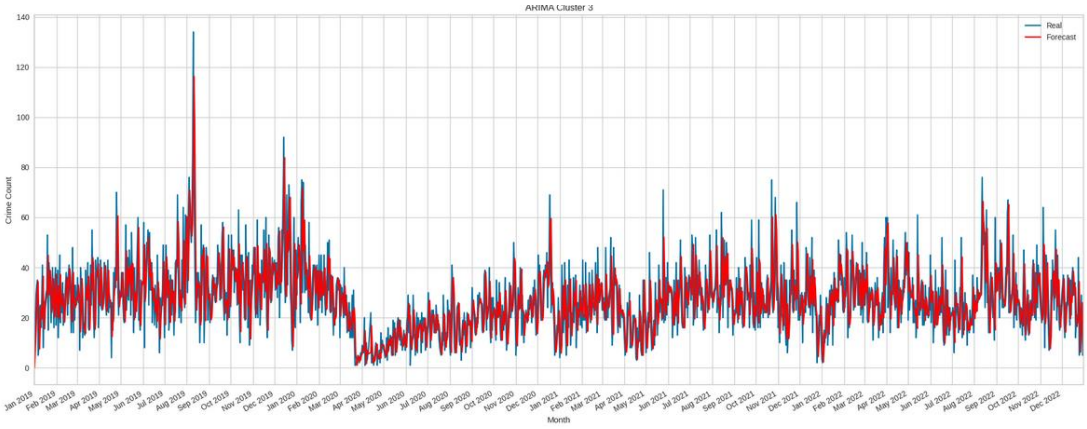


Ilustración 26. Forecasting ARIMA para cluster 3

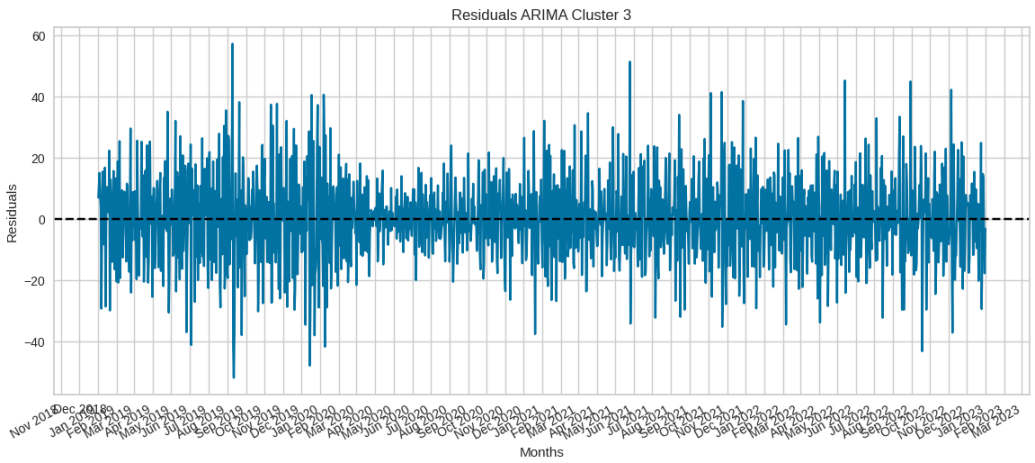


Ilustración 27. Residuos ARIMA para el cluster 3

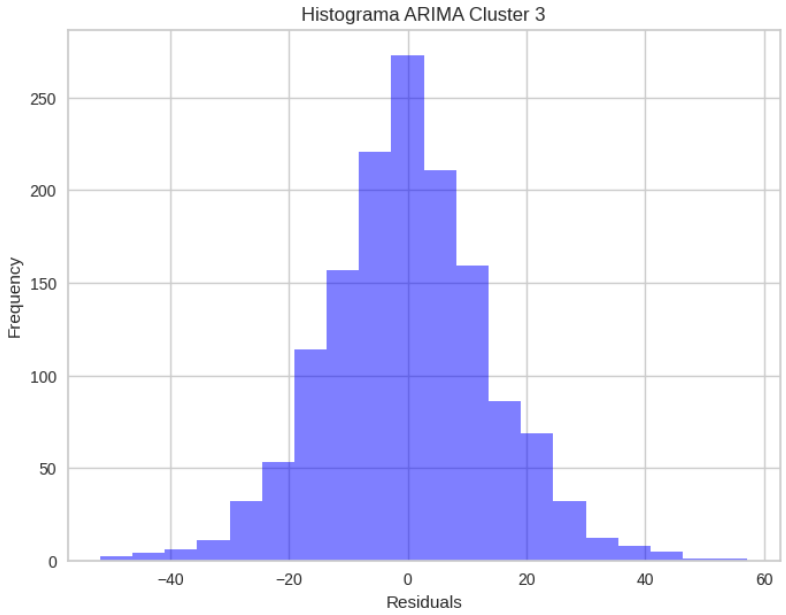


Ilustración 28. Histograma de residuos ARIMA para el cluster 3

5.4. Resultados modelo SARIMA

Este modelo presentó resultados interesantes donde se utilizó un orden (1,0,0) y con orden estacional de (1,0,0,12) indicando estacionalidad anual, pero sin presentar resultados altamente variables. Para todas las métricas de medición, a pesar de variar los órdenes de los modelos, se tuvo una variación en promedio de 2 puntos desfavorables para el modelo SARIMA. Esto puede indicar que los delitos no siguen un patrón estacional, y tienden a ocurrir de manera más aleatoria y menos predecible. Si bien factores sociales, económicos y culturales pueden influir en la ocurrencia de delitos, estos no necesariamente siguen ciclos predefinidos a lo largo de los meses o años. Es importante comprender que es posible que la variabilidad en la cantidad de delitos puede estar más relacionada con eventos esporádicos o situaciones contextuales, como crisis económicas o eventos sociales,

5.5. Resultados modelo ARMAX

Este modelo, al incluir variables exógenas, mostró resultados levemente mejores en comparación con el modelo ARIMA tradicional. Aunque la mejora no fue drástica, la incorporación de factores externos permitió afinar ligeramente la precisión en la predicción de delitos. Esto se debe a que los delitos, pueden estar influenciados por variables contextuales externas, como indicadores socioeconómicos, condiciones climáticas o eventos específicos en la ciudad. En la ilustración 29 se evidencia como el cluster 2 sigue la forma y escala de una manera más definida.

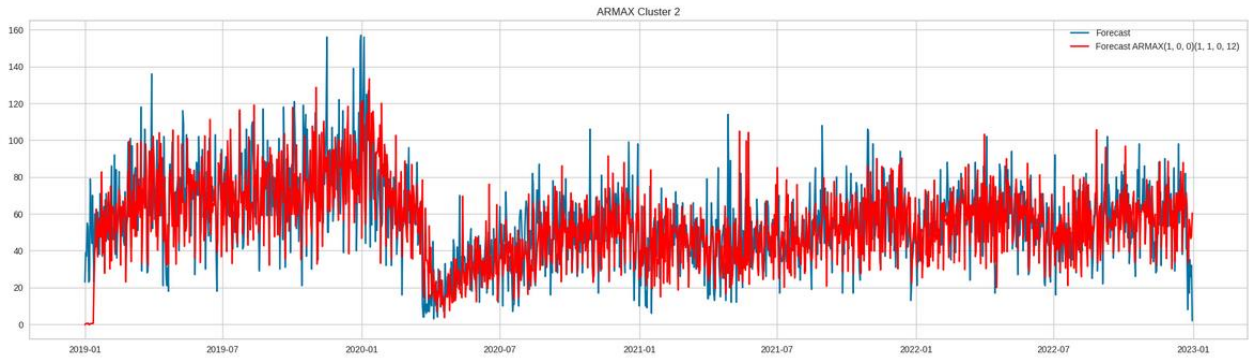


Ilustración 29. Forecasting para cluster 2 con el modelo ARMAX

En la tabla 3, se consolidaron todas las métricas obtenidas de los 3 modelos evaluados. Para la mayoría de los clusters, los modelos presentan valores altos de RMSE y MAE, lo que indica algunos errores en las predicciones. Esto se observa especialmente en los clusters 0 y 2, donde ninguno de los modelos logra un ajuste adecuado, reflejado también en los valores negativos de R^2 en varios casos. Estos valores negativos de R^2 sugieren que los modelos no están capturando bien la variabilidad en los datos, lo cual afecta la precisión de las predicciones.

Cluster	Modelo	RMSE	MAE	R2
0	ARIMA	17,5924	13,4518	-0,02153
	SARIMA	19,1364	14,6309	-0,20871
	ARMAX	18,6352	14,0445	-0,14623
1	ARIMA	11,3271	8,7235	-0,00446
	SARIMA	12,6026	9,7037	-0,24342
	ARMAX	11,9391	9,05898	-0,11594
2	ARIMA	21,8533	16,7708	0,08167
	SARIMA	23,688	18,6397	-0,07905
	ARMAX	19,7975	15,3325	0,24632
3	ARIMA	13,8719	10,6646	0,0000609
	SARIMA	14,8292	11,4150	-0,14207
	ARMAX	14,1393	10,6442	-0,03828

Tabla 3. Resultados de los modelos para los 4 clusters

5.5. Resultados modelo RNN LSTM

Este modelo presentó más dificultades al momento de configurarlo con sus hiperparámetros. Para mejorar la precisión en las predicciones de la cantidad de delitos, primero se aplicó una transformación logarítmica a los datos. Esta transformación tuvo como objetivo reducir la varianza en los datos y facilitar el aprendizaje del modelo al suavizar las fluctuaciones extremas, ya que los resultados estaban en escalas muy alteradas debido a la variabilidad de los datos.

Para la organización de los datos, el conjunto se dividió en tres grupos: entrenamiento, validación y prueba, con un 30% de los datos asignados para el entrenamiento, y el restante dividido equitativamente entre validación y prueba. Este asegura que el modelo se ajuste a los datos de entrenamiento y se evalúe tanto en validación como en prueba para comprobar su capacidad de generalización.

La estructura de la RNN LSTM se configuró con tres capas LSTM, cada una con 64 unidades y función de activación ReLU, intercaladas con capas de Dropout del 20% para evitar el sobreajuste. Finalmente, se añadieron capas densas, con una capa oculta de 128 unidades, seguida de una capa de salida de una unidad, que emite las predicciones. Esta configuración buscaba de manera efectiva las dependencias temporales y generar predicciones precisas a partir de los datos secuenciales. Además, se implementó `EarlyStopping` para monitorear la pérdida de validación y detener el entrenamiento si no se mejoraba después de 10 épocas consecutivas, lo que ayuda a evitar un ajuste excesivo del modelo y a preservar los mejores resultados. La gráfica en la ilustración 33 muestra como los datos se acoplan de manera escalada y ajustada en el cluster con mayor cantidad de delitos.

A continuación, en las ilustraciones 30, 31, 32 y 33, se evidencia la buena predicción que logra este modelo. Los valores RMSE, MAE y R2, brindan coherencia con los datos graficados, siendo los clusters 0 y 2 con mejor desempeño.

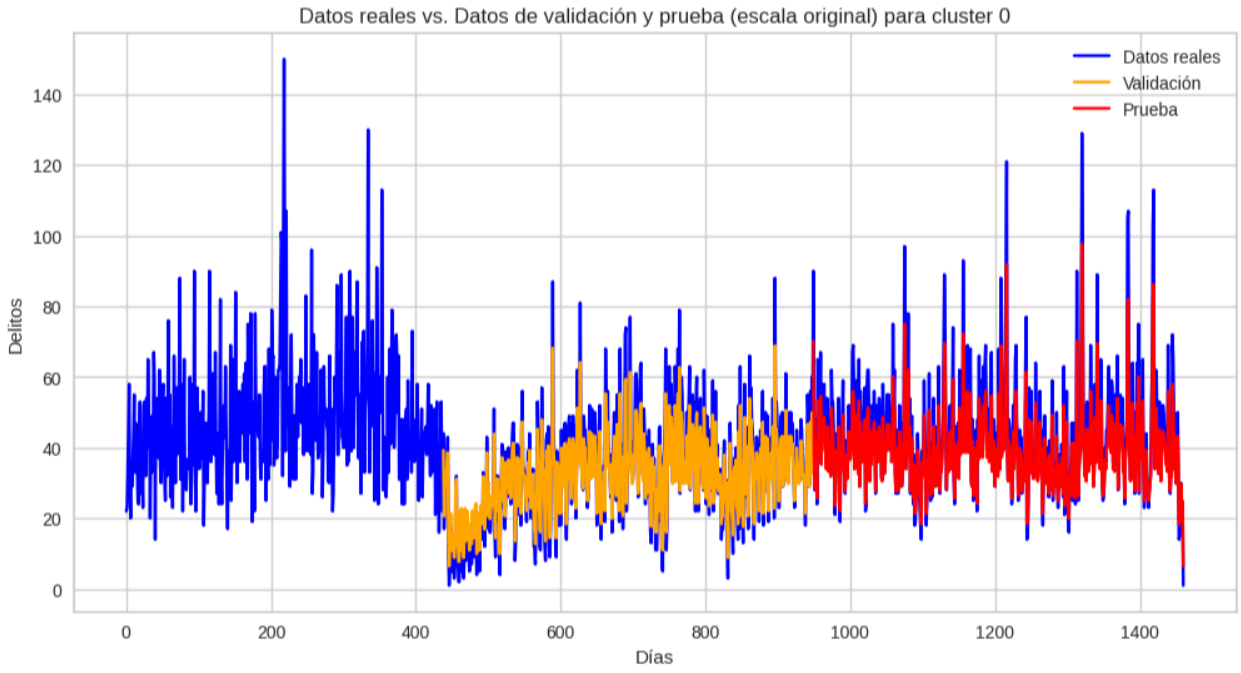


Ilustración 30. Predicción RNN STM para cluster 0

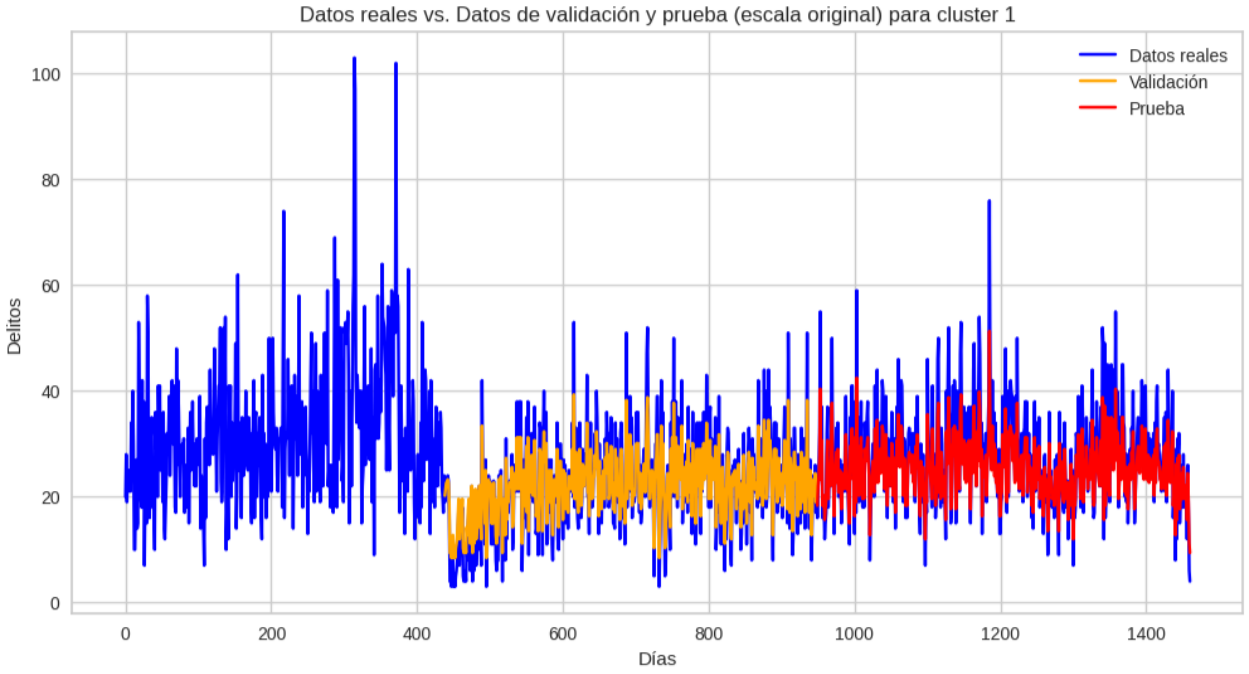


Ilustración 31. Predicción RNN STM para cluster 1

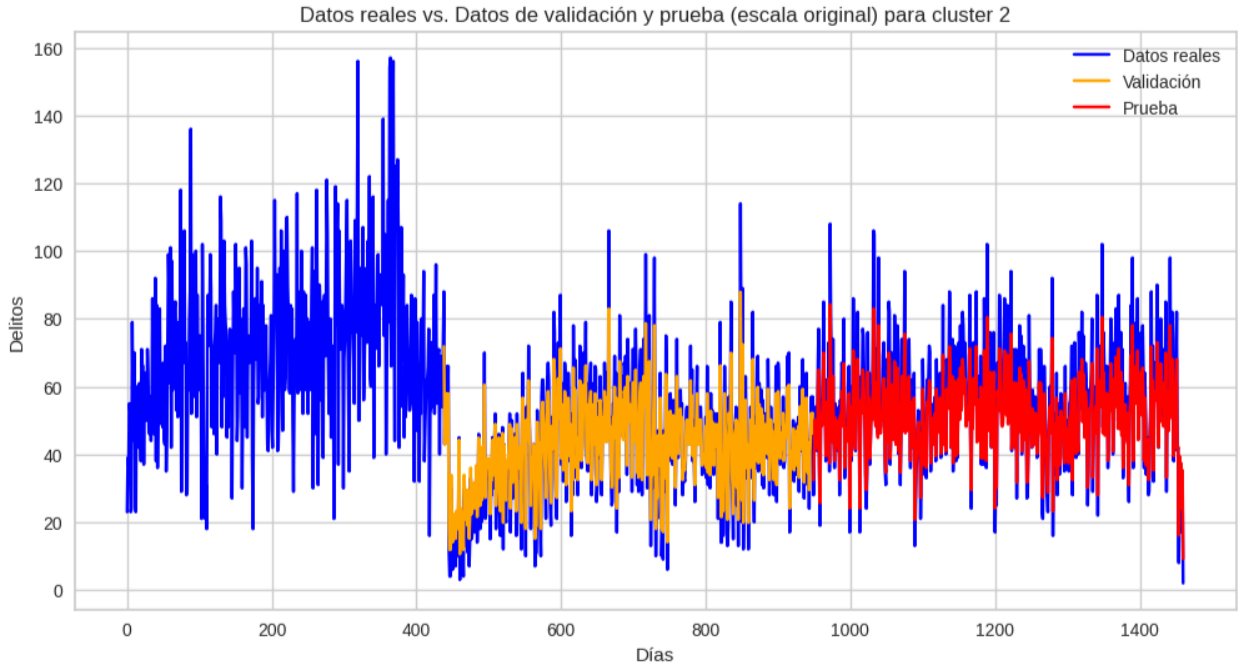


Ilustración 32. Predicción RNN STM para cluster 2

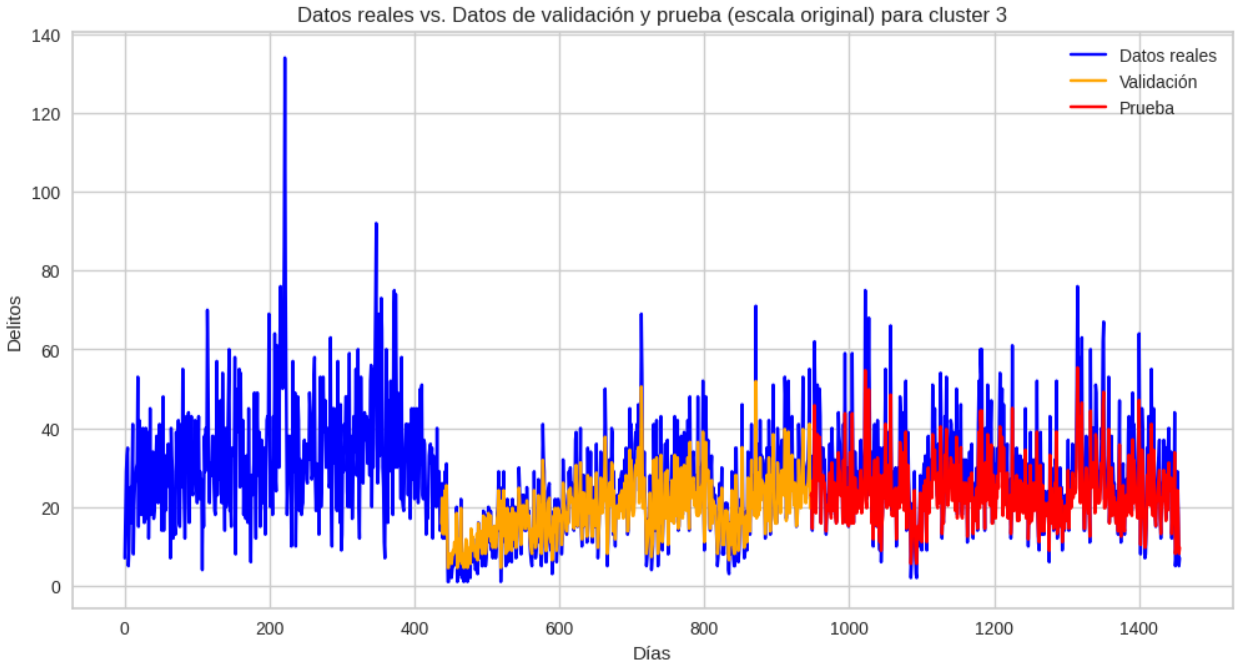


Ilustración 33. Predicción RNN STM para cluster 3

En la ilustración 34, se presenta el intervalo de confianza para el cluster con mejor desempeño, este indica el rango dentro del cual se espera que los valores reales caigan con un 30% de certeza. Esto ayuda a la incertidumbre en las predicciones del modelo, proporcionando una visualización clara del margen de error y la variabilidad esperada en las predicciones realizadas.

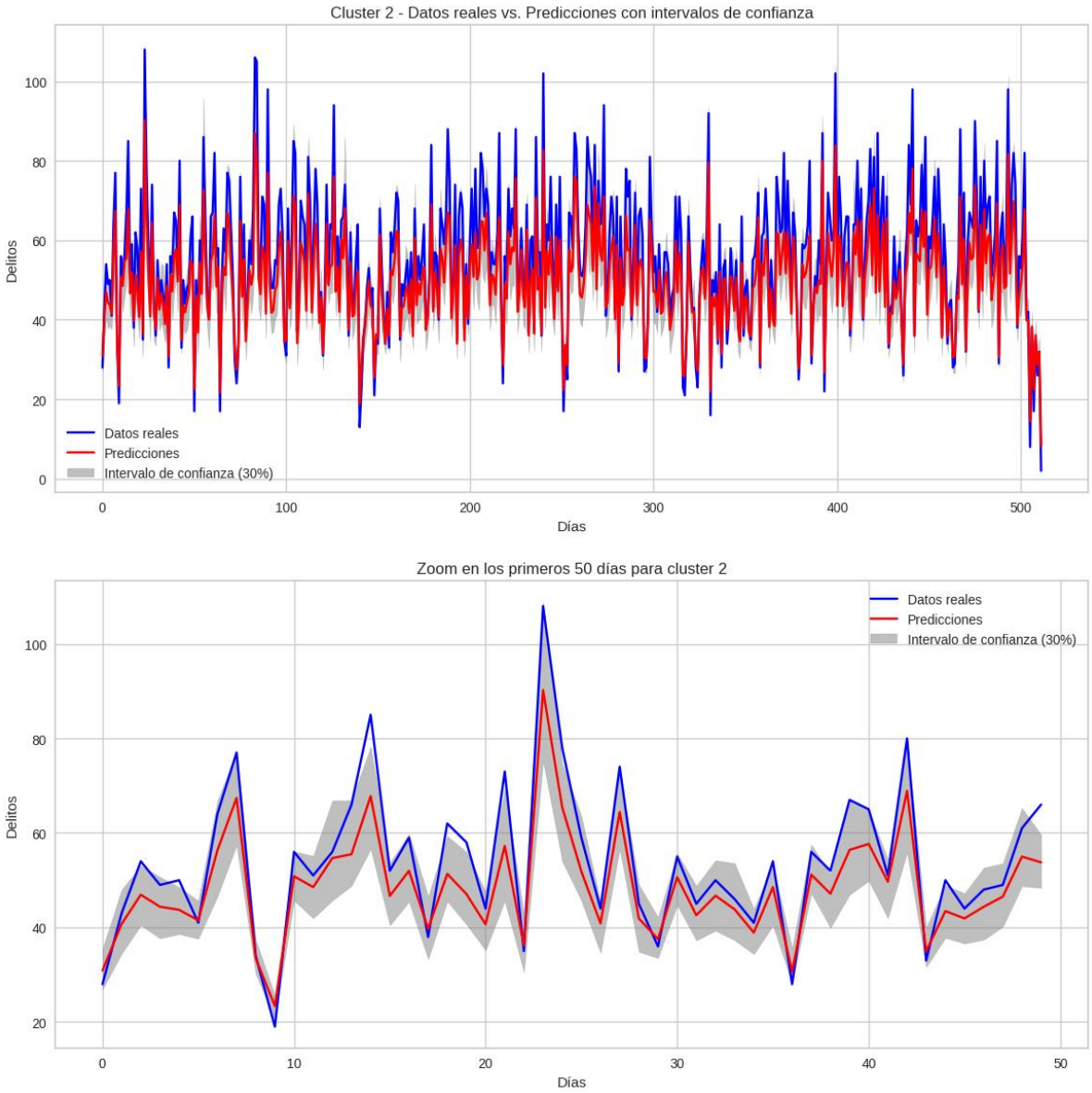


Ilustración 34. Predicción RNN LSTM con intervalo de confianza para cluster 2.

En la tabla 4, se presentan la consolidación de las diferentes métricas realizadas a todos los modelos, siendo RNN LSTM con el mejor desempeño en estas series de tiempo analizadas. Es importante destacar que cada serie de tiempo tiene características únicas, y su desempeño depende de múltiples factores asociados. Estas variables pueden influir tanto de manera positiva como negativa en los resultados del modelo.

Cluster	Modelo	RMSE	MAE	R2
0	ARIMA	17,5924	13,4518	-0,02153
	SARIMA	19,1364	14,6309	-0,20871
	ARMAX	18,6352	14,0445	-0,14623
	RNN LSTM	7,07489	5,27338	0,80655
1	ARIMA	11,3271	8,7235	-0,00446
	SARIMA	12,6026	9,7037	-0,24342
	ARMAX	11,9391	9,05898	-0,11594
	RNN LSTM	4,6631	3,48219	0,75323
2	ARIMA	21,8533	16,7708	0,08167
	SARIMA	23,688	18,6397	-0,07905
	ARMAX	19,7975	15,3325	0,24632
	RNN LSTM	7,48945	5,96073	0,80616
3	ARIMA	13,8719	10,6646	0,0000609
	SARIMA	14,8292	11,415	-0,14207
	ARMAX	14,1393	10,6442	-0,03828
	RNN LSTM	7,16433	5,21085	0,66089

Tabla 4. Consolidación resultados de todos los modelos.

6. CONCLUSIONES

Este trabajo se basó en el análisis comparativo de diferentes modelos para la predicción de delitos en la ciudad de Medellín, enfocándose en técnicas para el pronóstico de series temporales. Se realizaron análisis de agrupamiento utilizando tres algoritmos de clustering: K-means, DBSCAN, y BIRCH, que proporcionaron diferentes resultados en el agrupamiento de datos según ciertas características. Posteriormente, el trabajo se centró en la evaluación de 4 modelos de predicción, los cuales fueron ARIMA, SARIMA, ARMAX y RNN LSTM, para capturar tendencias en los datos. Las métricas RMSE, MAE y R2 permitieron medir su desempeño.

La incorporación de variables exógenas es importante en la predicción de delitos, ya que aporta información adicional sobre factores externos que podrían influir en la frecuencia de eventos delictivos en diferentes regiones y contextos. Factores como el clima, la temperatura, días de pagos, fines de semana y otros patrones de comportamiento pueden impactar en la frecuencia y tipo de delitos cometidos. Modelos como ARMAX, que permiten la inclusión de estas variables, han mostrado un desempeño adecuado, pero los resultados sugieren que el modelo RNN LSTM puede captar mejor las relaciones no lineales y complejas entre variables cuando se integran adecuadamente. En futuros estudios, la integración de variables exógenas relacionadas, como indicadores económicos o índices de población, podría optimizar las predicciones y proporcionar una visión más completa del fenómeno delictivo.

Los hiperparámetros indicaron ser un elemento importante en el desempeño de todos los modelos implementados. En el caso de la RNN LSTM, la configuración de parámetros como el número de neuronas en cada capa LSTM, el número de capas, la tasa de dropout y la función de activación han permitido mejorar la precisión de las predicciones. El ajuste adecuado de estos hiperparámetros ha sido sustancial para lograr un equilibrio entre la capacidad del modelo para aprender patrones complejos y su capacidad de generalización. Esta importancia también se refleja en modelos tradicionales como ARIMA y SARIMA, donde la selección de parámetros específicos de estacionalidad y orden ha influido directamente en los resultados. Dedicar tiempo a optimizar los hiperparámetros permite aprovechar al máximo el potencial de cada modelo y ajustarlos mejor a las características de los datos de delitos.

Los resultados obtenidos muestran que el modelo RNN LSTM se destacó en todos los clusters, presentando menores valores de RMSE y MAE en comparación con los modelos ARIMA, SARIMA y ARMAX. Esto indica que el RNN LSTM es más eficaz en la captura de patrones no lineales y complejos en los datos de delitos, superando a los modelos estadísticos en la mayoría de los casos. Los valores de R2 también respaldan este hallazgo, ya que el modelo LSTM obtiene coeficientes más cercanos a 1, indicando una mayor capacidad de ajuste en las predicciones. La diferencia en el desempeño evidencia la capacidad del modelo de redes neuronales recurrentes para adaptarse a la naturaleza no lineal de las series de tiempo delictivas.

Por lo tanto, este trabajo indica la relevancia de utilizar datos de calidad, identificar patrones en el agrupamiento de datos, explorar diferentes configuraciones de hiperparámetros para los modelso de predicción y considerar factores externos que puedan influir en el comportamiento delictivo. Si bien un modelo de predicción demostró ser el más adecuado en este estudio, la combinación de enfoques y la integración de variables exógenas podrían diversificar más la precisión de las predicciones.

Departamento de Ingeniería Electrónica y de Telecomunicaciones

Evaluación de modelos de Machine Learning en polígonos irregulares en la ciudad de Medellín para la predicción de delitos

UNIVERSIDAD DE ANTIOQUIA
1803
Facultad de Ingeniería

PRACTICANTE: Gabriel Antonio Lopera Madrid

ASESOR: Jaime Alberto Vergara Tejada

PROGRAMA: Ingeniería de Telecomunicaciones

Semestre de la práctica: 2024-1

Introducción

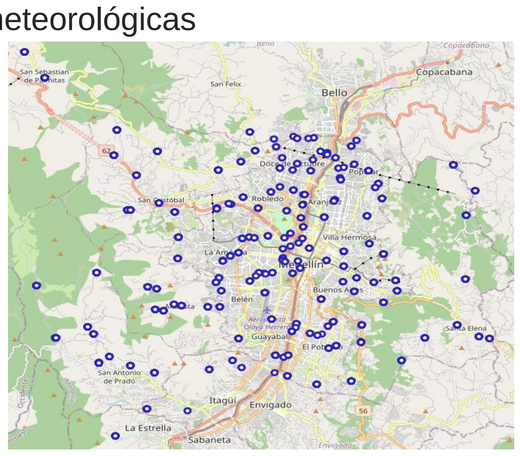
La seguridad ciudadana es esencial para el bienestar y desarrollo de las sociedades, permitiendo a los ciudadanos vivir sin temor a la violencia. En Medellín, aunque ha habido avances significativos en seguridad, persisten retos importantes. Actualmente, el análisis de los datos de seguridad se realiza manualmente, lo que retrasa las respuestas oportunas. Este trabajo propone una solución basada en el uso de datos históricos de delitos y factores externos, como condiciones meteorológicas y socioeconómicas, utilizando algoritmos de agrupamiento y modelos de Machine Learning. Con esto, se pretende identificar patrones y realizar predicciones de incidencia delictiva, facilitando herramientas para que las autoridades y la ciudadanía puedan actuar de manera preventiva, responsable y segura.

Objetivos

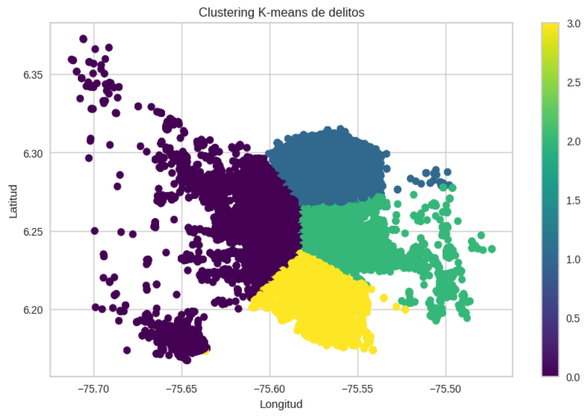
- Seleccionar e integrar datos históricos de la plataforma pública Medata con variables socioeconómicas y meteorológicas.
- Definir polígonos irregulares a partir de la distribución de delitos utilizando diferentes modelos de agrupamiento.
- Implementar y ajustar diferentes algoritmos de ML utilizados para la predicción de delitos.
- Evaluar la precisión de los modelos desarrollados por medio de las métricas: RMSE, MAE, R^2.

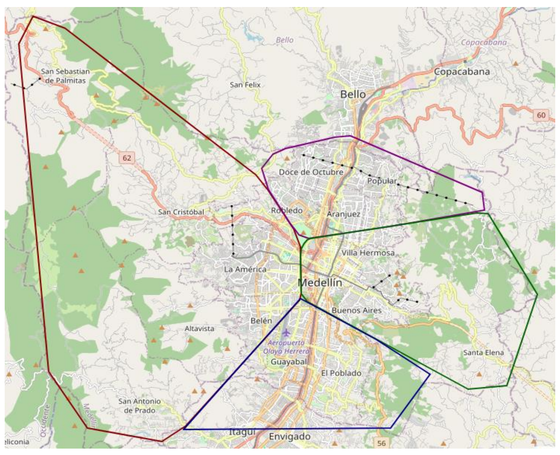
Metodología

- Preprocesamiento de datos de delitos
- Recopilación y distribución de variables meteorológicas

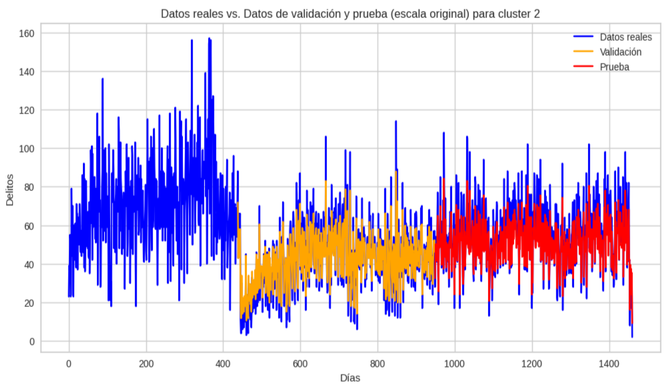


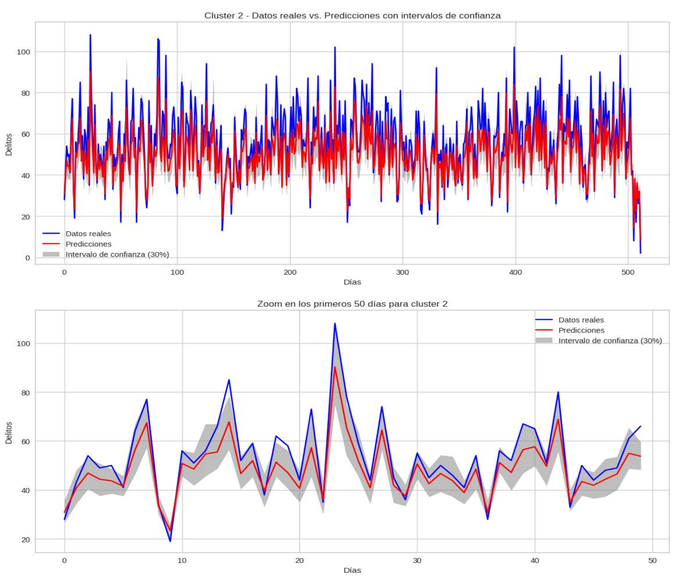
- Implementación algoritmos de agrupación





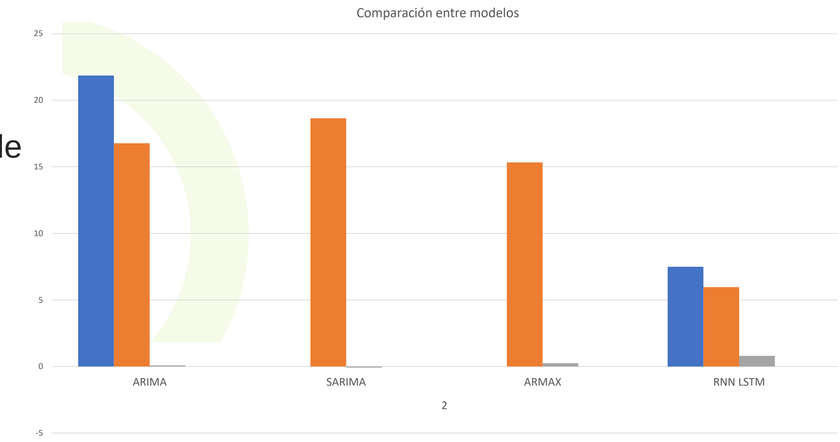
- Análisis de estacionariedad y autocorrelación de la serie temporal.
- Análisis e implementación algoritmos de agrupación





Resultados

- K-Means fue el algoritmo de agrupación que presentó mejores resultados.
- El modelo RNN LSTM, fue el que mejor se adaptó a la escala de datos. Lo resultados fueron coherentes
- Los modelos ARIMA, SARIMA, ARMAX presentaron predicciones aceptables pero con gran margen de mejora.
- Autocorrelación brindan el orden adecuados de los modelos.



Conclusiones

- Es importante de contar con datos precisos y completos sobre los delitos para obtener predicciones confiables.
- La incorporación de variables exógenas es importante en la predicción de delito
- Factores como el clima, la temperatura, días de pagos, fines de semana y otros patrones de comportamiento pueden impactar en la frecuencia y tipo de delitos cometidos.
- RNN LSTM puede captar mejor las relaciones no lineales y complejas entre variables cuando se integran adecuadamente.

DATOS DE CONTACTO DEL AUTOR:

+57 3104586366

antonio.lopera@udea.edu.co

https://www.linkedin.com/in/gabriel-lopera-madrid/

REFERENCIAS

[1] Kurniawan TB. The Clustering Algorithm with Geolocation data - TheLorry Data, Tech & Product - Medium. Medium. https://medium.com/thelorry-product-tech-data/the-clustering-algorithm-with-geolocation-data-d6dd07ed36a. Published enero 7, 2022.

[2] Na, S., Xumin, L., & Yong, G. (2010). Research on K-Means Clustering Algorithm: An improved K-Means Clustering Algorithm. Third International Symposium On Intelligent Information Technology And Security Informatics. https://doi.org/10.1109/iitsi.2010.74

[3] Jauregui AF. DBSCAN en Python: aprende cómo funciona. Ander Fernández. https://anderfernandez.com/blog/dbscan-python/. Published 19 de agosto de 2023.

[4] Zhang, T., Ramakrishnan, R., & Livny, M. (1996). BIRCH: an efficient data clustering method for very large databases. ACM sigmod record, 25(2), 103-114.

[5] Generator, M. (s. f.). View of Analysis of Elbow, Silhouette, Davies-Bouldin, Calinski-Harabasz, and Rand-Index Evaluation on K-Means Algorithm for Classifying Flood-Affected Areas in Jakarta. https://jurnal.polibatam.ac.id/index.php/JAIC/article/view/4947/2017

[6] Sagala, N. T. M., & Gunawan, A. A. S. (2022). Discovering the Optimal Number of Crime Cluster Using Elbow, Silhouette, Gap Statistics, and NbClust Methods. ComTech Computer Mathematics And Engineering Applications, 13(1), 1-10. https://doi.org/10.21512/comtech.v13i1.7270

[7] G. E. Box, G. M. Jenkins, G. C. Reinsel y G. M. Ljung, "Análisis de series temporales: previsión y control," John Wiley e hijos, 2015

[8] Box, G. E., Jenkins, G. M., Reinsel, G. C., & Ljung, G. M. (2015). Time Series Analysis: Forecasting and Control (5th ed.). Wiley.

[9] Box, G. E., Jenkins, G. M., & Reinsel, G. C. (2015). Time Series Analysis: Forecasting and Control. Wiley

[10] "Series Temporales, Modelo ARIMA Metodología de Box - Jenkins". Estadística.net. [en línea]. Consultado: 2024-03-09. Disponible en: https://www.estadistica.net/ECONOMETRIA/SERIES-TEMPORALES/modelo-arima.pdf

[11] Hyndman, R. J., & Athanasopoulos, G. (2018). Forecasting: Principles and Practice (2nd ed.). OTexts.

[12] Das, K., & Das, K. (2023, 17 octubre). How Recurrent Neural Network (RNN) works - Dataaspirant. Dataaspirant - A Data Science Portal For Beginners. https://dataaspirant.com/how-recurrent-neural-network-rnn-works/

[13] Areiza Jiménez, Juan Pablo (2024). Evaluación del desempeño de diferentes modelos de aprendizaje automático para la predicción de delitos en la ciudad de Medellín. Proyecto de investigación. Disponible en: https://bibliotecadigital.udea.edu.co/dspace/handle/10495/41488?mode=full