MODELO PREDICTIVO DE ZONAS DE RIESGO ESPACIO – TEMPORAL DE ACCIDENTES DE TRÁFICO EN LA CIUDAD DE MANIZALES

Juan Camilo Cardona Álvarez

Trabajo de grado presentado como requisito para optar por el título de:
Magister en Ingeniería Computacional

Directora:
Ing. MARÍA HELENA MEJÍA S., PhD.

Universidad de Caldas
Facultad de Ingenierías
Maestría en Ingeniería Computacional
Manizales 2023

A mi novia por su apoyo incondicional desde que decidí embarcarme en este proyecto. A mi familia y amigos por contribuir de diferentes formas para alcanzar la determinación necesaria y llegar hasta aquí.

RESUMEN

Los accidentes de tráfico representan una preocupación crucial para las autoridades gubernamentales a nivel mundial, ya que buscan reducir la pérdida de vidas y daños materiales. En Manizales, Colombia, los accidentes de tráfico son la tercera causa principal de muerte, después de los homicidios y suicidios. La predicción del riesgo de accidentes con alta resolución espacio-temporal es un desafío significativo, principalmente debido al complejo entorno de tráfico, el comportamiento humano y la falta de datos en tiempo real. A pesar de que las técnicas de aprendizaje automático han mostrado mejoras notables en la predicción en comparación con los modelos tradicionales, aún existen dificultades para aplicar estas técnicas en países como Colombia, donde la infraestructura y la disponibilidad de datos no siempre son óptimas.

En este estudio, se propone utilizar múltiples técnicas de aprendizaje automático para mejorar la precisión de las predicciones. Estas técnicas incluyen enfoques basados en series temporales, como Facebook Prophet, así como algoritmos más avanzados, como Light Gradient Boosting y redes neuronales recurrentes, específicamente Long Short-Term Memory (LSTM). Estas técnicas se seleccionan en base a su capacidad demostrada en el estado del arte y su capacidad para procesar datos y aprender patrones.

Para llevar a cabo esta investigación, se construyen modelos para toda la ciudad y cada zona utilizando datos de 18 años en general y de 7 años por zonas, lo que permite obtener un panorama amplio y detallado del comportamiento de los accidentes de tráfico en la ciudad. Se analiza el desempeño de cada modelo en función de su capacidad para predecir accidentes, y se selecciona el mejor modelo para cada zona. Con el fin de evaluar el rendimiento de los modelos y la precisión de las predicciones, se emplean las métricas MAPE y RMSE, que son ampliamente utilizadas en el campo de la predicción y permiten una comparación efectiva entre diferentes técnicas y enfoques.

Los resultados muestran el MAPE y el RMSE para cada zona con cada modelo y en diferentes horizontes temporales, incluyendo predicciones diarias, semanales, entre semana, fines de semana, quincenales y mensuales.

Finalmente, tras la aplicación de los algoritmos en distintas zonas y horizontes temporales de predicción, se observa que no existe un algoritmo claramente superior a los demás. Por lo tanto, se concluye que, al realizar las predicciones, se puede utilizar el algoritmo que presente el mejor desempeño según la zona y el horizonte temporal en cuestión.

Palabras clave: Accidentes de tránsito, tráfico, modelo predictivo, machine learning, espacio-temporal, Prophet, LSTM, LGB

ABSTRACT

Traffic accidents represent a critical concern for governmental authorities worldwide as they strive to reduce loss of life and material damages. In Manizales, Colombia, traffic accidents are the third leading cause of death, following homicides and suicides. Predicting accident risk with high spatiotemporal resolution poses a significant challenge, primarily due to the complex traffic environment, human behavior, and lack of real-time data. Although machine-learning techniques have demonstrated notable improvements in prediction compared to traditional models, there are still difficulties in applying these techniques in countries like Colombia, where infrastructure and data availability are not always optimal.

This study proposes multiple machine learning techniques for predicting traffic accidents in Manizales, Colombia, ranging from time-series-based approaches like Facebook Prophet to more advanced algorithms such as Light Gradient Boosting and recurrent neural networks, specifically Long Short Term Memory (LSTM). These methods seek to leverage data processing capabilities and learning patterns to enhance prediction accuracy.

To conduct this research, models are constructed for the entire city and each zone using 18 years of general data and 7 years of data per zone, providing a comprehensive and detailed view of traffic accident behavior in the city. Each model's performance is analyzed based on its ability to predict accidents, and the best model for each zone is selected. To evaluate the models' performance and the accuracy of the predictions, the MAPE and RMSE metrics are employed, which are widely used in the prediction field and allow for effective comparisons between different techniques and approaches.

The results display the MAPE and RMSE for each zone with each model and across different temporal horizons, including daily, weekly, weekdays, weekends, fortnightly, and monthly predictions.

Finally, conclusions and recommendations for taking action in future work are presented, which could lead to additional aspects for predicting traffic accidents and ultimately save lives and reduce material damages.

Keywords: Traffic accidents, traffic, predictive model, machine learning, spatiotemporal, Prophet, LSTM, LGB

CONTENIDO

Contenido

Lista de figuras

Lista de tablas

1. INTRODUCCIÓN

Con el rápido desarrollo de la urbanización, el auge del número de vehículos ha dado lugar a graves accidentes de tráfico, que han provocado víctimas y enormes pérdidas económicas. La capacidad de predecir el riesgo de accidente de tránsito es importante para la prevención de la ocurrencia de accidentes y para reducir los daños causados por los accidentes de una manera proactiva (Ren, Song, Wang, Hu, & Lei, 2018).

En la sociedad moderna, con el rápido desarrollo de las ciudades se ha generado el auge de vehículos de todo tipo, lo que ha traído una serie de problemas asociados a este medio de transporte como la congestión del tráfico, la contaminación atmosférica, los accidentes de tránsito entre otros. Los accidentes de tránsito han sido uno de los problemas de seguridad pública más importantes. Según la Organización Mundial de la Salud WHO (2018), cada año mueren alrededor de 1.3 millones de personas en accidentes de tránsito. Las afectaciones van mucho allá de sólo afectaciones en salud, como pérdidas económicas lo que se puede traducir en daños a la propiedad, gastos médicos, pérdida de ingresos debido a incapacidad laboral y gastos de rehabilitación, sumado a los costos legales y judiciales, adicionalmente se puede generar impacto emocional y psicológico significativo afectando la calidad de vida, también se puede generar una sobrecarga en los servicios de emergencia y se estima que las pérdidas asociadas a los accidentes de tránsito en todo el mundo ascienden al 3% del PIB global.

Por este motivo, el interés científico por los accidentes de tránsito ha aumentado en los últimos años, y proponer soluciones que permitan entender y reducir los accidentes se ha convertido en una cuestión crucial para el mejoramiento del transporte y la seguridad pública (Alver, Onelcin, Cicekli, & Abdel-Aty, 2021)

Con la ayuda de los grandes datos de tráfico durante las últimas décadas, se han dedicado considerables esfuerzos a analizar los datos de accidentes a distintos niveles de análisis, como la identificación de variables asociadas a los accidentes, el nivel de severidad de las lesiones o la predicción de zonas de riesgo, esta última se realiza con predicción en unidades basadas en cuadrículas (Ren et al., 2018). Con la ayuda de los grandes datos de tráfico y el aprendizaje profundo, la predicción del flujo de tráfico en tiempo real ha permitido a las personas evitar los atascos eligiendo rutas menos congestionadas. Los big data de tráfico y el aprendizaje profundo también pueden proporcionar una solución prometedora para predecir o reducir el riesgo de accidentes de tránsito (Liu, Wu, Wang, & He, 2018).

El desarrollo de las tecnologías de la información ha facilitado la recolección, digitalización y transmisión de la información. Tecnologías emergentes como internet de las cosas han permitido que organizaciones de diferentes sectores recolecten grandes volúmenes de información en tiempo real. En consecuencia, este océano de datos que se extiende cada día ha dado origen al análisis de grandes volúmenes de datos, donde por medio de las tecnologías emergentes (Big Data) es posible procesar la información y aplicar técnicas de análisis y modelamiento de datos de forma que se identifiquen patrones y asociaciones que faciliten comprender y predecir diferentes fenómenos en el contexto de la información.

A partir de las estadísticas históricas de accidentes de tránsito recopiladas a lo largo de los años, se puede obtener una visión general de las regiones de riesgo. Sin embargo, la distribución del riesgo varía mucho según la hora del día, el día de la semana y el mes del año, y hay factores complejos que pueden afectar al riesgo de accidente, como la densidad de población, el flujo de tráfico, el tiempo, los acontecimientos, etc. Además, las estadísticas históricas no permiten predecir el riesgo de accidente de forma precisa y dinámica (Caparrós, 1999). Por lo tanto, para pronosticar el cambio dinámico del riesgo de accidente de forma cuantitativa y detallada, se puede utilizar el método de aprendizaje profundo en la predicción del riesgo de accidente de tránsito, numerosos trabajos se han realizado en este sentido, no obstante, una de las limitaciones de estos trabajos es que no tuvieron en cuenta los patrones temporales de los accidentes de tráfico en su modelo. Sin esta información, el poder predictivo del modelo podría verse debilitado (Ren et al., 2018).

De acuerdo a lo anterior este trabajo tiene como objetivo construir un modelo predictivo basado en tecnologías de machine learning (ML) que utilice los datos espacio-temporales de los accidentes de tránsito y otras variables relacionadas, para la predicción de accidentes en la ciudad de Manizales, el cual estime la probabilidad de ocurrencia de accidentes para cada zona geográfica establecida en el trabajo, de forma que anticipe estos sucesos y se puedan tomar acciones para su prevención en un espacio temporal definido.

La investigación que se plantea en este documento corresponde a la Tesis de Maestría en Ingeniería Computacional de la Universidad de Caldas y está vinculada al Grupo de Investigación Piensa, Línea de investigación Inteligencia Artificial y Big Data, gestión y analítica de datos e inteligencia de negocios, contribuyendo a su objetivo de apropiar y desarrollar tecnología, propiciando ambientes que generen proyectos de Investigación, Desarrollo e Innovación y (I+D+I) que aporten al país y a la comunidad científica.

Este proyecto de investigación cuenta con el apoyo de la Universidad de Caldas, específicamente por medio de su Facultad de Ingenierías y el programa de Maestría en Ingeniería Computacional, otorgando conocimiento al candidato a magister en áreas relacionadas al tema de esta tesis; y a su vez es financiado de manera no monetaria por la secretaría de movilidad de la Alcaldía de Manizales, proporcionando información, su know-how y propiciando espacios para que el desarrollo de la investigación planteada pueda llevarse a cabo.

Esta investigación recibió apoyo del Fondo de Becas de Investigación Manizales + Innovadora, creado por la Alcaldía de Manizales y Manizales Campus Universitario. Las opiniones, tesis y argumentos expresados son de propiedad exclusiva del autor y no representan el punto de vista de la Alcaldía de Manizales ni de ninguna de las instituciones que hacen parte del programa Manizales Campus Universitario.

1.1 Planteamiento del problema

Aproximadamente 1,3 millones de personas mueren en las carreteras del mundo y entre 30 y 50 millones resultan heridas cada año. Los accidentes de tráfico son una de las principales causas de muerte en todos los grupos de edad y la principal causa de muerte de

niños y adultos jóvenes de entre 5 y 29 años. El riesgo de morir en un accidente de tránsito es más de 3 veces mayor en los países de ingresos bajos que en los países de ingresos altos (WHO, 2018).

Uno de los problemas que se está presentando en los países desarrollados y en desarrollo es el crecimiento acelerado de la tasa de accidentes de tráfico y están invirtiendo en esfuerzos para intentar reducir la frecuencia y gravedad de éstos (Tan, 2018). Los sistemas de transporte han estado estrechamente relacionados a las transformaciones históricas en el comercio, la inversión y la movilidad del capital humano a escala mundial (Díaz Fuentes, 2014), una parte importante del flujo de actividad económica mundial está directamente relacionada con los sistemas de transporte, sin embargo, según Díaz Fuentes (2014) junto a su impacto económico y social positivo sobre la economía y la sociedad, los sistemas de transporte y comunicaciones, también tienen efectos y externalidades negativas, tales como la congestión, la polución o los accidentes de tráfico.

Para entender la historia de los sistemas de movilidad actuales, debemos situarnos en la segunda mitad del siglo XIX donde posteriormente al estallido de la revolución industrial tuvo un auge del desarrollo de la industria a nivel mundial, causando así desplazamiento de las personas hacia las urbes, dando paso a las grandes ciudades que conocemos hoy en día. El modelo general de desarrollo urbano está dado por la configuración de las ciudades en los lugares donde las personas realizan sus actividades cotidianas, algunas de las cuales se desarrollan afuera de sus viviendas y para las cuales se requiere el uso de diversos medios de transporte tales como automóviles, autobuses, motocicletas, ferrocarriles entre otros (Quintero-González, 2017). Los medios de transporte más utilizados son, automóviles, autobuses y motocicletas, dado que constituyen una parte fundamental de la movilidad en las ciudades. Varios estudios recientes han dicho que los problemas de movilidad urbana están directamente relacionados con los rápidos avances que ha tenido la urbanización (Quintero-González, 2017), y teniendo en cuenta que actualmente el nivel de uso de estos medios de transporte es cada vez mayor, así mismo se ven incrementados los niveles de congestión polución y accidentes de tráfico.

Sin embargo, el uso constante de los medios de transporte conlleva a una serie de riesgos, según la Organización Mundial de la Salud WHO (2021), los traumatismos causados por el tránsito causan pérdidas económicas considerables a las personas, sus familias y las naciones en su conjunto. Estas surgen del costo de tratamientos médicos, así como de la pérdida de productividad de las personas muertas o discapacitadas por sus lesiones, y de los miembros de la familia que necesitan ausentarse del trabajo o de la escuela para atender a los heridos. Los accidentes de tráfico cuestan a la mayoría de los países el 3% de su producto interno bruto.

Los actores más afectados por los accidentes de tránsito son los peatones, ciclistas y motociclistas (WHO, 2021). A raíz de esta problemática, y que la inmensa mayoría de las muertes y lesiones graves que se producen por siniestros viales son prevenibles, la ONU declaró el periodo 2021-2030 como el Segundo Decenio de Acción por la Seguridad Vial y plantea como objetivo reducir las muertes y lesiones por accidentes de tráfico en al menos un 50% en ese período. Los objetivos del Decenio 2011-2020 no se cumplieron (Portafolio, 2020).

Colombia no es ajena a esta problemática, en donde la seguridad vial constituye un problema grave de salud pública y constituye la segunda causa de muerte después de los homicidios. Según informes del Instituto Nacional de Medicina Legal y Ciencias Forenses INMLCF (2019), entre enero y diciembre de 2019, 6.892 personas perdieron la vida y más de 35.000 sufrieron lesiones graves en accidentes de tránsito. La tercera parte de estas muertes estuvo relacionada con incidentes de vehículos livianos. Según cifras de la Agencia Nacional de Seguridad Vial, entre enero y diciembre de 2020, 5.641 personas perdieron la vida en siniestros viales, de éstos 3.140 eran motociclistas (56%), 1.229 peatones (22%), 456 ciclistas (8%) ente otros.

En el caso específico de la ciudad de Manizales, Caldas, Colombia, según cifras de la secretaría de movilidad de la alcaldía de Manizales en el año 2019 se presentaron 2.708 accidentes, los cuales dejaron como resultado 39 personas fallecidas, 15 peatones, 15 motociclistas, 3 usuarios de vehículos, 2 de bicicleta, 1 usuario de transporte público y 3 usuarios de transporte de carga, sumado a lo anterior se dejaron cientos de personas heridas y múltiples daños materiales, todo esto sin contar con las afectaciones generadas a hospitales, impacto en la movilidad de la ciudad por cierres parciales o totales de vías y gastos médicos o funerarios. Para el año 2022 ocurrieron más de 850 accidentes de tránsito que incluyeron a personas lesionadas, con un saldo trágico de 45 siniestros y 50 fallecimientos y entre enero a abril del 2023 ya se han presentado más de 247 accidentes de tránsito, con un saldo de 214 lesionados y 13 víctimas fatales.

Adicionalmente, de acuerdo con las bases del PND 2018-2022, el impacto que esta problemática tiene sobre la economía colombiana asciende a 23,9 billones de pesos anuales, lo que equivale a 3,6% del Producto Interno Bruto.

La ONU instituyó el tercer domingo de noviembre de cada año, en recuerdo de las personas que mueren o sufren lesiones graves por esta causa, con el fin de visibilizar el impacto social y económico que producen los incidentes de tránsito y reconocer el sufrimiento de las víctimas.

La ONU recomienda trabajar desde diferentes frentes bajo el enfoque de un sistema seguro: fortalecer instituciones y datos, mejorar la infraestructura, controlar comportamientos riesgosos (como el exceso de velocidad), atender a las víctimas y contar con vehículos más seguros que ayuden a prevenir incidentes viales y que protejan a pasajeros y usuarios de las vías en caso de choque.

Según la revista Semana (2021), para el presente año Colombia tiene un parque automotor con un total de 16.042.336 vehículos. La radiografía presentada por el Registro Único Nacional de Tránsito (RUNT), sobre el parque automotor en Colombia dejó ver que de estos vehículos registrados, el 59 % son motocicletas, es decir, 9.419.374, junto a esto, se reportó que vehículos como automóviles, camiones, camionetas, buses, busetas y ciclomotores representan el 40 % de este parque (6.453.355) y la maquinaria como remolques y semirremolques tienen un 1 % del mercado (169.607).

Por tipo de servicios, los vehículos particulares son los de mayor relevancia en Colombia, acogiendo el 92 % de todo el parque automotor, seguidos del servicio público que tienen una representación del 6 % y otras clases de servicios el 2 %.

Todo este parque automotor ha tenido una tasa de crecimiento superior al de la capacidad de las carreteras del país, lo cual incrementa las posibilidades de que se presenten accidentes de tráfico.

Las cifras mencionadas anteriormente, evidencian, por una parte, el dramático problema de salud pública que representa la accidentalidad vial y por otra, la urgencia en la ejecución de medidas que frenen las tendencias actuales y que mitiguen este fenómeno que se ha convertido en un factor de fuerte impacto no sólo en términos humanos sino también económicos en todo el mundo y en el país.

Según la OMS la mayoría de estos accidentes sean fatales o no, son evitables, y es necesario avanzar hacia la creación de factores que favorezcan la disminución de accidentes.

Específicamente en caso de Colombia motociclistas y peatones ponen casi el 80% de los muertos en las vías (Tiempo, 2021). En la última década por cada carro que entró en circulación ingresaron 2 motos, y la tasa de años de vida perdidos por muertes prematuras en accidentes de tránsito es de 1,3 años por cada 1.000 mujeres, mientras que para los hombres es de 6,8. Si bien durante el año 2020 y 2021, se presentó una disminución en los accidentes viales y la cantidad de personas afectadas, es de asociar este comportamiento a razones derivadas del aislamiento por el covid-19 y no por una tendencia real en condiciones no pandémicas, es decir que la siniestralidad vial no ha mejorado a pesar de las campañas y los esfuerzos de las autoridades nacionales y regionales.

La tasa de víctimas mortales no ha logrado disminuirse significativamente, sin tener en cuenta el año 2020, dado que en el año 2000 la tasa nacional de muertos en accidentes de tránsito era de 15.5 por cada 100.000 habitantes (Dominguez Márquez, López Castro, & González, 2004), y en el año 2019 de 14,2 por cada 100.000 habitantes, si tenemos en cuenta que en un periodo de 19 años solo hubo una pequeña disminución que hace que aun falte mucho camino por recorrer, debido a que en cambio, en los 35 países miembros de la Organización para la Cooperación y el Desarrollo Económicos (OCDE), para el año 2018 la tasa media de mortalidad por dichas causas fue de 8 por cada 100.000 habitantes (BM, 2018).

Según el Banco Mundial BM (2018), se deben hacer mejoras en diferentes ámbitos para mejorar la seguridad vial, para conseguir estas mejoras, en el informe se enumeran intervenciones como reducir y hacer cumplir los límites de velocidad, disminuir la conducción bajo los efectos del alcohol, incrementar el uso del cinturón de seguridad mediante la exigencia del cumplimiento de esta obligación y a través de campañas de sensibilización del público, e incorporar la seguridad vial en todas las etapas de planificación, diseño y operación de la infraestructura vial.

Teniendo en cuenta que los accidentes de tránsito, constituyen un problema para el desarrollo de salud pública en el mundo, su predicción ha sido materia de investigación en diversos países en todo el mundo, no obstante, la gran mayoría de éstos estudios no son aplicados dentro del contexto colombiano y para poder aplicarse en algunos casos deben ser ajustados. En la literatura actual los modelos son formulados considerando varias variables propias del entorno local, de los datos disponibles y de otros factores como el clima, causando que el modelo creado sea específico para una región.

Por otra parte, las investigaciones realizadas orientadas a la predicción de los accidentes de tránsito, están orientadas a la asociación de otras variables diferentes a las de tipo espacio-temporal, en los trabajos en los cuales se encuentra asociada la ubicación geográfica, no son asociadas las variables temporales, lo cual afecta significativamente el desempeño de los modelos, puesto que los accidentes de tráfico están asociados al flujo vehicular, lo cual es variable en función de la hora, día, mes, y año.

Actualmente no se encuentran herramientas computacionales que implementen modelos predictivos de accidentes de tránsito espacio-temporales basados en series de tiempo para la ciudad de Manizales, Colombia, como medida para afrontar este problema. Adicionalmente de los trabajos realizados hasta el momento en el país se utilizan otras técnicas diferentes a las propuestas en este estudio.

1.2 Justificación

Se requiere una herramienta tecnológica de análisis predictivo, que permita a las ciudades estimar la cantidad de accidentes en sus carreteras, y que sea usada para tomar decisiones sobre las acciones de planificación y prevención de accidentes de tránsito necesarias, que conlleven a una reducción de los índices de accidentalidad y todos los índices relacionados, impactando así positivamente la prevención de eventos negativos los cuales pueden generar pérdidas humanas y/o materiales. En vista de que no existe una sola metodología, y que se deben explorar diferentes algoritmos para escoger el que genere el mejor resultado de predicción, la aplicación de técnicas de validación como cross-validation permiten conocer que tan exactas son las predicciones de cada modelo, permitiendo identificar el mejor.

Ofrecer datos actualizados sobre zonas con más alta probabilidad de accidentes y ubicarlos geográficamente permite hacer frente a los altos índices de mortalidad en la ciudad de Manizales, en donde en el año anterior fallecieron 42 personas, teniendo problemáticas críticas como el caso de peatones afectados y motociclistas, tener datos disponibles facilita la toma de decisiones que generen disminuciones en la cantidad de accidentes sin lesionados, pero que dejan grandes pérdidas económicas ya que las personas deben asumir los costos de reparaciones por daños a sus bienes o a terceros. La estimación de zonas de riesgo acertada tanto geográfica como temporalmente, permite a las autoridades gubernamentales la toma de decisiones necesarias para la prevención de accidentes, lo cual impacta positivamente los indicadores de mortalidad, lesionados y daño a bienes, también tienen un impacto positivo en la movilidad de las ciudades, mediante la no ocurrencia de accidentes que generan grandes congestiones. Por otra parte, aumentar la seguridad vial permite mejorar la calidad de vida de los habitantes de las ciudades y permite aumentar la expectativa de vida al evitar muertes prematuras. Otro beneficio es la optimización de recursos destinados para la prevención de accidentes en conjunto con recursos destinados para el mejoramiento de la infraestructura vial, puesto que se pueden priorizar las zonas con mayor impacto en la accidentalidad para ejecutar estas obras, evitando que los recursos se desperdicien siendo utilizados en obras de bajo impacto, agravando la condición de accidentalidad en puntos críticos por demoras adicionales.

Ofrecer soluciones que permitan disminuir los accidentes, permite prevenir gran cantidad de lesiones en las personas involucradas, y las afectaciones a las que conllevan estas lesiones, como el pago de terapias, pérdida de la capacidad laboral, e inclusive lesiones con secuelas de por vida.

Más del 90% de las defunciones debidas a colisiones causadas por el tránsito se registran en los países de ingresos bajos y medianos, y las tasas más elevadas se registran en África. Incluso en los países de ingresos altos, las personas de nivel socioeconómico más bajo corren más riesgo de verse involucradas en estas colisiones (WHO, 2021). Teniendo en cuenta que Colombia pertenece al grupo de países con un ingreso mediano (BM, 2021), hace necesario que se apliquen en Colombia modelos computacionales que permitan prevenir accidentes.

Se requiere una herramienta que permita predecir la cantidad de accidentes de tránsito espacio-temporales en el corto y mediano plazo, en la ciudad de Manizales, Colombia, teniendo en cuenta el contexto de la región. Adicionalmente es necesario emplear métodos de validación del modelo de predicción que garanticen que las estimaciones sean exactas.

1.3 Objetivos

1.3.1 Objetivo general

Generar un modelo predictivo de las zonas de riesgo espacio-temporales de accidentes de tráfico en la ciudad de Manizales utilizando técnicas de aprendizaje automático

1.3.2 Objetivos específicos

- Analizar y organizar el conjunto de datos para aplicar minería de datos en series temporales, basado en la información histórica de accidentalidad de la ciudad de Manizales.
- Construir el modelo predictivo de las zonas de riesgo de accidentes de tráfico utilizando técnicas de aprendizaje automático para series temporales.
- Probar el modelo utilizando las técnicas de validación para modelos predictivos basados en series de tiempo.

2. MARCO DE REFERENCIA

El término big data es asociado con el término macrodatos, a priori este término nos lleva a pensar que el requisito para considerar un conjunto de datos como big data se refiere únicamente a su tamaño, pero otros autores han sugerido diferentes dimensiones de este

concepto, por ejemplo Laney (2001) sugirió que Volumen, Variedad, y Velocidad (las tres V) son las tres dimensiones de desafíos en la gestión de datos. Otros autores como, Por ejemplo, Gartner, Inc. define big data en términos similares:

> *"Los macrodatos son activos de información de gran volumen, alta velocidad y / o gran variedad que exigen formas rentables e innovadoras de procesamiento de información que permitan una mejor comprensión, toma de decisiones y automatización de procesos.". (Gartner, 2012)*

Del mismo modo, otros autores definen big data de la siguiente manera:

> *"Big Data es un término que describe grandes volúmenes de datos de alta velocidad, complejos y variables, que requieren técnicas y tecnologías avanzadas que permitan la captura, almacenamiento, distribución, gestión y análisis de la información." (Mills et al., 2012)*

Definición de las tres V.

El volumen se refiere a la magnitud de los datos y estos se pueden expresar en medidas de terabytes o petabytes. El volumen de big data puede ser relativa y variable dependiendo del tipo de datos que se almacenan, adicionalmente si se definiera una escala actual como medida estándar, en futuro esta escala podría no ser útil debido a la alta velocidad de crecimiento de la capacidad de procesamiento de datos de los hardware futuros, y también debido a que las técnicas de análisis de información utilizadas para una base de datos tabular y una de almacenamiento de imágenes o sonidos son diferentes y tienen capacidades diferentes.

La variedad se refiere a la heterogeneidad estructural en un conjunto de datos. Los avances tecnológicos permiten a las empresas utilizar varios tipos de datos estructurados, semiestructurados y no estructurados. Datos estructurados, que constituyen solo el 5% de todos los datos existentes (Cukier, 2010), los datos de bases datos relacionales componen este porcentaje. Los otros tipos de datos como imágenes, textos, videos, audios son datos no estructurados por lo tanto el análisis sobre este tipo de datos requiere de técnicas diferentes a las tradicionales.

En el mundo se recogen datos no estructurados desde hace varios años, sin embargo, con el surgimiento de las nuevas tecnologías se ha permitido usar estos datos como fuente de aprovechamiento para procesos comerciales y todo tipo de procesos, algunos ejemplos de esto son las tecnologías de reconocimiento facial, patrones de movimiento en ciudades por GPS, seguimiento de clics en páginas web, los cuales proporcionan diariamente cantidades enormes de información.

La velocidad se refiere a dos ámbitos, velocidad de generación de los datos y velocidad de análisis y captura. Sistemas informáticos alimentados por las redes sociales y por teléfonos celulares computadores, en donde cada habitante de una ciudad diariamente está generando una cantidad ingente de datos cada día, abre la necesidad de crear soluciones que permitan analizar datos en tiempo real, para que esta información sea aprovecha en el momento oportuno.

Además de las tres V también se han mencionado otras características de la definición de Big Data, algunas son:

- Veracidad: Representa la fiabilidad de algunas fuentes de datos, ya que por ejemplo en las redes sociales se capturan datos que son producto de expresiones y sentimientos humanos, lo cual puede llegar a ser altamente subjetivo, sin embargo, es importante analizar esta información.
- Variabilidad y complejidad: La primera se refiere a la tasa cambiante de flujo de datos, ya que no es igual en el tiempo, y la segunda se refiere al hecho de que los datos provienen de diferentes fuentes y es necesario conectar, combinar, limpiar y transformar los datos recibidos de diferentes fuentes.
- Valor: Según la definición de Oracle los datos se caracterizan por una densidad de valor relativamente baja, sin embargo, se puede obtener un alto valor, analizando grandes volúmenes de datos.

Big data se refiere a conjuntos de datos o combinaciones de varios conjuntos de datos, en los cuales su tamaño, complejidad y variabilidad es muy alta, así mismo se entiende que la velocidad de crecimiento de las bases de datos es muy alta, lo cual dificulta la manipulación de estos utilizando elementos tradicionales de manejo de datos. Según Pérez (2015) El término big data suele aplicarse a conjuntos de datos que superan la capacidad de software habitual y no existe un consenso sobre su tamaño, este término suele referirse a los siguientes tipos de datos:

- Datos de la empresa tradicional
- Machine-generated / sensor data
- Datos de medios sociales
- Grandes bases de datos
- Grandes conjuntos de datos

Y a su vez estos tipos de datos pueden ampliarse un poco más:

- Web and social media: Contenido de información de páginas web y redes sociales
- Machine-to-machine (M2M): Información proveniente de sensores o medidores que capturan datos automáticamente
- Big transacción data: Contiene registros de transacciones en todo tipo de telecomunicaciones, son estructurados y no estructurados
- Biometrics: Información biométrica como huellas digitales, reconocimiento facial, genes, retina etc.
- Human generated: Contiene diferentes registros de información generada por las actividades humanas, en sus diferentes actividades.

2.1 Análisis de big data

En el punto anterior se habló sobre las condiciones que se tienen en cuenta para considerar un conjunto de datos como Big Data, sin embargo, para extraer el conocimiento

inmerso en los conjuntos de datos debemos ejecutar el análisis de Big Data, el cual permitirá traducir los datos en información tangible para la toma de decisiones. En el sector de las tecnologías de la información, Big Data hace referencia a la manipulación de grandes volúmenes de datos, lo que abre la puerta a una dificultad para realizar el tratamiento de estos en todas sus etapas, captura, almacenamiento, búsqueda, modelamiento, análisis y visualización.

Por otra parte Gandomi and Haider (2015), identifican los pasos de la administración de los datos y el análisis:

Administración de datos

1. Adquisición y almacenamiento.
2. Extracción, limpieza y registro de metadatos.
3. Integración, agregación y representación.

Análisis de datos

1. Modelado y análisis
2. Interpretación

La primera etapa correspondiente a la administración de los datos refiere la parte previa al análisis, esta fase debe garantizar que los grandes volúmenes de datos que están en diferentes fuentes, con diferentes estructuras, sean depurados y estén seguros de una manera eficiente en cuanto a costos (M. Chen, Mao, & Liu, 2014). Las tecnologías de big data ayudan a gestionar los datos durante la administración, facilitando la captura y el almacenamiento de datos en tiempos inferiores a los de otras tecnologias, minimizando tiempos de costos de procesamiento y verificando el valor de los datos antes de invertir recursos (Oussous, Benjelloun, Ait, & Belfkih, 2017).

Dentro de la administración de datos encontramos el primer paso, adquisición y almacenamiento. El almacenamiento de datos busca retener y administrar los datos de una forma escalable, como una plataforma que permita cumplir con los requerimientos de aplicaciones que leen y analizan datos constantemente. Un almacenamiento de big data ideal debe permitir almacenar cantidades de datos virtualmente ilimitadas, a altas tasas de escritura y lectura, utilizando modelos de datos flexibles y eficientes que soporten tanto datos estructurados como no estructurados y que opere con datos encriptados para garantizar la seguridad (Strohbach M., 2016).

Una vez se tengan los datos correctamente almacenados, se procede al segundo paso, la extracción y limpieza. Grandes volúmenes de datos son generados diariamente por sensores y dispositivos en todo el mundo. Los datos pueden contener valores atípicos, generadores de ruido, datos erróneos o incompletos, el reto es como limpiar, organizar y filtrar la información confiable para el análisis (M. Chen et al., 2014). Según Brownlee (2020) la preparación de datos es importante debido a que para poder extraer información de los datos, éstos deben ser analizados por diferentes programas informáticos que utilizan algoritmos, la mayoría de éstos algoritmos funciona con datos estructurados o datos

tabulados, estos datos los podemos observar mediante hojas de cálculo o matrices, dependiendo de los parámetros de entrada del programa o del algoritmo, se deben preparar los datos para que puedan ser leídos, de lo contrario el análisis no se realizará (L. Chen, Jia, Sellis, & Liu, 2014).

El análisis de big data es la sub-área de big data que se ocupa de añadir estructura a los datos para apoyar la toma de decisiones, así como de apoyar los escenarios de uso específicos del dominio (Cavanillas, Curry, & Wahlster, 2016). En análisis es la fase más importante en la cadena del valor de big data (M. Chen et al., 2014). Una vez que los datos sean estructurados o no estructurados y los cuales pueden ser recolectados en formatos diferentes, se han transformado y almacenado correctamente, se procede a hacer uso para su análisis. Generalmente el análisis de datos se clasifica de acuerdo al ámbito de la aplicación, descriptivo, predictivo o prescriptivo. Basado en requerimientos de tiempos de respuesta y volumen de datos L. Chen et al. (2014) clasifica el análisis de datos en dos dimensiones, tiempos de respuesta y nivel de almacenamiento.
Tiempos de respuesta:

- Análisis de datos en tiempo real: Orientado a sectores que tienen la necesidad inherente de evaluar los datos en tiempo real, como las industrias, comercio electrónico, salud, operación aeronáutica etc.
- Análisis de datos offline: Usado donde no se requiere una respuesta inmediata, se utiliza machine learning u otros análisis estadísticos.

Nivel de almacenamiento:

- Análisis a nivel de memoria: Cuando el flujo total de los datos no supera el máximo de memoria RAM del servidor, permite análisis en tiempo real.
- Análisis para la inteligencia de negocios (BI): Generalmente los datos superan el tamaño total de RAM del servidor, adicionalmente los datos son llevados en ambientes de análisis de datos como por ejemplo datos estructurados para cubos OLAP.
- Análisis masivo: Los datos superan la capacidad de la memoria RAM y supera las capacidades de los ambientes de análisis de datos, requiere análisis más avanzados y no en tiempo real.

2.2 Análisis predictivo

El análisis predictivo es un área de la estadística que se ocupa de extraer información de los datos y la utiliza para predecir tendencias y patrones de comportamiento (Ongsulee, Chotchaung, Bamrungsi, & Rodcheewit, 2018). Generalmente los análisis predictivos son usados para hacer predicciones futuras, sin embargo, se podrían utilizar para eventos que han ocurrido en el pasado, presente o futuro, por ejemplo, la predicción de sospechosos de un crimen que ya ocurrió. El núcleo del análisis predictivo se basa en capturar las relaciones entre las variables explicativas y las variables predichas a

partir de sucesos pasados y explotarlas para predecir el resultado desconocido (Ongsulee et al., 2018).

El análisis predictivo evalúa valores futuros a través de modelos de predicción los cuales se construyen a partir de una variedad técnicas y algoritmos que son utilizados para la caracterización de la información histórica, lo cual puede ser usado para predecir futuros eventos u ocurrencias (Pusala, Salehi, Katukuri, Xie, & Raghavan, 2016). Por ejemplo, las industrias utilizan el análisis predictivo para predecir los fallos de las máquinas utilizando los datos de los sensores (IBM, 2020). Aerolíneas, universidades aplicaciones de comercio electrónico utilizan diariamente el análisis predictivo como un factor primario para su operación diaria. La Universidad de Purdue utiliza el análisis de big data para prevenir problemas de comportamiento académico (Rijmenana, 2013).

El análisis predictivo abarca una variedad de técnicas estadísticas, técnicas de modelado predictivo, aprendizaje automático y de datos que analizan hechos actuales e históricos para hacer predicciones sobre acontecimientos futuros o desconocidos (Nyce, 2007).

Proceso de análisis predictivo:

1. Definir el proyecto: Definir los objetivos del proyecto, el resultado, el alcance del esfuerzo, los objetivos empresariales, identificar los conjuntos de datos.
2. Recogida de datos: La minería de datos para el análisis predictivo prepara los datos de múltiples fuentes para análisis.
3. Análisis de datos: El análisis de datos es el proceso de inspeccionar, limpiar y modelar los datos con el objetivo de descubrir información útil, para llegar a una conclusión.
4. Estadística: El análisis estadístico permite validar los supuestos, las hipótesis y ponerlas a prueba utilizando modelos estadísticos.
5. Modelización: La modelización predictiva proporciona la capacidad de crear automáticamente modelos predictivos precisos sobre el futuro.
6. Despliegue: El despliegue del modelo predictivo proporciona la opción de desplegar los resultados analíticos en proceso diario de toma de decisiones para obtener resultados informes y resultados mediante la automatización de las decisiones en la modelización.
7. Monitorización del modelo: Los modelos se gestionan y para revisar el rendimiento del modelo y asegurarse de que para asegurarse de que proporciona los resultados esperados.

De acuerdo a los pasos anteriores el análisis predictivo se genera en los pasos 4 y 5, las técnicas utilizadas pueden categorizarse de dos tipos, estadísticas y minería de datos, las técnicas estadísticas son utilizadas en donde los conjuntos de datos tienen relaciones más simples y el número de variables a analizarse es relativamente pequeño, para el caso de la minería de datos es aplicada a conjuntos de datos que son masivos y tienen relaciones de carácter complejo y heterogéneo. En la minería de datos encontramos las siguientes técnicas: *Arboles de decisión:* Se realizan divisiones en subconjuntos de datos por niveles

de acuerdo a la influencia de un dato respecto a los demás, construyendo un árbol que luego servirá para determinar una variable a predecir. *Clasificador Bayesiano*: Construye un modelo probabilístico a partir de la probabilidad de cada variable objetivo dadas las variables de entrada conocidas. *Reglas de inducción*: Busca encontrar una serie de reglas para relacionar variables entre sí, y predecir una a partir de la otra. Las reglas permiten expresar disyunciones de manera más fácil que los árboles de decisión y tienden a preferirse con respecto a éstos por tender a representar "pedazos" de conocimiento relativamente independientes (Robles Aranda, Trujillo Rasúa, & Sotolongo León, 2013). *K-vecinos más cercanos*: Se almacenan los ejemplos de entrenamiento de datos históricos y cuando se requiere clasificar a un nuevo objeto, se extraen los objetos más parecidos y se usa su clasificación para clasificar al nuevo objeto. Los vecinos más cercanos a una instancia se obtienen, para el caso de los atributos continuos, utilizando la distancia Euclidiana sobre los n posibles atributos (Orea, Vargas, & Alonso, 2005). *Máquinas de soporte vectorial*: Una maquina aprende la superficie de decisión de dos clases distintas de los puntos de entrada. Como un clasificador de una sola clase, la descripción dada por los datos de los vectores de soporte es capaz de formar una frontera de decisión alrededor del dominio de los datos de aprendizaje con muy poco o ningún conocimiento de los datos fuera de esta frontera (Betancourt, 2005). *Facebook Prophet*: Este algoritmo de pronóstico de series temporales fue desarrollado por Facebook y se basa en un modelo aditivo en el que se combinan componentes de tendencia, estacionalidad y feriados. La técnica es escalable y flexible, lo que permite acomodar diferentes patrones en los datos de series temporales y es capaz de lidiar con la presencia de valores atípicos y faltantes. Prophet se ha utilizado ampliamente en diversos dominios, incluyendo pronósticos de demanda, ventas y recursos energéticos, siendo efectivo tanto en predicciones a corto como a largo plazo (Taylor, 2018). *SARIMAX*: El modelo SARIMAX (Seasonal AutoRegressive Integrated Moving Average with eXogenous regressors) es una extensión del modelo ARIMA que incorpora factores estacionales y variables exógenas. Este algoritmo es adecuado para series temporales con componentes estacionales y no estacionales y utiliza la autorregresión, la integración y la media móvil para ajustar y predecir los datos. Además, el modelo SARIMAX considera variables externas que pueden afectar la serie temporal, mejorando la precisión de las predicciones. SARIMAX ha sido aplicado en diversos campos, como pronósticos económicos, de tráfico y meteorológicos (Box, 2013). *Light Gradient Boosting (LGB)*: Este algoritmo es una implementación eficiente del método de Gradient Boosting, que combina árboles de decisión débiles en un modelo aditivo y optimiza iterativamente el rendimiento utilizando el gradiente descendente. Light Gradient Boosting es conocido por su rapidez y eficiencia en el manejo de conjuntos de datos grandes y de alta dimensionalidad. El algoritmo también permite la personalización de hiperparámetros para adaptarse a diferentes problemas, lo que lo convierte en una opción popular en una amplia gama de aplicaciones, como clasificación, regresión y ranking (Guolin Ke, 2017). *Redes neuronales*: Busca generar salidas que representen la variable dependiente a partir de las variables de entrada, operaciones matemáticas y el peso de acuerdo a los resultados se van ajustando para obtener una mayor exactitud, en el punto de

partida se asignan valores aleatorios, posteriormente se van ajustando los valores de acuerdo a los resultados obtenidos.

Las técnicas estadísticas son usadas en conjuntos de datos de menos tamaño, en los cuales hay relaciones bien definidas. Algunas de estas técnicas son: *Regresión lineal*: Modelo basado en la ecuación lineal para para predecir valores de la variable de salida respecto a los valores de la variable de entrada. *Regresión logística*: Modelo utilizado para predecir valores binomiales usando predictores o atributos que pueden ser de clase numérica.

2.3 Aplicación de técnicas para el análisis predictivo

El uso de las técnicas de análisis predictivo en el mundo ha tenido un auge en los últimos tiempos gracias al avance de las herramientas computacionales y al aumento de la capacidad de procesamiento de información el cual es cada vez mayor. Las organizaciones en todo el mundo han aprovechado esta nueva oportunidad de uso de la tecnología para la resolución de cualquier tipo de problemas y cada día se ven usos más diversos para la aplicación de técnicas de análisis predictivo (Integra, 2021). Las organizaciones de diferentes sectores están utilizando el análisis predictivo para reducir riesgos, optimizar operaciones, eliminar desperdicios, calcular demandas, incrementar la utilidad entre otros, compañías de todos los sectores y gobiernos, así como administraciones de ciudades están haciendo lo propio en búsqueda de solución a sus problemas más comunes. Por ejemplo, en el sector de transporte, en los últimos años los datos de tráfico se han disparado y realmente se ha entrado en la era del big data para el transporte. Los métodos de predicción de tráfico, accidentes, riesgo de colisiones, en otros países se han aplicado metodologías de análisis predictivo para estimar la probabilidad de ocurrencia de accidentes de tráfico en determinadas áreas de las ciudades, y también se ha utilizado para anticipar muchos tipos de información en organizaciones de todo tipo en el mundo, a continuación, algunos ejemplos

Tabla 1

Aplicaciones de análisis predictivo en organizaciones

Sector	Organización	Beneficio	Referencia
Comercial	FMCG	Pronosticar la demanda de productos en el mercado FMCG. Creando modelos de regresión que permitieron predecir las ventas con precisión utilizando Azure ML Studio.	(Pavlyuchenko, Panfilov, & Gorshkov, 2021)
Comercial	Amazon	Optimización de operaciones logísticas, prediciendo el comportamiento de compra de sus clientes.	(Joel R. Spiegel, 2004)

Telecomunicaciones	Syriatel Telecom	Segmentación de los clientes de telecomunicaciones según su comportamiento y nivel de lealtad, lo que permitió ofrecer ofertas y servicios personalizados.	(Wassouf, Alkhatib, Salloum, & Balloul, 2020)
Financiero y bancario	Grupo Bancolombia	Aumento en la calidad de los reportes de transacciones sospechosas y ahorro del 80% reduciendo el personal necesario para revisar la cantidad masiva de transacciones diarias.	(IBM, 2010)
Gobierno y estado	Commonwealth Scientific and Industrial Research Organization (CSIRO)	Predicción del ingreso de pacientes a la sala de emergencias del hospital Gold Coast con una precisión del 93%.	(Howarth, 2014)
Manufacturero y cadena de suministro	Transportadores marítimos	Predicción de lugares en los que se requiere un contenedor, así como el tiempo que tomaría arribar al puerto.	(Banker, 2016)

Elaboración propia.

La predicción de accidentes de tránsito es fundamental para prevenir lesiones, muertes y daños materiales. Los análisis de gravedad de causalidad de los accidentes de tránsito son esenciales para mejorar la respuesta de rescates en los accidentes, reduciendo así las víctimas y las pérdidas de propiedad causadas por los accidentes viales (Zong, Chen, Tang, Yu, & Wu, 2019). Los accidentes de tránsito suelen provocar graves bajas humanas y enormes pérdidas económicas en escenarios del mundo real. La predicción oportuna y precisa de los accidentes de tráfico tiene un gran potencial para proteger la seguridad pública y reducir las pérdidas económicas (Yu et al., 2021). La creciente cantidad de datos sobre accidentes de tránsito que se está recolectando en todo el mundo, se utiliza para entrenar modelos predictivos, aunque esta es una tarea desafiante debido a la relativa rareza de los accidentes, las interdependencias de los accidentes de tráfico tanto en el tiempo como en el espacio, y la alta dependencia del comportamiento humano (de Medrano & Aznarte, 2021). Recientemente algunas técnicas de aprendizaje profundo han mostrado mejoras en cuanto a la predicción de accidentes respecto a los modelos tradicionales.

2.4 Predicción con series de tiempo

Las series de tiempo son una secuencia de observaciones, medidas en determinados momentos del tiempo, son ordenadas cronológicamente y espaciados entre sí de una manera uniforme, así los datos usualmente son dependientes entre sí, el principal objetivo de una serie de tiempo, es su análisis para hacer pronostico (Villavicencio, 2010). Las series temporales pueden ser utilizadas en una gran variedad de campos como producción, finanzas, medioambiente, en donde todos sus datos son capturados cronológicamente. Los datos posteriormente son utilizados para establecer relaciones en el tiempo y pueden predecir valores futuros.

Según Villavicencio (2010), los componentes de una serie temporal son:

- Componente de tendencia: Se puede definir como un cambio a largo plazo que se produce en la relación al nivel medio, o el cambio a largo plazo de la media. La tendencia se identifica con un movimiento suave de la serie a largo plazo.
- Componente estacional: Muchas series temporales presentan cierta periodicidad o, dicho de otro modo, variación de cierto período (semestral, mensual, etc.).
- Componente aleatoria: Esta componente no responde a ningún patrón de comportamiento, sino que es el resultado de factores fortuitos o aleatorios que inciden de forma aislada en una serie de tiempo.

Las series temporales se pueden clasificar en *Estacionarias:* Cuando es estable a lo largo del tiempo, los valores de la serie tienden a oscilar alrededor de una media constante. *No estacionarias*: Series en las cuales la tendencia y variabilidad cambian en el tiempo. Los cambios determinan una tendencia a crecer o decrecer a largo plazo. En los accidentes de tránsito se utilizan las series temporales debido a que, por la naturaleza de los sucesos, el tiempo puede determinar patrones importantes, como las horas pico en las cuales más vehículos transitan al mismo tiempo.

2.5 Predicción de accidentes

La previsión de accidentes de tráfico en tiempo real es cada vez más importante para la seguridad pública y la gestión urbana (p. Ej., Planificación de rutas seguras en tiempo real y despliegue de respuesta a emergencias) (Zhou, Wang, Xie, Chen, & Liu, 2020). Existen diversos enfoques para la predicción de accidentes que se han desarrollado en los últimos años, no todos los trabajos relacionados con la predicción de accidentes de tránsito buscan predecir esta variable, sino además busca predecir otros factores asociados inherentemente a los accidentes de tránsito, por ejemplo, la identificación de factores desencadenantes de los accidentes de tránsito, aunque la predicción a partir de estos factores es compleja ya que son muchas variables y tienen una relación compleja con los accidentes, la predicción del tráfico espacio-temporal y la predicción de accidentes en tiempo real, en este último una de las limitaciones es que no tuvieron en cuenta los patrones temporales de los accidentes de tráfico en su modelo. Sin esta información, el poder

predictivo del modelo podría verse debilitado (Ren et al., 2018). Dado lo anterior se podrían realizar estudios que realicen la predicción de accidentes a partir de datos espacio-temporales, para poder identificar si este enfoque arroja mejores resultados en las predicciones.

2.6 Predicción de accidentes espacio – temporales

La dependencia espacio-temporal es una característica importante del tráfico y puede utilizarse para mejorar la precisión de la predicción del tráfico (Ren et al., 2018). En los últimos años se han realizado algunos trabajos que utilizan información del espacio geográfico y tiempo en el cual se presentan los accidentes para crear modelos predictivos de accidentes de tránsito, sin embargo, en este trabajo para poder identificar que avances en investigación se han realizado, se elaboró un análisis del estado del arte en predicciones de accidentes de tránsito basadas en datos espacio-temporales utilizando técnicas de aprendizaje de máquina mediante una revisión sistemática de la literatura.

2.7 Estado del arte de las predicciones espacio-temporales de accidentes de tránsito

Se realizó un estudio de mapeo sistemático donde primero se buscaron otras revisiones para formular preguntas relacionadas con el objetivo de la revisión y luego se seleccionó la estrategia de búsqueda con términos clave en las bases de datos disponibles. Luego se definieron los criterios de exclusión de artículos que no aportan información relacionada con el tema principal y finalmente se realizó la extracción de datos y síntesis de información. El estudio se efectuó sin una fecha de partida para la clasificación de documentos, puesto que se evidenció que la mayoría correspondían a estudios realizados en un tiempo no mayor a 6 años.

Metodología

- Identificación de la necesidad de una revisión:

La estrategia utilizada se basó en las directrices propuestas por (Wen, Li, Lin, Hu, & Huang, 2012) y fue utilizada por (Franco-Bedoya, Ameller, Costal, & Franch, 2017). Para definir la necesidad de una revisión sistemática, se realizaron búsquedas en las revisiones publicadas y disponibles sobre el tema de interés. Se utilizó la ecuación 1 indicada a continuación para buscar en las bases de datos bibliográficas mencionadas en la tabla 2.

Ecuación 1
Fórmula de revisión sistemática

(crash OR "traffic accident" OR "Big traffic accident" OR "road accidents") AND ("Spatial Analysis" OR "spatio-temporal" OR "Temporal Analysis" OR spatio) AND (review OR "systematic review" OR overview OR "state of the art" OR "systematic mapping"))

Tabla 2
Recursos bibliográficos utilizados en la revisión

Base de datos	Enlace
Scopus	https://www.scopus.com
Science direct	https://www.sciencedirect.com/
Springer link	https://link.springer.com/
PubMed	https://www.ncbi.nlm.nih.gov/pubmed/
Nature	https://www.nature.com/

Nota. Bases de datos utilizadas en la búsqueda de artículos académicos
Elaboración propia.

Las palabras clave se seleccionaron en función de lo siguiente: (i) tipo de información que se pretendía recuperar (Accidentes de tránsito), (ii) técnicas orientadas a análisis espacio-temporales, y (iii) palabras clave relacionadas a estudios secundarios.

Todas las bases de datos mostraron resultados utilizando la ecuación 1 de búsqueda, sin embargo, no se encontraron revisiones sistemáticas que utilizaran parámetros orientados a análisis espacio-temporal, dado esto se halló la necesidad de realizar una revisión sistemática de la literatura.

Preguntas de investigación

El objetivo general de esta revisión es resumir y aclarar tendencias, métricas, beneficios y posibles técnicas y arquitecturas que han tenido mejor rendimiento en la predicción de accidentes de tránsito y hallar algunas que aún no se han abordado en la predicción de accidentes. Teniendo esto en cuenta se formularon las siguientes preguntas:

1. P1: ¿Cuáles son las ventajas que tienen los enfoques de machine learning (ML) para la predicción de accidentes de tránsito en comparación con los enfoques tradicionales?

2. P2: ¿Qué técnicas de machine learning (ML) se utilizan actualmente para predecir accidentes de tránsito y cuales han tenido mejores resultados?

Diseño de búsqueda y selección de estudios

Una vez que se identificó la necesidad de una revisión y se formularon las preguntas de investigación, se desarrolló la estrategia de búsqueda para encontrar artículos de investigación, libros de capítulos, actas de congresos y otros artículos de revisión en las bases de datos que se muestran en la tabla 1. Similar a la ecuación 1 de búsqueda de la literatura, se utilizaron palabras clave relacionadas con los accidentes de tránsito. No se utilizaron palabras clave como “Accidentes”, puesto que se encontraron pocos resultados relevantes sobre el tema ya que hablaban acerca de otro tipo de accidentes en contextos diferentes a los de tránsito y transporte. Las palabras clave principales fueron separados por

el operador “AND”, y las palabras clave relacionadas se vincularon usando el operador “OR”, como se muestra en la ecuación 2:

Ecuación 2

Fórmula de revisión sistemática ajustada

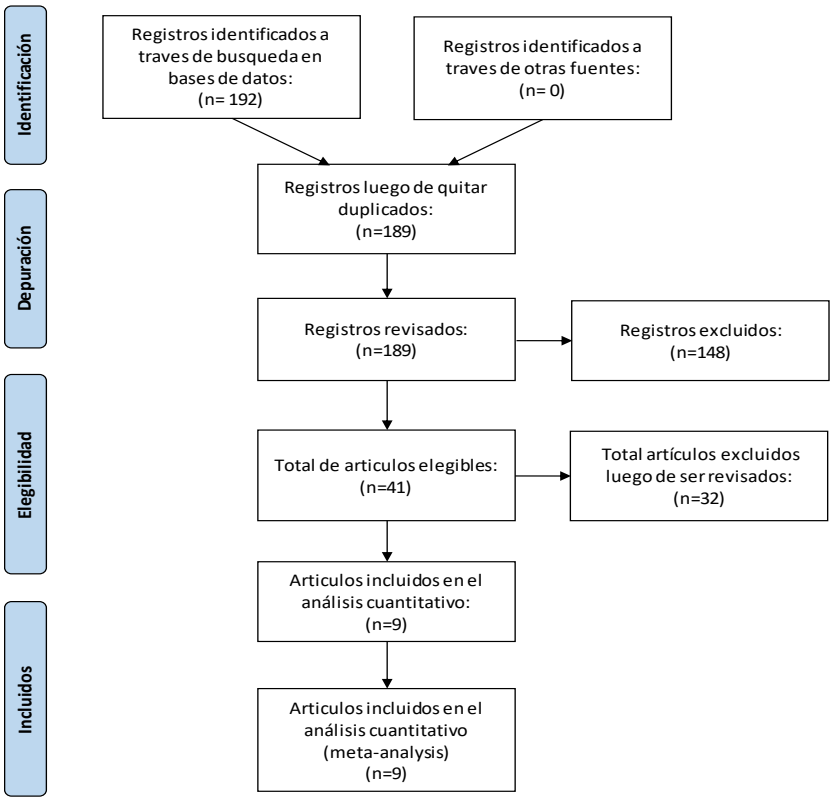
(crash OR "traffic accident" OR "road accidents") AND ("Spatial Analysis" OR "spatio-temporal" OR "Temporal Analysis" OR spatio) AND ("deep learning" OR "machine learning" OR "neural networks")

En la búsqueda de literatura usando la ecuación 2, se encontraron 192 artículos candidatos de los cuales fueron eliminados los que fueron repetidos (a) repetidos (el mismo estudio se encontró en diferentes bases de datos); (b) de diferentes tipos (libros, carteles, artículos breves, cartas y resúmenes); (c) escrito en otros idiomas (idiomas distintos del inglés).

Luego se realizó un proceso de lectura rápida es decir (título, resumen y conclusión) para detectar artículos que pudieran contribuir a responder las preguntas de investigación, luego de este proceso de selección que se resume en la figura 1, se identificaron 11 artículos de investigación relevantes para extraer y resumir la información.

Figura 1

Diagrama de flujo de prisma para el proceso de búsqueda y selección de artículos



Nota. El gráfico representa el proceso de búsqueda y filtrado de artículos científicos relacionados con el tema principal. Tomado de (Moher, Liberati, Tetzlaff, Altman, & Group, 2009)

Extracción y síntesis de datos

En esta etapa primero se leyeron las publicaciones seleccionadas para extraer la información que responda a las preguntas de investigación. Luego se registró cada artículo en una tarjeta de extracción de datos propuesta por Wen et al. (2012) con algunas adaptaciones para esta revisión. La tarjeta contenía información sobre el identificador de la publicación, el año, el nombre de la publicación, las preguntas de investigación relacionadas y la información en sí.

Tabla 3

Aporte de cada publicación a cada pregunta de investigación

Identificador de publicación	Año	P1	P2	Referencias
P1	2021	X	X	Yu et al. (2021)
P2	2021	X	X	de Medrano and Aznarte (2021)
P3	2020	X	X	W. Bao, Yu, and Kong (2020)
P4	2018	X	X	Ren et al. (2018)
P5	2018	X	X	Yuan, Zhou, and Yang (2018)
P6	2017	X		Shafabakhsh, Famili, and Bahadori (2017)
P7	2021	X		Amiri, Nadimi, Khalifeh, and Shams (2021)
P8	2019	X	X	Zhou (2019)
P9	2019	X	X	J. Bao, Liu, and Ukkusuri (2019)

Elaboración propia

En el paso final se sintetizó toda la información recolectada y se obtuvo evidencia para poder responder las preguntas de investigación propuestas.

Beneficios del ML en la predicción de accidentes de tránsito (P1)

Existe literatura sobre aplicaciones de ML en la predicción de accidentes, como (Ren et al., 2018), que indica que con la observación de datos históricos de accidentes recopilados, se puede obtener una visión general de las zonas de riesgo, sin embargo, esta distribución varía mucho del día de la semana o la hora del día y existen factores complejos que pueden afectar el riesgo de accidente como la densidad de población, flujo de tráfico, estado de la vía, el clima, etc. Aunque la investigación de los accidentes de tráfico desde la perspectiva de la Inteligencia Artificial en sí misma es todavía un campo joven, la importancia de la aplicación de algoritmos de inteligencia artificial se ha convertido en el elemento central de un gran número de estudios sobre Sistemas Inteligentes de Transporte.

Además, la tecnología de aprendizaje profundo ha crecido rápidamente durante los últimos años en los campos de investigación de la visión por ordenador, el procesamiento del lenguaje natural, la inteligencia artificial y el reconocimiento de patrones. (J. Bao et al., 2019). Algunos estudios recientes también han comenzado a aplicar la tecnología de

aprendizaje profundo en los campos del transporte, como la predicción del flujo de tráfico a corto plazo (Li, Wang, Liu, Bigham, & Ragland, 2013), la estimación del tiempo de viaje en la red y la inferencia del modo de viaje (Dabiri & Heaslip, 2018). En comparación con los modelos estadísticos tradicionales, el aprendizaje profundo basado en machine learning puede modelar relaciones no lineales complejas utilizando la representación de características distribuidas y jerárquicas (J. Bao et al., 2019), lo que ha demostrado su superioridad en la predicción del flujo de tráfico a corto plazo y la velocidad del tráfico. Por lo tanto, la aparición de la tecnología de big data y de aprendizaje profundo tiene el potencial de avanzar en la comprensión de los problemas de transporte tradicionales y permitir la predicción del riesgo de colisión a corto plazo en toda la ciudad (J. Bao et al., 2019).

Según Yuan et al. (2018) los trabajos realizados con técnicas diferentes al ML tienen mayores problemas a la hora de contemplar la información temporal, lo cual puede afectar la precisión del modelo, y por eso proponen un enfoque de aprendizaje profundo para la predicción de accidentes de tráfico basado en grandes datos heterogéneos.

Otros autores como J. Bao et al. (2019), utilizaron dos métodos, modelos econométricos y modelos de ML, para la tarea de predicción del riesgo de colisión semanal, los modelos econométricos muestran generalmente un mejor rendimiento que los modelos de aprendizaje automático. Para la tarea de predicción del riesgo de colisión diaria, los modelos de aprendizaje automático obtienen mejores resultados que los modelos econométricos.

Dado lo anterior se encontró que, aunque existen modelos diferentes a los de ML que han efectuado predicciones de accidentes de tránsito, sin embargo, a la hora de tener encuentra diversas variables especialmente las de tipo temporal, se encontraran mejores resultados utilizando los modelos de ML, medidos en términos de MSE, MAE y MAPE las cuales son medidas comunes utilizadas para evaluar el rendimiento de un modelo de predicción.

Técnicas de ML utilizadas en predicción de accidentes (P2)

El modelo de ML utilizado por Ren et al. (2018), fue un modelo profundo sobre la base de una red neuronal recurrente para inferir el riesgo de accidentes de tránsito, y en él se introdujeron los datos procesados para predicción, para este modelo profundo se utilizó un método basado en LSTM, la razón del uso de LSTM es que esta puede capturar la característica periódica del accidente de tránsito, y debido a que las RNN tradicionales muestran un pobre rendimiento cuando se analiza un largo periodo de tiempo y para su uso se utilizó La arquitectura de TARPML se basa en Keras, que es una biblioteca de aprendizaje profundo de Python. Los modelos de referencia seleccionado son Lasso, Ridge, Support Vector Regression (SVR), Decision Tree Regression (DTR), Random Forest Regression (RFR), Multilayer Per- ceptron(MLP) y Autoregressive Moving Average Model (ARMA), sin embargo, el mejor resultado se obtuvo con TARPML. Otros trabajos como de Medrano and Aznarte (2021) realizaron una regresión basada en una red neuronal recurrente para modelar series temporales de procesos espaciales usando los modelos XSTNN y STNN en donde asociaron variables exógenas y se encontró que XSTNN

aprende a distinguir mejor entre rangos de tiempo y zonas espaciales, es posible encontrar otras situaciones en las que, de nuevo, este modelo ofrece más información y asimila mejor la dinámica del sistema.

Otros trabajos como Yuan et al. (2018) utilizaron predicción de accidentes de tráfico utilizando el modelo de red neuronal Convolutional Long Short-Term Memory (ConvLSTM), y proponen un marco de aprendizaje profundo que utiliza la red neuronal convolucional de memoria a corto plazo a largo plazo (LSTM) con un enfoque de conjunto de modelos para abordar aún más la heterogeneidad espacial en los datos y mejorar la precisión de la predicción, Long Short-Term Memory (LSTM) es un tipo de estructura de nodo de red neuronal recurrente que se sabe que tiene un buen rendimiento al manejar datos de series de tiempo con auto correlaciones temporales. Un nodo en una red neuronal LSTM consta de una celda de memoria, una puerta de entrada, una puerta de salida y una puerta de olvido. Durante la fase de entrenamiento, se aprende una función ponderada en cada una de las puertas de un nodo LSTM para controlar la capacidad de "memorización" y "olvido" de la red, lo cual es ideal para realizar predicciones con variables temporales.

En la tabla 4, se presenta un resumen sobre las técnicas utilizadas en los artículos seleccionados:

Tabla 4
Técnicas utilizadas en artículos revisados

Características del conjunto de datos	Tarea	Tipo de ML	Marco o lenguaje	Año	Referencias
Datos temporales del accidente, latitud y longitud	Predecir probabilidad de accidente para cada cuadricula geográfica	Redes Neuronales LSTM	Python Keras	2018	(Ren et al., 2018)
Variables urbanas y ambientales, como los accidentes pasados, la intensidad del tráfico, la precipitación, la temperatura y el viento, que se emparejan en el mapa con cada celda de la cuadrícula	Predecir probabilidad de accidente para cada cuadricula geográfica	Redes Neuronales Profundas XSTNN STNN	PyTorch	2021	(de Medrano & Aznarte, 2021)
Datos demográficos, datos de GPS de taxis de la ciudad y datos asociados a los accidentes	Predecir probabilidad de accidente para cada cuadricula geográfica	Red Neuronal Convolucional ConvLSTM STCL-Net	Python ArcGIS TensorFlow	2019	(J. Bao et al., 2019)

Datos de características de las vías, meteorológicas, demográficas y de accidentes	Predecir probabilidad de accidente para cada cuadricula geográfica	Red Neuronal Convolucional Res Net ASRAP	Python	2019	(Zhou, 2019)
Datos de accidentes, histórico GPS de taxis, datos meteorológicos, y datos de las carreteras	Predecir probabilidad de accidentes para cada punto donde existe una carretera	Red Neuronal Convolucional DSTGCN	Python	2021	(Yu et al., 2021)
Datos de condición de vías, clima, volumen de tráfico	Predecir probabilidad de accidentes para cada punto donde existe una carretera	Red Neuronal Convolucional ConvLSTM	TensorFlow	2018	(Yuan et al., 2018)
Datos de condición del clima, histórico de accidentes	Predecir probabilidad de accidentes por zona geográfica	Red Neuronal Recurrente GCRN	Python	2020	(W. Bao et al., 2020)

Elaboración propia.

2.8 Trabajo Desarrollado

De acuerdo con la problemática expuesta en la predicción de accidentes de tránsito en la ciudad de Manizales, en el presente trabajo se utilizan técnicas de minería de datos, en específico de aprendizaje automático, para predecir los accidentes de tránsito en la ciudad. Para ello, se realizaron las siguientes acciones:

- Construir una base de datos histórica de los accidentes de tránsito en la ciudad de Manizales.
- Entender e identificar las características de los datos y su significado en el contexto del tráfico vehicular en la ciudad.
- Preparar los datos e implementar diferentes modelos de aprendizaje automático para la predicción de accidentes de tránsito.
- Evaluar los resultados de cada modelo de acuerdo con su desempeño, seleccionando el mejor modelo.
- Elaborar un informe final con los resultados y conclusiones obtenidos.
- Socializar y publicar el trabajo desarrollado con el fin de contribuir a la prevención de accidentes de tránsito en la ciudad de Manizales y mejorar la calidad de vida de sus habitantes.

3. METODOLOGÍA

El desarrollo de este trabajo se basa en el modelo de procesos CRISP-DM (Clifton & Thuraisingham, 2001) para guiar la metodología de trabajo en la realización de un sistema de predicción de accidentes de tránsito en la ciudad de Manizales. El modelo CRISP-DM es un enfoque comúnmente utilizado en proyectos de ciencia de datos, que se enfoca en cuatro grandes fases: comprensión del problema, comprensión de los datos, modelado y evaluación.

En la primera fase, se lleva a cabo un análisis detallado de la recolección de los datos, su estructura y su significado en el contexto del proyecto. Es importante comprender los objetivos de modelo y los requisitos del proyecto para definir el alcance y los límites del mismo.

En la segunda fase, se realiza una comprensión de los datos, incluyendo la identificación de posibles problemas y la selección de las variables relevantes para el modelo. Es necesario también realizar una limpieza y preparación de los datos para su posterior análisis.

Una vez los datos están preparados, se procede a la etapa de experimentación con diferentes modelos de predicción. Se evalúan diversas técnicas de aprendizaje automático con el objetivo de seleccionar la más adecuada para el problema en cuestión. La selección del modelo adecuado es fundamental para lograr una buena precisión en las predicciones.

En la última fase, se realiza la afinación de los parámetros del modelo seleccionado para lograr una mayor precisión en las predicciones. Además, se evalúa el rendimiento del modelo en términos de las métricas obtenidas en las diferentes técnicas y configuraciones propuestas para la predicción.

3.1 Recolección y entendimiento.

En este estudio se abarcan los accidentes de tránsito ocurridos en la jurisdicción del municipio de Manizales ciudad capital del departamento de Caldas, Colombia con una población de 434.403 habitantes (DANE, 2019), en este estudio se consideran todos los tipos de accidentes ocurridos en la ciudad de Manizales, incluyendo aquellos que resultan en lesiones personales, daños a vehículos o bienes materiales, siempre y cuando la autoridad de tránsito y transporte haya elaborado un croquis correspondiente. Al mencionar vehículos, nos referimos a todos los motorizados, así como también a las bicicletas, dado que se clasifican como un medio de transporte. Los datos se recolectan a través de un sistema de información el cual es alimentado por medio de dispositivos móviles de tipo PDA (Personal Digital Assitant), los datos recolectados pertenecen al levantamiento del Croquis el cual se define como el plano descriptivo de los pormenores de un accidente de tránsito donde resulten daños a personas, vehículos, inmuebles, muebles o animales, levantado en el sitio de los hechos por el agente, la policía de tránsito o por la autoridad competente (Terrestre, 2002)

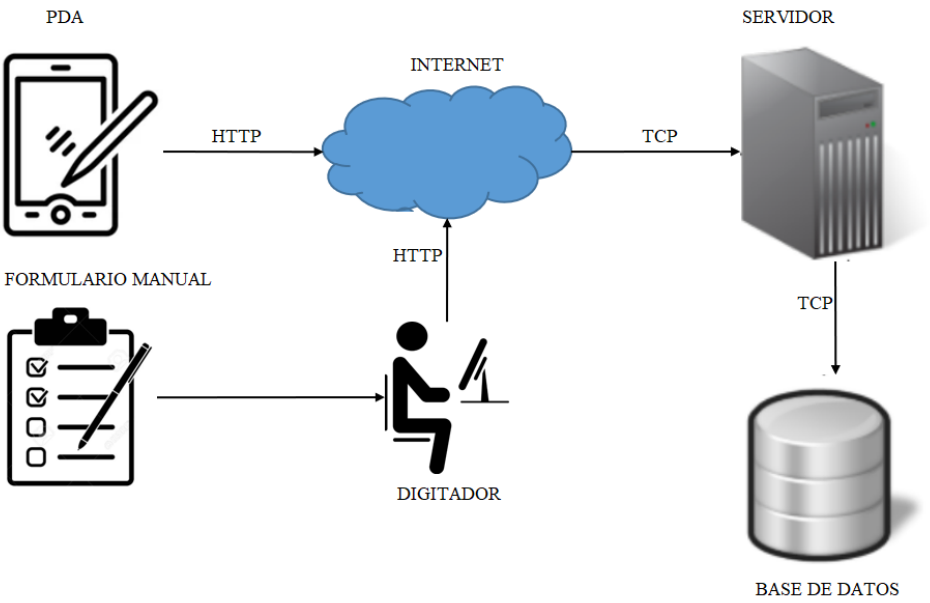
Según el código nacional de tránsito terrestre (Terrestre, 2002) Articulo 149 el agente de tránsito que conozca el hecho levantará un informe descriptivo de sus pormenores, con

copia inmediata a los conductores, quienes deberán firmarlas y en su defecto, la firmará un testigo. El informe contendrá por lo menos:

- Lugar, fecha y hora en que ocurrió el hecho.
- Clase de vehículo, número de la placa y demás características.
- Nombre del conductor o conductores, documentos de identidad, número de la licencia o licencias de conducción, lugar y fecha de su expedición y número de la póliza de seguro y compañía aseguradora, dirección o residencia de los involucrados.
- Nombre del propietario o tenedor del vehículo o de los propietarios o tenedores de los vehículos.
- Nombre, documentos de identidad y dirección de los testigos.
- Estado de seguridad, en general, del vehículo o de los vehículos, de los frenos, de la dirección, de las luces, bocinas y llantas.
- Estado de la vía, huella de frenada, grado de visibilidad, colocación de los vehículos y distancia, la cual constará en el croquis levantado.
- Descripción de los daños y lesiones.
- Relación de los medios de prueba aportados por las partes.
- Descripción de las compañías de seguros y números de las pólizas de los seguros obligatorios exigidos por este código.
- En todo caso en que produzca lesiones personales u homicidio la autoridad de tránsito deberá enviar a los conductores implicados a la práctica de la prueba de embriaguez, so pena de considerarse falta disciplinaria grave para el funcionario que no dé cumplimiento a esta norma.
    - El informe o el croquis, o los dos, serán entregados inmediatamente a los interesados y a la autoridad instructora competente en materia penal.
    - El funcionario de tránsito que no entregue copia de estos documentos a los interesados o a las autoridades instructoras, incurrirá en causal de mala conducta.
    - Para efectos de determinar la responsabilidad, en cuanto al tránsito, las autoridades instructoras podrán solicitar pronunciamiento sobre el particular a las autoridades de tránsito competentes.

Figura 2

Esquema de recolección de datos de los accidentes de tránsito



Elaboración propia.

En la figura 2 se puede observar la relación entre los elementos que intervienen en la recolección y almacenamiento de los datos. Cuando el sistema principal de PDA presenta fallas técnicas para el registro de los accidentes, la información es diligenciada de forma manual en un formulario de papel, el cual es llevado por parte del agente de tránsito hasta la sede de la secretaría de tránsito de la ciudad y allí es ingresada la información al sistema. En ocasiones la información recolectada en las PDA o de forma manual presenta información no precisa o que corresponda con la realidad de los hechos, causados por mal diligenciamiento de la información por parte del agente de tránsito, los cuales comprenden letra no legible, campos no diligenciados o diligenciados con información errónea, estos fallos generan patrones irregulares en los datos como ruido, rupturas en las secuencias de datos, datos incompletos y duplicidad que deben posteriormente corregirse.

El valor de las cantidades de accidentes ocurridos es interpretado en cantidades diarias de croquis levantados por las autoridades de tránsito de la ciudad de Manizales, constituyendo una serie de valores a lo largo del tiempo. De acuerdo con lo anterior, el registro de los accidentes de tránsito es en esencia una serie de tiempo, donde el valor numérico de la cantidad de accidentes de tránsito es registrado en intervalos de tiempo no uniformes, sin embargo, pueden ser resumidos y tratados en cualquier unidad temporal.

Para este trabajo se realizó una solicitud por medio de un derecho de petición a la secretaría de tránsito y transporte de la ciudad de Manizales, en donde se solicitó toda la información disponible desde que se tenga registro sistematizado de la ocurrencia de los accidentes de tránsito, disponibilizando toda la información almacenada relacionada con los accidentes de tránsito de la ciudad, posteriormente la información recibida contiene un conjunto de datos de accidentes desde el año 2002 hasta el año 2020, sin embargo, sólo los

accidentes ocurridos desde el año 2013 contienen la información de las coordenadas Latitud y Longitud del sitio de cada accidente, para este trabajo seleccione un conjunto de datos entre los años 2002 a 2019 de acuerdo a la relevancia que representan los accidentes más actuales sobre años anteriores y excluyendo el año 2020 debido a que contiene variaciones atípicas causadas por la pandemia del COVID 2019.

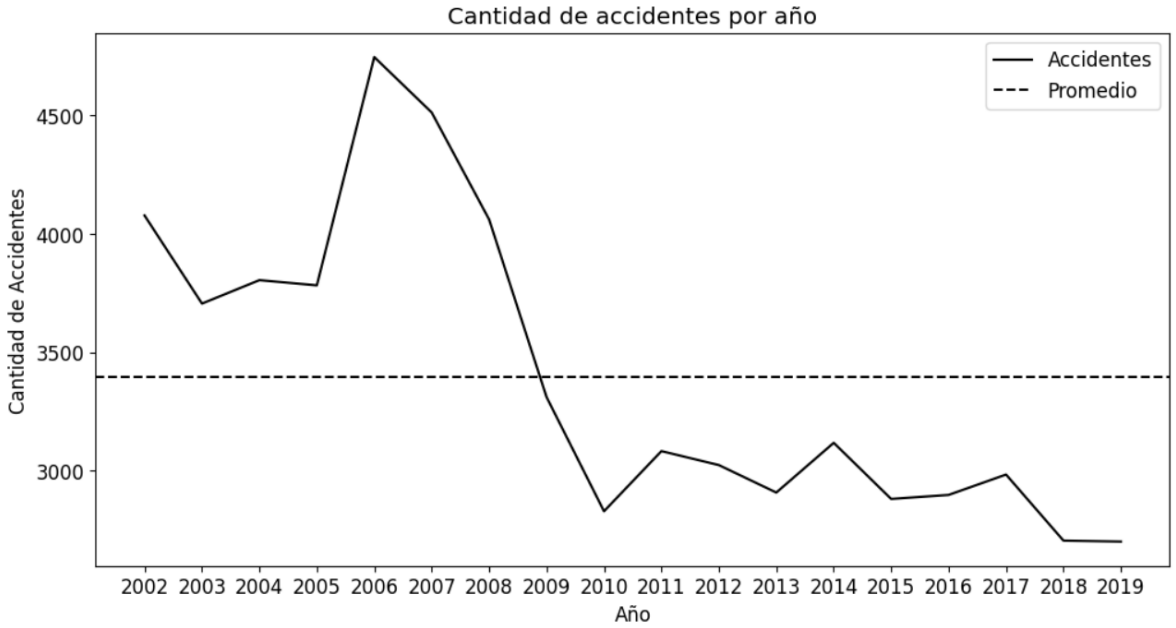
Con el propósito de conocer la forma de los datos a un mayor nivel de detalle, recopilé procesé y agrupé los datos de toda la ciudad construyendo un conjunto de datos de exploración. Posteriormente a través del lenguaje de programación Python y de la herramienta de visualización de datos Microsoft Power BI, se generaron diferentes gráficas de ocurrencia de accidentes a nivel general y por zonas de la ciudad, en diferentes periodos de tiempo específicamente Anual, Mensual, Diario y por horas, a continuación, se presentan dichas gráficas y su interpretación.

La cantidad total de accidentes de tránsito para el periodo comprendido entre el 1 de enero de 2002 y el 31 de diciembre de 2019 es de 61.156 accidentes de tránsito, los accidentes de tránsito tenidos en cuenta en este estudio incluyen todas las categorías, en las cuales existen accidentes con heridos o sólo daños materiales, producidos en cualquier tipo de vehículo.

La cantidad promedio diaria de accidentes es de aproximadamente 9.3 accidentes diarios. En la figura 3 podemos observar la cantidad de accidentes por cada año.

Figura 3

Grafica de la cantidad de accidentes anual



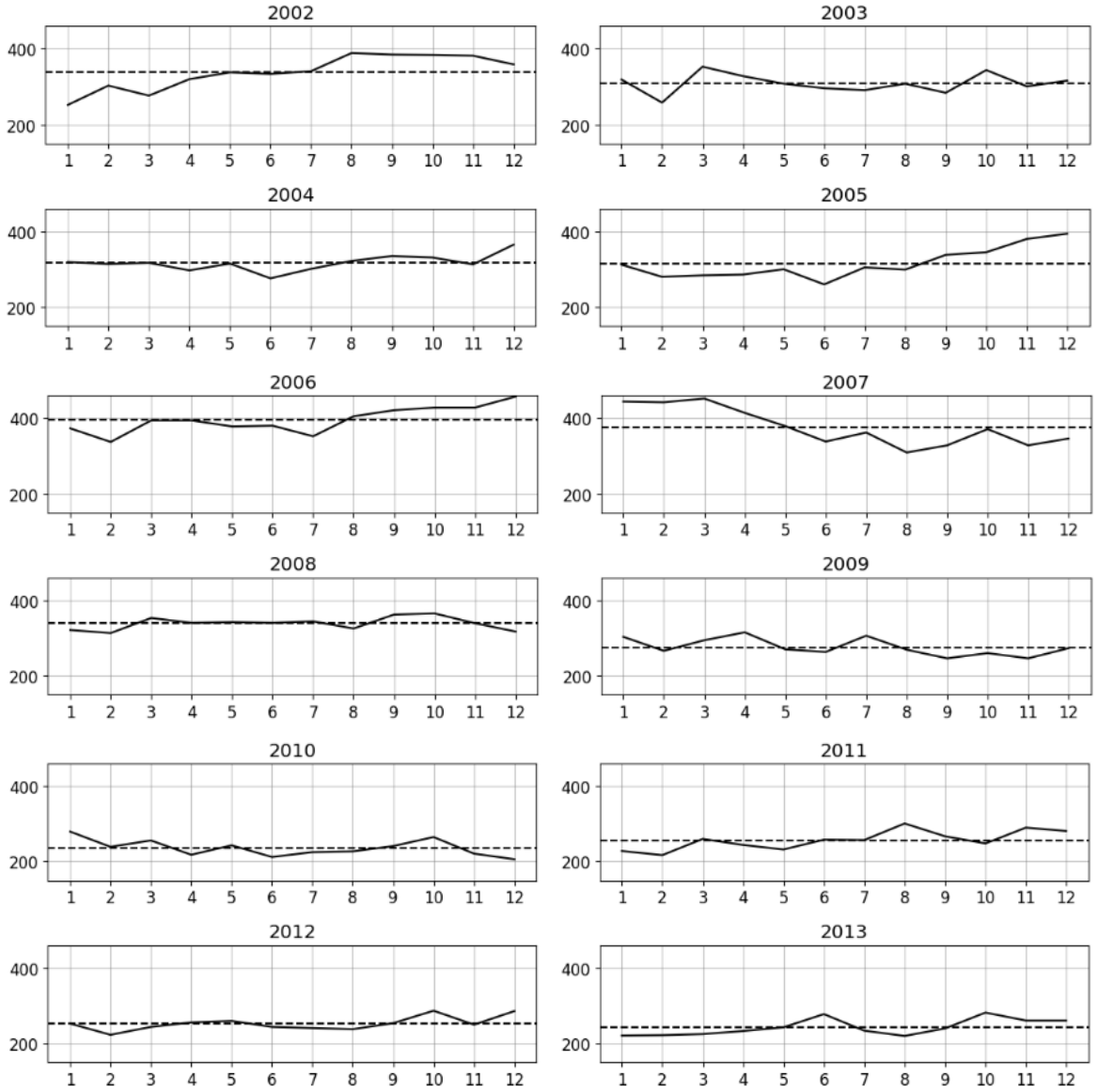
Elaboración propia.

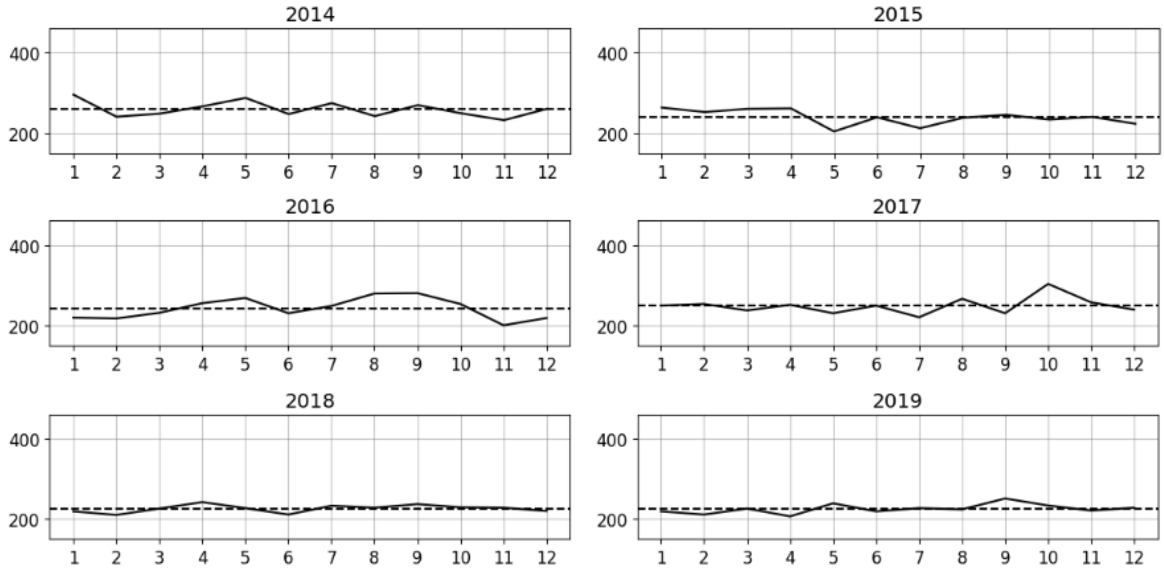
El promedio anual durante este periodo es cercano a los 3.396 accidentes cada año, se omite el año 2020 debido a que contiene variaciones atípicas afectadas por la pandemia del COVID 2019.

Por otra parte, se observa que de manera general la ocurrencia de accidentes de tránsito viene disminuyendo desde el año 2006 y ha tenido valores más constantes desde el año 2010, lo cual podría ser consecuencia de diferentes medidas tomadas por la administración municipal, como lo son la construcción de diferentes obras de infraestructura en toda la ciudad y su área metropolitana.

Figura 4

Grafica de accidentes mensuales entre 2002 y 2019





Elaboración Propia.

Al observar el gráfico podemos notar que el año 2006 tuvo el mayor número de accidentes de tráfico, con un total de 4,056, mientras que el año 2019 tuvo el menor número de accidentes de tráfico, con un total de 2,682. Esta tendencia general a la disminución del número de accidentes de tráfico es alentadora, y puede deberse a un mayor énfasis en la seguridad en las carreteras, la educación sobre la conducción responsable y la mejora en las condiciones del vehículo y de la infraestructura.

Además, se puede observar que los meses de mayo, junio y octubre parecen tener la mayor cantidad de accidentes de tráfico en promedio, mientras que los meses de enero, febrero y diciembre parecen tener la menor cantidad de accidentes de tráfico. Esto puede deberse a una variedad de factores, como las condiciones climáticas, los días festivos y la cantidad de tráfico en la carretera durante esos meses.

En conclusión, los datos proporcionados muestran una tendencia general a la disminución del número de accidentes de tráfico a lo largo del tiempo, con algunos picos ocasionales. Además, hay una variación estacional en la cantidad de accidentes de tráfico, con una mayor cantidad en los meses de mayo, junio y octubre y una menor cantidad en los meses de enero, febrero y diciembre.

Se puede observar que existe una alta variabilidad en la ocurrencia de los accidentes para cada año y mes analizado, esto se puede explicar debido a la complejidad y la gran cantidad de factores que pueden influir en la ocurrencia de un accidente de tránsito. Estos factores pueden incluir desde las condiciones meteorológicas, el tipo de vehículo, la infraestructura de la vía, el comportamiento del conductor, la densidad del tráfico, entre otros.

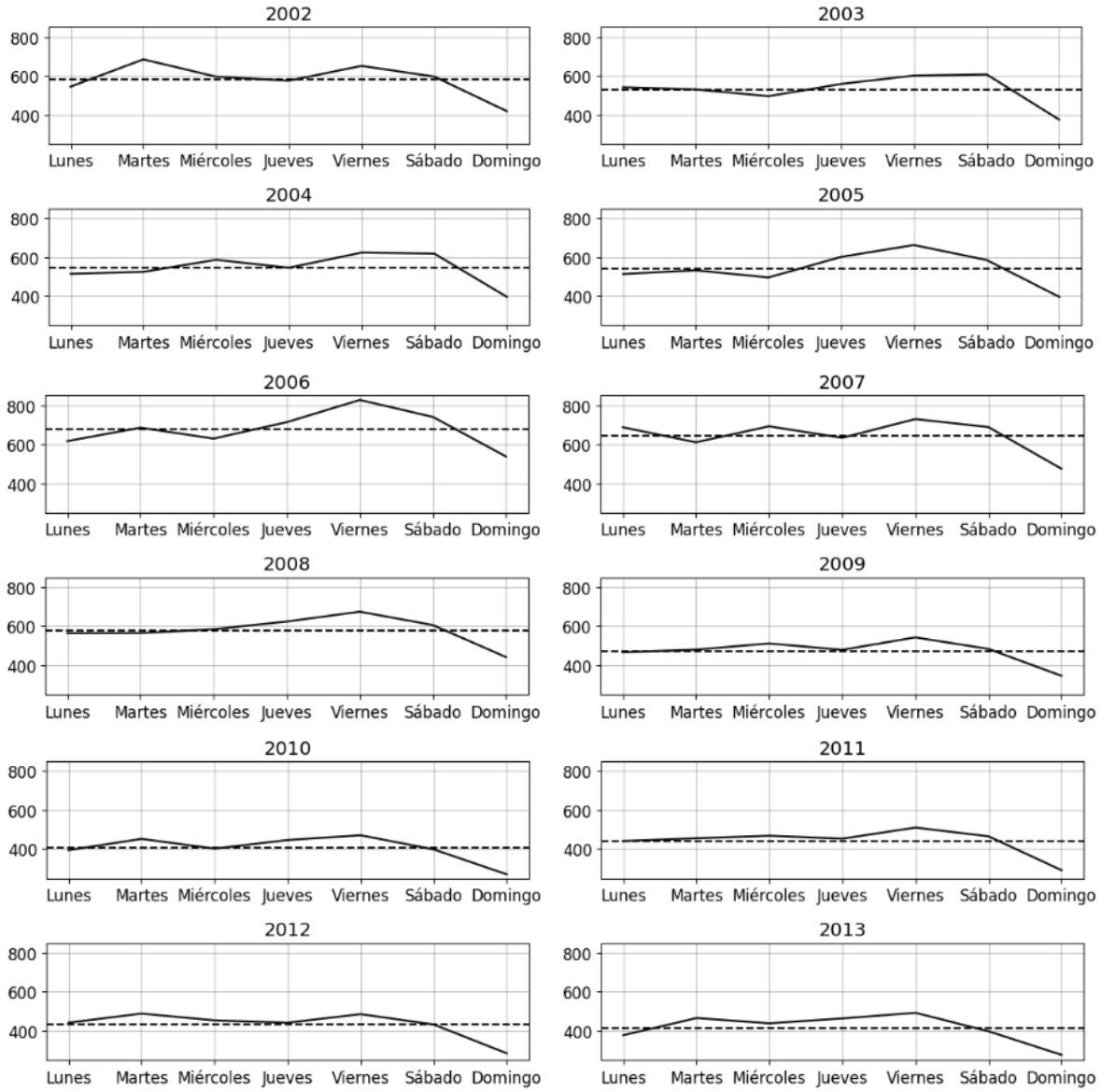
Además, al realizar el análisis visual de los datos se puede observar que estos no ocurren de manera uniforme en el tiempo ni en el espacio, por lo que la frecuencia y la gravedad de los accidentes pueden variar ampliamente de un día a otro, de una hora a otra, de una vía a otra, y de una región a otra.

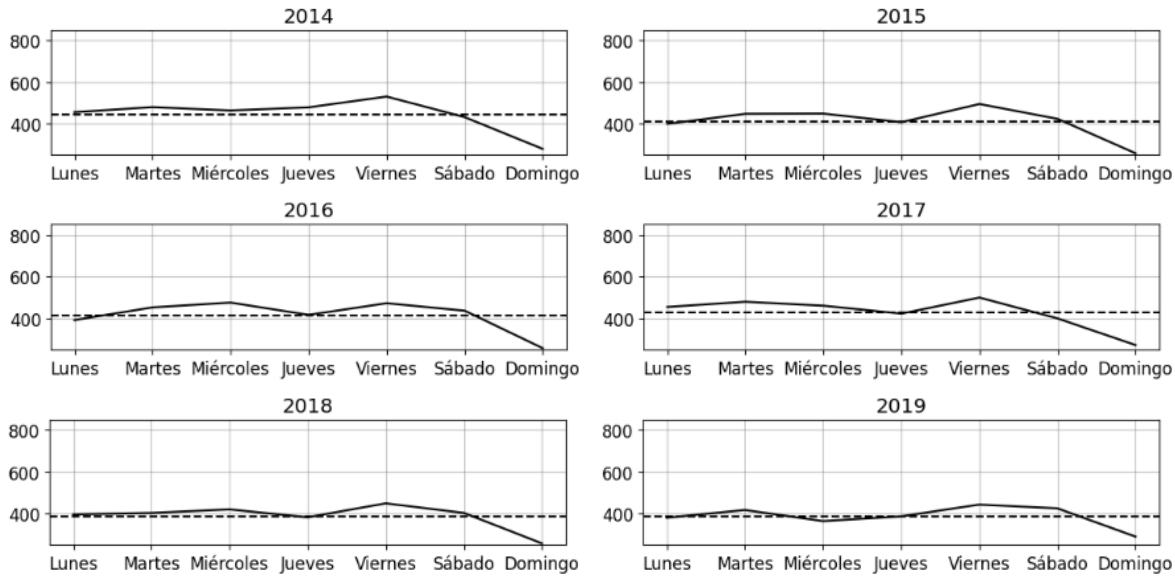
Esta variabilidad y heterogeneidad de los datos de accidentes de tránsito hace que su análisis y modelamiento sean un desafío.

A continuación, vamos a ver la ocurrencia de los accidentes de tránsito a través de los mismos años según el día de la semana:

Figura 5

Gráfica de accidentes por día de la semana entre 2002 y 2019





Elaboración Propia.

Al graficar la frecuencia de accidentes de tránsito por día de la semana para cada año, se observa una tendencia general hacia una mayor cantidad de accidentes los días viernes y sábados, mientras que los domingos suelen ser los días con menor cantidad de accidentes. Sin embargo, es importante destacar que hay cierta variabilidad en esta tendencia a lo largo de los años.

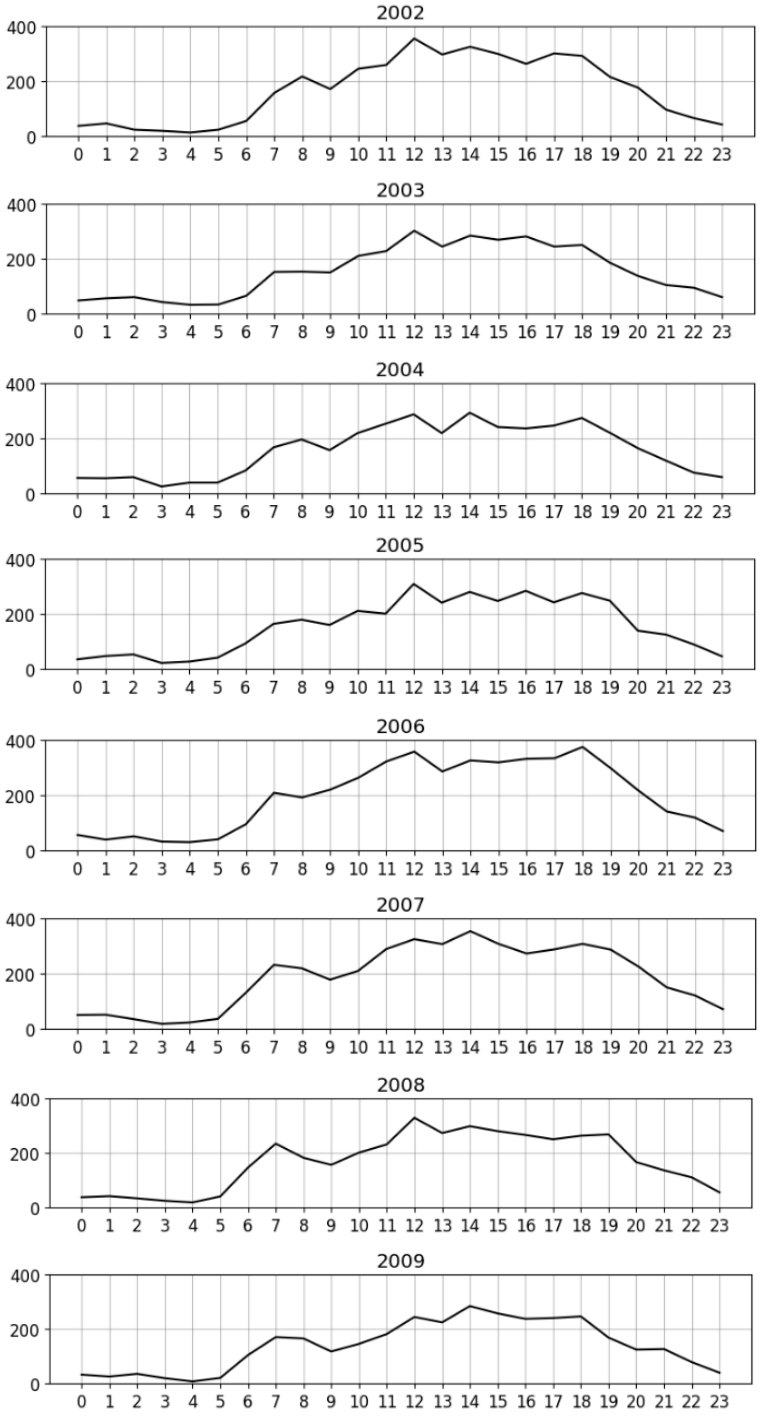
También es interesante notar que la frecuencia de accidentes varía considerablemente entre los días de la semana, lo que sugiere que hay ciertos factores específicos que influyen en la ocurrencia de accidentes en días particulares. Por ejemplo, los días viernes y sábados suelen ser días en los que se realizan más actividades sociales y se consume más alcohol, lo que puede contribuir a un aumento en la cantidad de accidentes.

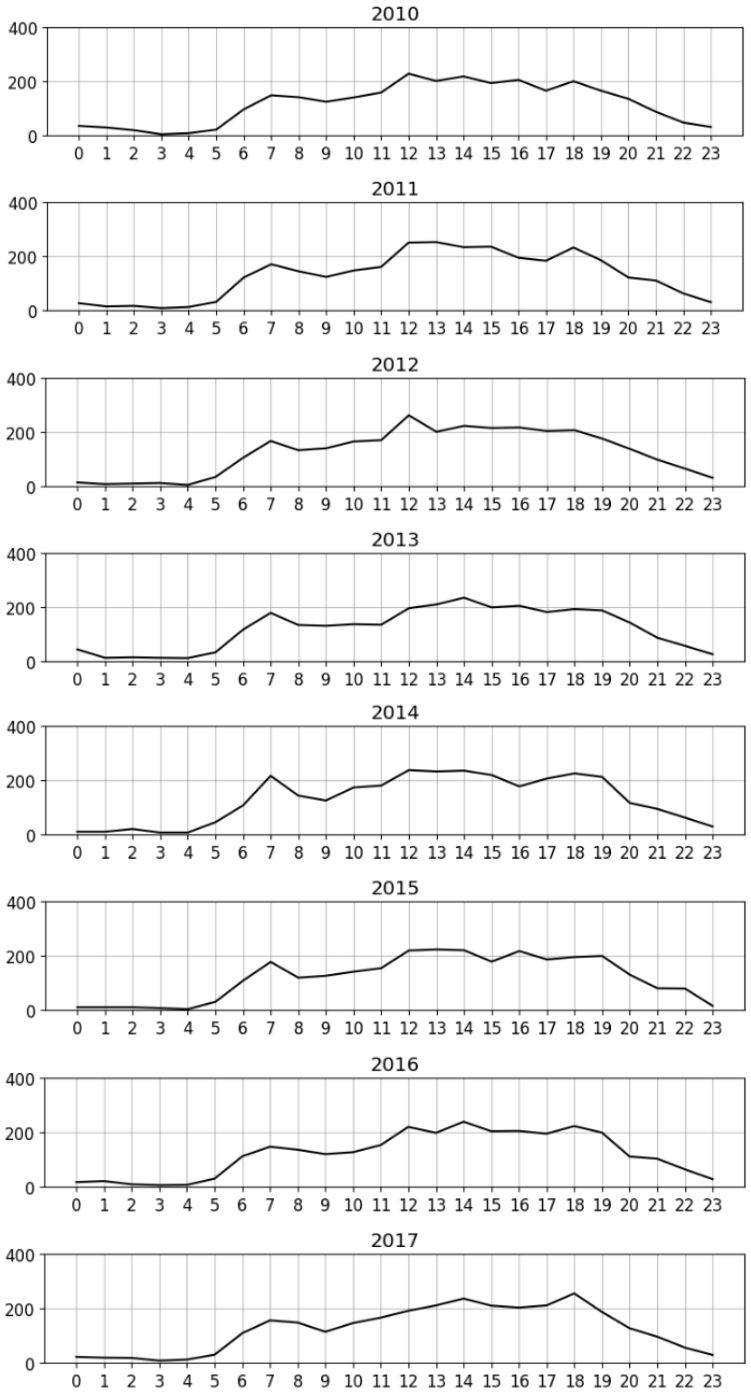
En general, estos resultados sugieren que hay ciertas tendencias en la ocurrencia de accidentes de tránsito por día de la semana y año, y que es importante considerar estos patrones al diseñar políticas y estrategias para prevenir y reducir la cantidad de accidentes. Por ejemplo, se podrían implementar medidas específicas para reducir la cantidad de accidentes los días viernes y sábados, como aumentar la vigilancia policial en las carreteras o promover el uso de transporte público en lugar de conducir bajo la influencia del alcohol.

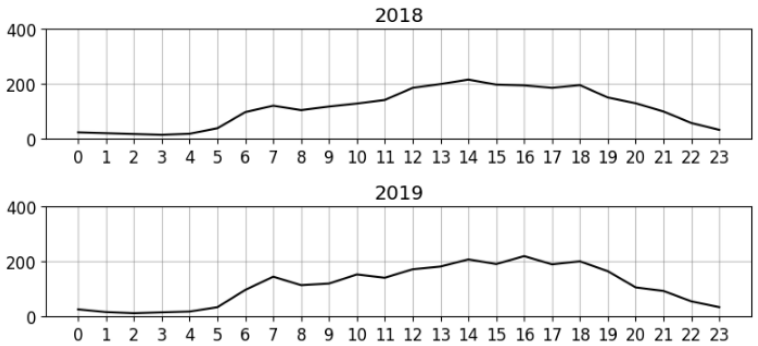
A continuación, se presenta un gráfico de la cantidad de accidentes por año según la hora del día

Figura 6

Grafica de accidentes por hora del día entre 2002 y 2019




2010
2011
2012
2013
2014
2015
2016
2017
400
200
0
0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23



Elaboración Propia.

En cuanto a la hora del día en que se producen los accidentes, se puede observar que durante el día (de 7:00 a 19:00) es cuando se produce la mayoría de los accidentes. En particular, las horas con mayor cantidad de accidentes son las 13:00 y las 14:00, mientras que las horas con menor cantidad de accidentes son las 4:00 y las 5:00. Por la noche (de 19:00 a 7:00) se producen menos accidentes en general, y la hora con mayor cantidad de accidentes es las 14:00.

Se observa también que en la mañana existe un pico de accidentes a las 7:00, este se puede dar debido a que es el horario en el que muchas personas se trasladan a sus lugares de trabajo o estudio. Durante este horario, hay mayor congestión en las vías y las personas pueden estar apuradas o cansadas, lo que aumenta el riesgo de accidentes de tránsito.

La distribución de accidentes a lo largo de los años es estable es decir que no hay mucha variación de un año a otro, esto se debe principalmente porque hay patrones de comportamiento más estables en el tiempo, como los picos de tráfico durante las horas pico de trabajo o las congestiones en las principales vías de la ciudad.

Con el objetivo de identificar una zona más pequeña de la ocurrencia de los accidentes dentro de la jurisdicción de Manizales, se abordaron múltiples enfoques experimentales para realizar la creación de las zonas, primero se establecieron polígonos de diferentes tamaños, sin embargo, se evidenció que no existen suficientes datos como para poder hacer una serie de tiempo en un espacio tan pequeño, por consiguiente, se crearon 9 zonas en las cuales se clasificaron nuevamente los accidentes.

3.2 Tratamiento de datos

El estudio analiza los datos de accidentes de tránsito ocurridos en la ciudad de Manizales desde 2002, los cuales están almacenados en una base de datos relacional. Para realizar el estudio, se extrajeron los datos de la base de datos y se convirtieron en un archivo plano para su análisis. Los datos son recolectados manualmente por la autoridad de tránsito que acude a la escena del accidente y luego son digitalizados y validados por el personal de la secretaría de movilidad de Manizales antes de ingresarlos al sistema. La mayoría de los datos se almacenan de manera correcta, evitando valores erróneos, y durante la validación se encontraron solo algunas pequeñas partes con valores atípicos o faltantes.

En la figura 7, vemos el formato de la información suministrada por la secretaria de movilidad de Manizales:

Figura 7

Imagen de la data sin transformación suministrada por la secretaria de movilidad del municipio de Manizales

	NRO_CROQUIS	Fecha del accidente	Hora del accidente	Dirección del accidente	Clase de accidente	Latitud	Longitud
0	M002994	5/4/2002	7:20:00	NaN	Choque	NaN	NaN
1	A000003761	9/7/2013	13:45:00	CR 25 FRENTE VIVIENDA N 55B 73	Choque	5.060858	-75.493692
2	A000003762	10/7/2013	9:00:00	CL 45 CR 24A	Choque	5.065189	-75.502808
3	A000003763	10/7/2013	0:00:00	AV KEVIN ANGEL CL 63 Y 64	Choque	5.058528	-75.485645
4	A000003764	10/7/2013	0:00:00	CL 51B FRENTE AL N 17 08	Choque	5.069484	-75.496214
...	...	...	...	...	...	...	...
63987	9995	29/10/2004	11:20:00	NaN	Choque	NaN	NaN
63988	9996	29/10/2004	10:20:00	NaN	Choque	NaN	NaN
63989	9998	28/10/2004	19:20:00	NaN	Choque	NaN	NaN
63990	9999	28/10/2004	18:40:00	NaN	Choque	NaN	NaN
63991	NaN	27/7/2003	16:00:00	NaN	Choque	NaN	NaN

63992 rows × 7 columns

Elaboración propia.

A partir de la tabla anterior se realizaron todas las transformaciones

La preparación de los datos para el análisis de predicción de accidentes de tránsito en la ciudad de Manizales se llevó a cabo en dos fases principales:

Preprocesamiento y reestructuración de los datos: Inicialmente, se recibieron datos de la Secretaría de Movilidad, donde cada accidente tenía asignado un número de croquis, así como una fecha y hora específica.

Estos datos debían transformarse en un formato adecuado para su uso con los algoritmos de predicción. En este caso, se crearon dos columnas llamadas 'ds' y 'y', donde 'ds' contiene las fechas de los accidentes y 'y' representa la cantidad de accidentes. Para ello, se registró cada accidente con sus respectivas columnas 'ds' y 'y', y luego se agruparon los datos por fecha específica. Dado que había días sin accidentes, se crearon registros con 'y'=0 para esas fechas, con el fin de obtener una serie de tiempo completa. Finalmente, se obtuvo un dataframe con un registro único para cada día, desde el 1 de enero de 2002 hasta la fecha final del conjunto de datos 31 de diciembre de 2019.

El formato resultante se puede observar en la figura 8. la cantidad de accidentes está representada en la columna ‘y’, la cual se compone de valores enteros positivos y la columna ‘ds’ representa una fecha específica de ocurrencia de los accidentes.

Figura 8

Imagen de la data transformada y lista para iniciar su uso en los algoritmos de machine learning

	ds	y
0	2002-01-01	6.0
1	2002-01-02	8.0
2	2002-01-03	7.0
3	2002-01-04	8.0
4	2002-01-05	5.0
...	...	...
6569	2019-12-27	1.0
6570	2019-12-28	8.0
6571	2019-12-29	4.0
6572	2019-12-30	13.0
6573	2019-12-31	8.0

6574 rows × 8 columns

Elaboración propia.

A partir de este dataframe, se generaron tablas adicionales agrupando los datos en ventanas de tiempo semanal, quincenal y mensual. Además, se segmentó la data semanal en tres categorías: semana completa, entre semana (lunes a viernes) y fines de semana.

Para la creación de conjuntos de datos en diferentes periodos de tiempo, se organizó la información en la escala temporal mencionada. Se creó una serie temporal de accidentes totales en toda la ciudad desde el 1 de enero de 2002 hasta el 31 de diciembre de 2019. Para las zonas detalladas más adelante en este documento, se estableció una serie temporal desde el 1 de enero de 2013 hasta el 31 de diciembre de 2019. A continuación, se describen los distintos periodos de tiempo de cada conjunto de datos:

Diario: se formó una serie temporal con un registro diario de accidentes, incluyendo la fecha y el número de accidentes ocurridos. Los días sin accidentes se registraron con un valor de cero.

Fines de semana: esta serie temporal incluye solo los accidentes ocurridos los sábados y domingos, almacenados en un periodo semanal con la fecha de inicio de la semana y la suma de accidentes en esos dos días.

Entre semana: esta serie temporal incluye solo los accidentes ocurridos de lunes a viernes, almacenados en un periodo semanal con la fecha de inicio de la semana y la suma de accidentes en esos cinco días.

Semanal: esta serie registra la fecha de inicio de cada semana y la suma total de accidentes ocurridos durante esa semana, incluidos todos los días.

Quincenal: esta serie temporal agrupa la información en periodos de dos semanas, registrando la fecha de inicio y la suma de accidentes en ese periodo.

Mensual: esta serie temporal agrupa la información en periodos de cuatro semanas, registrando la fecha de inicio y la suma de accidentes en ese periodo.

Además, para todos los periodos mencionados, se incluyó información resumida de días festivos, añadiendo una columna que indica la cantidad de días festivos en cada periodo de tiempo específico.

Todos los periodos que no tienen ocurrencia de accidentes se registraron con un valor de cero, con el fin de no dejar valores faltantes en la serie temporal.

Incorporación de variables exógenas:

En la segunda fase, se enriqueció el conjunto de datos con variables exógenas relevantes que podrían influir en la ocurrencia de accidentes. Estas variables incluyen días festivos, días con eventos que generan alteración del orden público, y otras características como el día de la semana y la zona geográfica donde ocurrió el accidente. La inclusión de estas variables exógenas permitió capturar patrones y tendencias adicionales que podrían mejorar la precisión y utilidad de las predicciones del modelo.

Se abordó la identificación de zonas geográficas mediante la creación de polígonos utilizando el sistema de coordenadas decimales. Los polígonos se definen como una serie de puntos expresados en coordenadas, los cuales están conectados entre sí a través de líneas rectas. Para llevar a cabo la creación de estos polígonos, se empleó la herramienta gratuita Google My Maps, la cual facilitó el proceso manual de construcción de las capas y extracción de las coordenadas correspondientes a cada uno de los polígonos creados.

Posteriormente, se procedió a cargar la información de cada polígono en el entorno de programación Python. Con el apoyo de la librería shapely.geometry fue posible identificar a qué zona geográfica definida pertenecía cada accidente registrado en el conjunto de datos. Para lograr esto, se utilizó el algoritmo de Ray Casting, que evalúa si un punto, representado por coordenadas geográficas, se encuentra dentro o fuera de un polígono específico, tomando en cuenta la información de las coordenadas conectadas que definen el área del polígono.

Una vez identificadas y clasificadas todas las incidencias de accidentes en función de las zonas geográficas establecidas, se procedió a cargar la información en una base de datos para su posterior análisis. Es preciso aclarar que para realizar este proceso se utilizaron solo los accidentes ocurridos a partir del 1 de Enero de 2013, puesto que sólo a partir de esa fecha se tiene la información de las coordenadas del accidente, y los polígonos fueron adaptados en varias oportunidades con el objetivo de que ningún accidente quedara por fuera del área de los polígonos. Este enfoque permitió realizar un estudio detallado de las características y distribución de los accidentes en función de las diferentes zonas geográficas, proporcionando información valiosa para la implementación de los algoritmos propuestos.

Esta metodología de preparación de datos permite generar una estructura adecuada y la inclusión de variables relevantes disponibles para desarrollar un modelo predictivo en el ámbito de la ingeniería computacional aplicada a la seguridad vial.

Inicialmente, se llevaron a cabo múltiples experimentos con diferentes tipos de modelos, los cuales fueron descartados debido a su bajo desempeño. Para mejorar los resultados, fue necesario transformar el dataset de diferentes maneras. Se experimentó creando un modelo global para cada día y hora del año, agrupando los datos a nivel de día y hora. También se desarrolló un modelo que consideró características como la zona de ocurrencia del accidente, el día de la semana, el mes y el año. Además, se consideró un modelo de múltiples salidas para la predicción de accidentes en diferentes zonas de la ciudad, mediante la creación de una cuadrícula de zonas. Sin embargo, los resultados para este tipo de modelo no fueron concluyentes.

Si bien en la base de datos aportada por la secretaría de movilidad del municipio de Manizales se encuentra la información relacionada con los accidentes tal como la descripción del tipo de accidente presentado, la característica de la vía, los vehículos involucrados, y las causas del accidente, dicha información no permite obtener variables que se puedan asociar a una ocurrencia futura por lo tanto para este estudio solo se tuvieron en cuenta la información de la fecha, hora, cantidad y ubicación georreferenciada de los accidentes de tránsito de la ciudad.

Zonas geográficas:

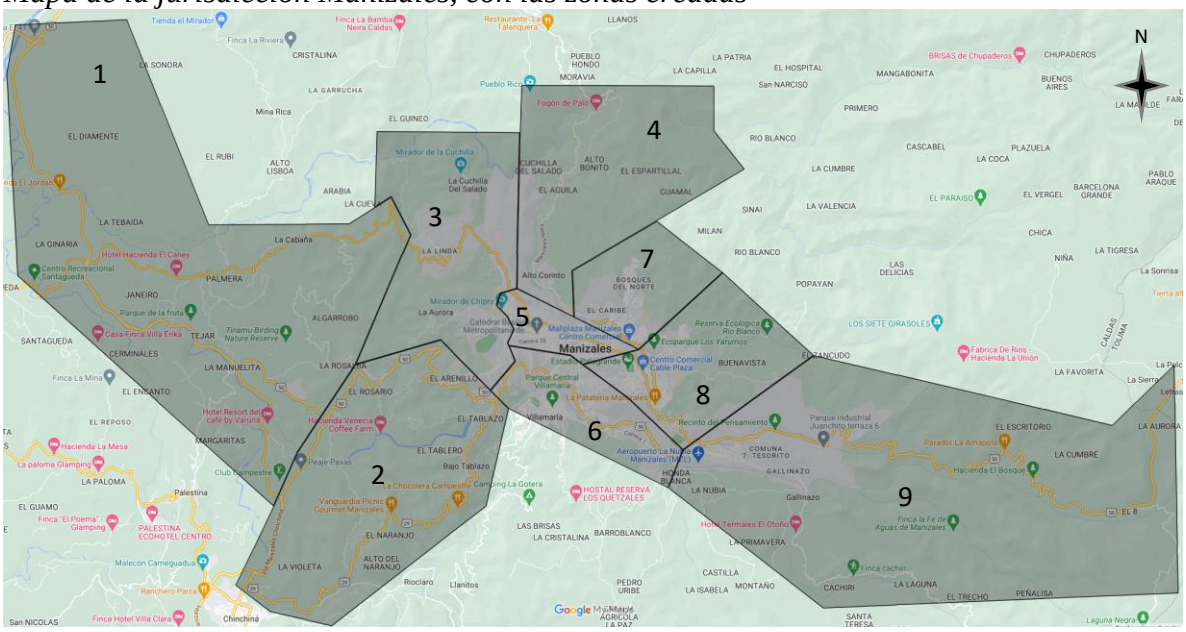
Debido a la vasta extensión geográfica que abarca la jurisdicción del municipio de Manizales, resulta imprescindible establecer subdivisiones geográficas más pequeñas, en las cuales se pueda identificar un patrón específico de accidentalidad en cada una de ellas. El propósito de este enfoque es poder realizar predicciones no solo a nivel general, sino también por zonas, lo que permitiría a las autoridades de tránsito tomar decisiones adaptadas a las características particulares de cada área.

Para lograr una mejor comprensión de la accidentalidad en cada zona, se ha desarrollado un mapa de calor que facilita la identificación visual de los puntos con mayor incidencia histórica de accidentes. Este mapa destaca, en color rojo, las áreas con los índices de accidentalidad más elevados, mientras que en azul claro se muestran aquellas zonas con menor cantidad de accidentes históricamente registrados. De esta manera, se logra un análisis efectivo y fácilmente comprensible de los puntos críticos de accidentes en el municipio.

A continuación, se presentarán las zonas creadas a partir de este enfoque, con el objetivo de proporcionar una herramienta para la planificación y toma de decisiones por parte de las autoridades de tránsito. Esta información permitirá enfocar los recursos y esfuerzos en las áreas de mayor necesidad, buscando mejorar la seguridad vial y reducir el número de accidentes en el municipio de Manizales. A continuación, vamos a ver las zonas creadas:

Figura 9

Mapa de la jurisdicción Manizales, con las zonas creadas



Elaboración propia.

Las zonas creadas en la figura 9 fueron construidas con la herramienta Google My Maps y pueden ser consultadas en el siguiente enlace: https://www.google.com/maps/d/u/0/edit?mid=155pOgqFghWZR8ppWge5fgtD3sMJ-gIJ8&ll=5.0712224656299805%2C-75.50139900000002&z=12

Las áreas han sido diseñadas considerando principalmente las diferentes vías de acceso a la ciudad, el centro y las zonas periféricas. Aunque algunas áreas son mucho más grandes que otras, la distribución de accidentes es inversamente proporcional al tamaño de la zona. Esto se debe a que la densidad poblacional de cada área influye en el aumento de desplazamientos, lo que a su vez incrementa la probabilidad de accidentes de tráfico.

Inicialmente, se crearon zonas más pequeñas, es decir, con mayor resolución; sin embargo, en muchos segmentos la ocurrencia de accidentes era casi nula en períodos cortos de tiempo, lo cual dificultaba la capacidad predictiva de los modelos. Por ello, se optó por diseñar áreas con menor resolución espacial y emplear gráficos de calor para identificar los puntos críticos de accidentalidad dentro de cada zona.

De esta manera, aunque la predicción se realice de forma general para una zona específica, también es posible visualizar los puntos con mayor probabilidad de accidentes dentro de dicha área. Así, las autoridades de tránsito pueden enfocar sus esfuerzos en zonas específicas en lugar de abarcar toda la zona completa.

En la siguiente tabla se muestra la cantidad de accidentes ocurridos en cada zona entre el año 2013 a 2019:

Tabla 5

Cantidad de accidentes por zona geográfica

Zona	Accidentes
Zona 1	309
Zona 2	703
Zona 3	478
Zona 4	148
Zona 5	9.023
Zona 6	2.331
Zona 7	1.953
Zona 8	3.600
Zona 9	1.470

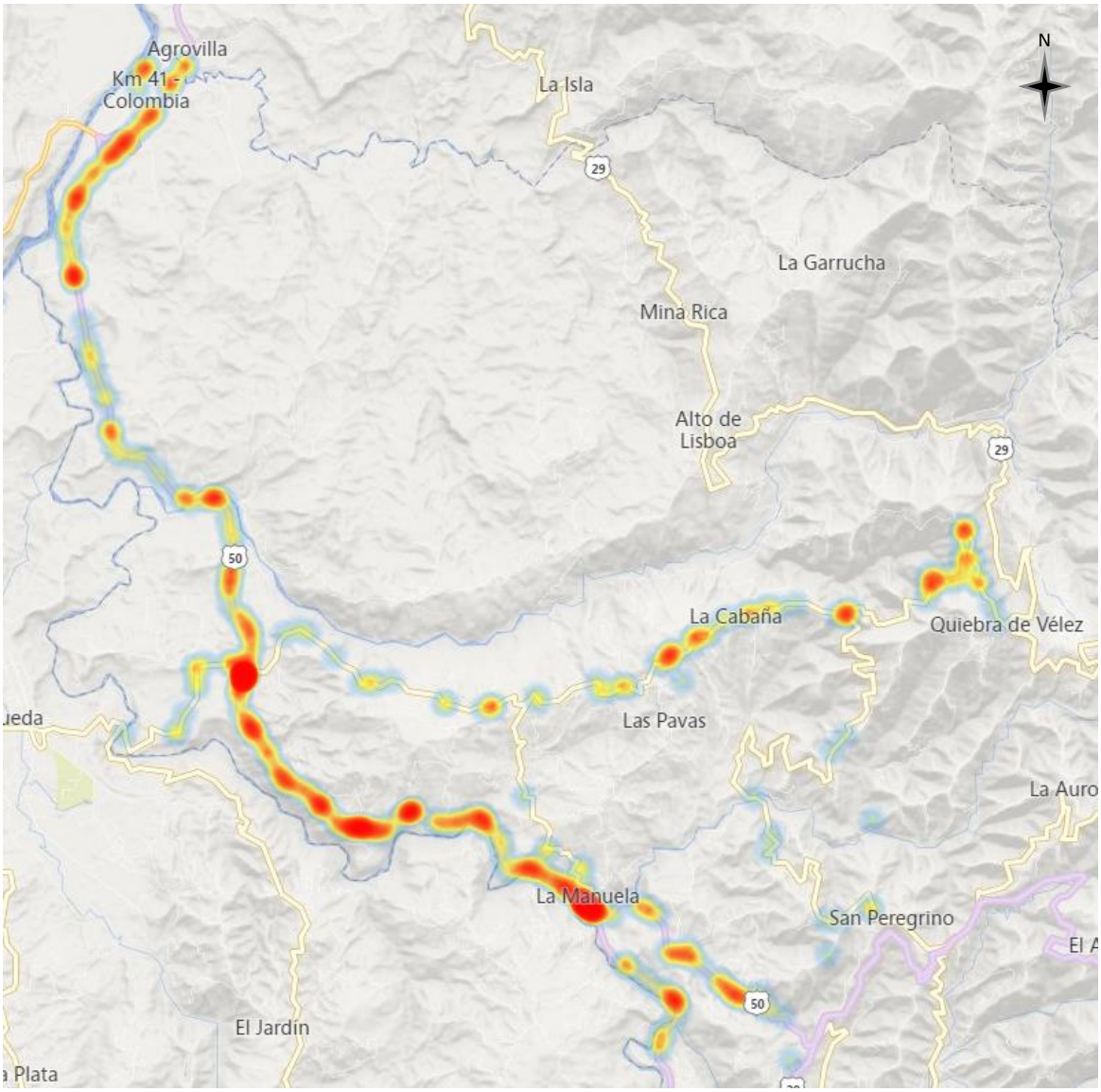
Elaboración propia

En la tabla mencionada, se puede apreciar que las Zonas 4, 1, 3 y 2 registran la menor cantidad de accidentes en orden respectivo. Por otro lado, las Zonas 5, 8 y 6 experimentan la mayor cantidad de accidentes, debido a que se encuentran en el área urbana más densa de la ciudad.

A continuación, se presentan cada una de las zonas usando mapas de calor.

Figura 10

Grafica de calor, según cantidad de accidentes Zona 1



Elaboración propia.

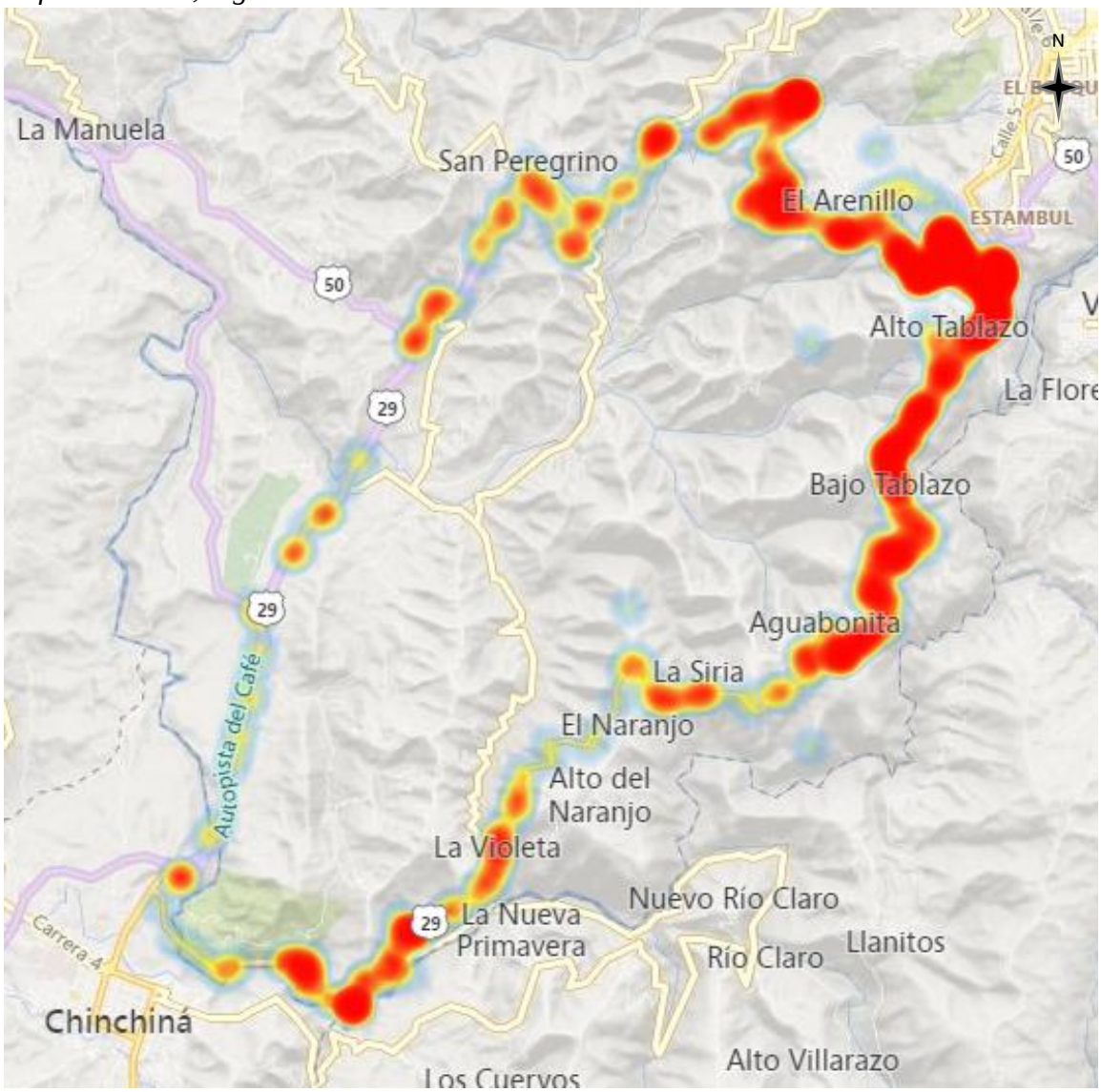
La Zona 1 se encuentra comprendida entre el peaje de la Quiebra de Vélez al oriente, la vereda El Kilómetro 41 al noroccidente, y los dos puntos de intersección entre la autopista del café Manizales – Chinchiná y la vereda la Manuela al sur.

Esta zona se caracteriza por ser completamente rural y estar compuesta principalmente por la carretera panamericana y vías terciarias, en general no se encuentra densamente poblado.

Se observa que el área de mayor accidentalidad se da principalmente en los puntos conocidos como La Manuela y Tres Puertas, también en el tramo de vía entre estos dos puntos, adicionalmente algunos puntos en sector de la Quiebra de Vélez y La Cabaña especialmente en curvas cerradas, otro punto es en cercanía a la vereda Kilómetro 41.

Figura 11

Grafica de calor, según cantidad de accidentes zona 2



Elaboración propia.

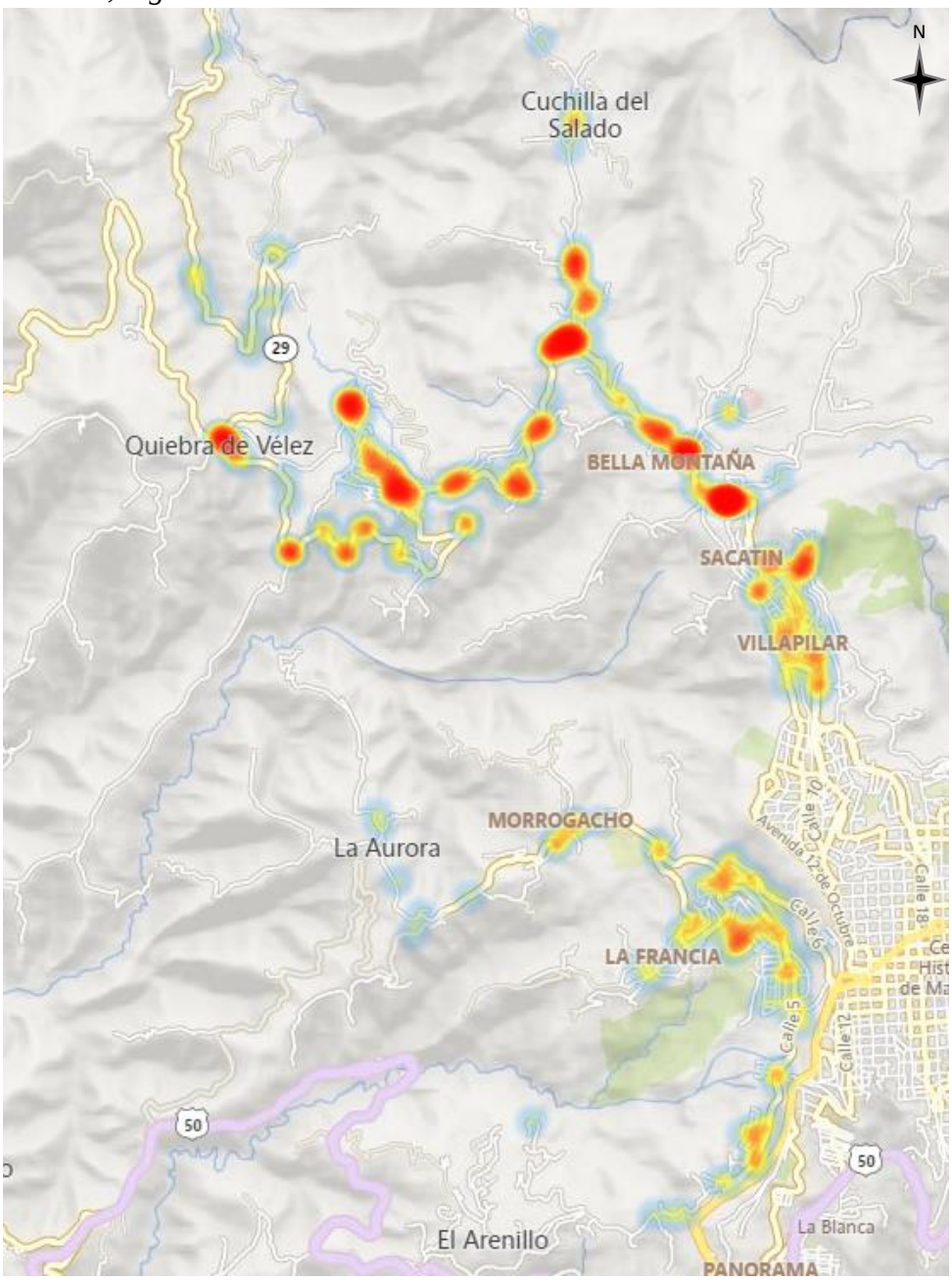
La Zona 2 se encuentra comprendida entre el punto conocido como La Estación Uribe al nororiente, y comprende las dos vías que comunican al municipio de Manizales con el Municipio de Chinchiná.

Esta vía se caracteriza por ser una vía de acceso a la ciudad la cual presenta un alto flujo vehicular y no tiene zonas densamente pobladas.

En esta zona se puede observar que el tramo de mayor accidentalidad está comprendido entre La Estación Uribe y la vereda San Peregrino principalmente en cercanías al primer punto, el segundo tramo con mayor accidentalidad está comprendido entra La Estación Uribe y la vereda Aguabonita, así como el tramo Chinchiná – La Violeta, podemos destacar que la vía de la izquierda es una autopista de doble carril y la de la derecha es una vía de un solo carril en cada sentido.

Figura 12

Grafica de calor, según cantidad de accidentes zona 3



Elaboración propia.

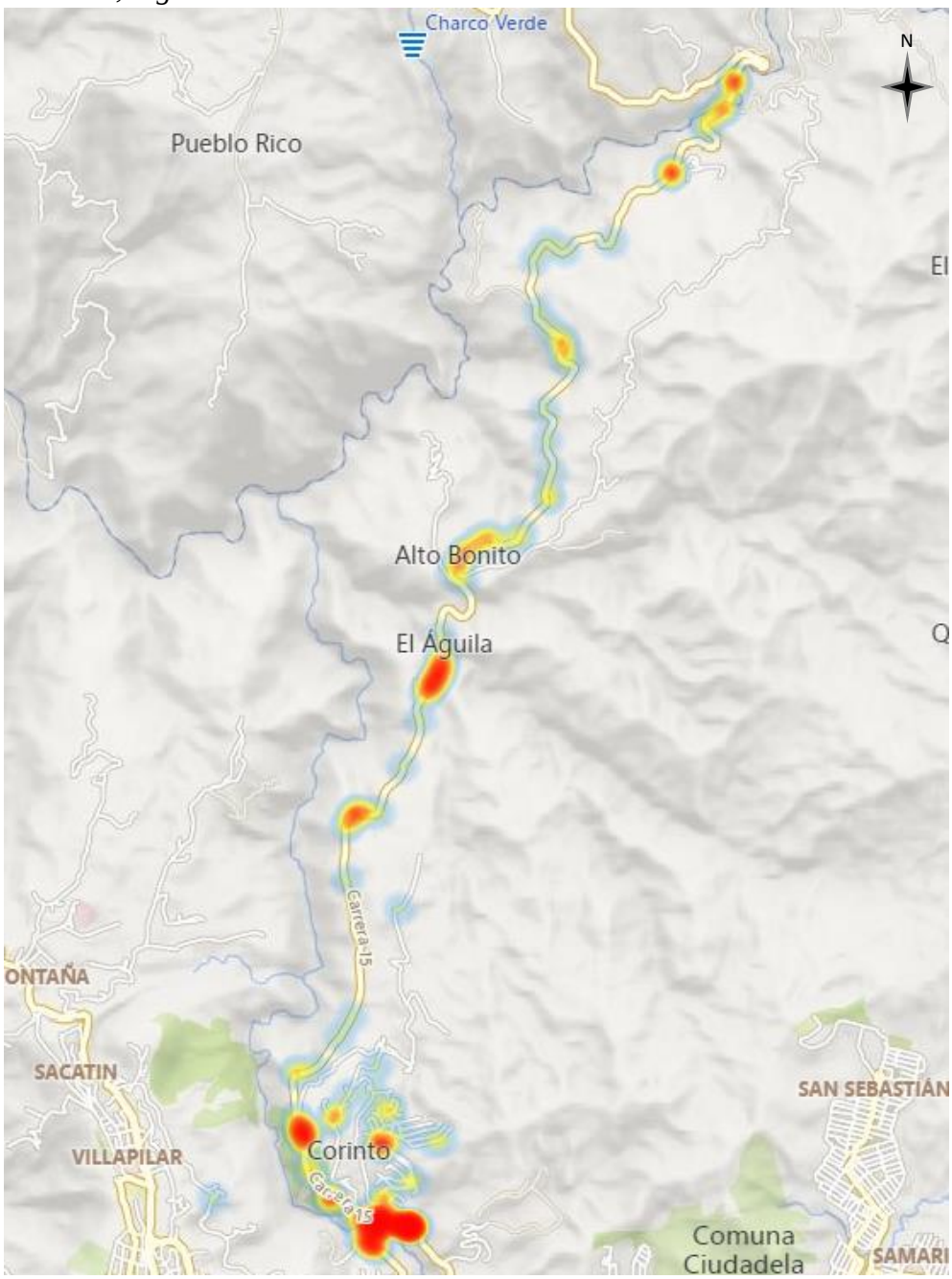
La Zona 3 comprende gran parte del llamado corregimiento panorama la cual es una zona de la ciudad que va desde el Hospital de Santa Sofía hasta la vereda la Aurora, y desde el round point del barrio Villa Pilar, Hasta la Quiebra de Vélez.

Esta Zona se caracteriza por tener parte urbana y rural, comprende algunas zonas densamente pobladas.

La zona de mayor accidentalidad se da principalmente en curvas cerradas entre el barrio Villapilar hasta el sector de la Quiebra de Vélez

Figura 13

Grafica de calor, según cantidad de accidentes zona 4



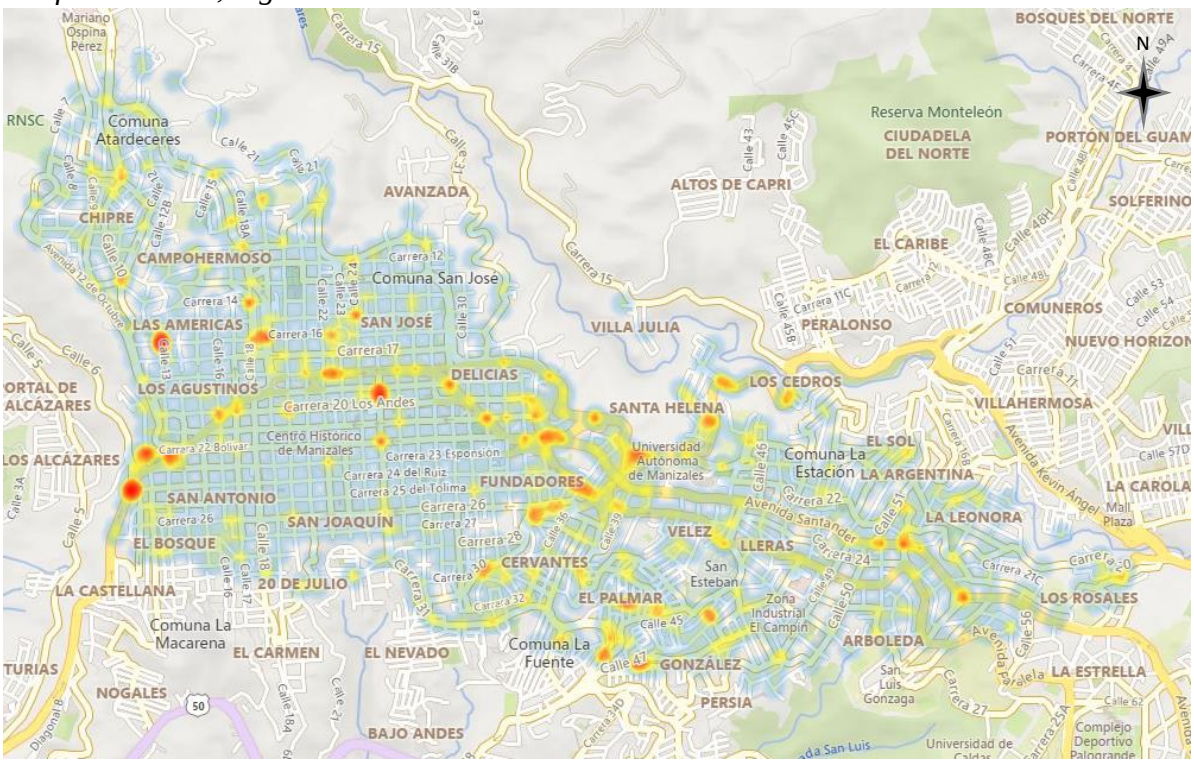
Elaboración propia.

La Zona 4 comprende principalmente la vía que comunica al municipio de Manizales con el Municipio de Neira, Caldas, Comprendida entre el sector de Los Cedros ubicado en la Avenida Kevin Ángel al sur hasta el rio Guacaica al norte.

Esta zona se caracteriza por ser una vía de acceso a la ciudad de Manizales desde el municipio de Neira, Caldas y sirve como conexión entre la ciudad y el norte del departamento, se compone principalmente de una vía sencilla y algunas zonas pobladas.

Figura 14

Grafica de calor, según cantidad de accidentes zona 5



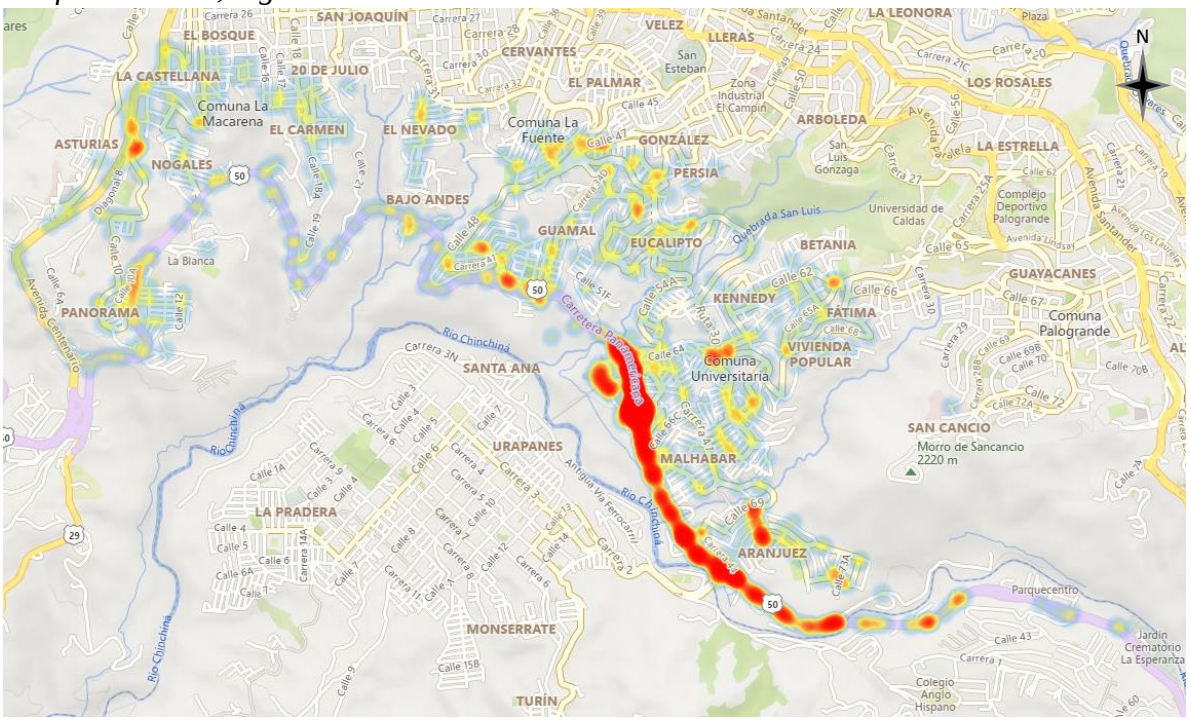
Elaboración propia.

La Zona 5 abarca principalmente el centro de Manizales, extendiéndose hacia el noroccidente desde el barrio Chipre hasta El Bosque. Hacia el sur, incluye el área entre El Bosque y el barrio Fanny González, atravesando La Arboleda. En el oriente, llega hasta el barrio Los Rosales, mientras que, hacia el norte, cubre el sector de Los Cedros y la comuna San José antes de retornar al punto de partida en el barrio Chipre.

Esta zona se caracteriza por ser totalmente urbana y densamente poblada, los puntos de mayor accidentalidad se dan en cercanías al Parque de Agua y la Avenida del Centro, así como diversos puntos a lo largo de toda la zona, lo cuales en su mayoría corresponden a cruces viales al mismo nivel de vía.

Figura 15

Grafica de calor, según cantidad de accidentes zona 6



Elaboración propia.

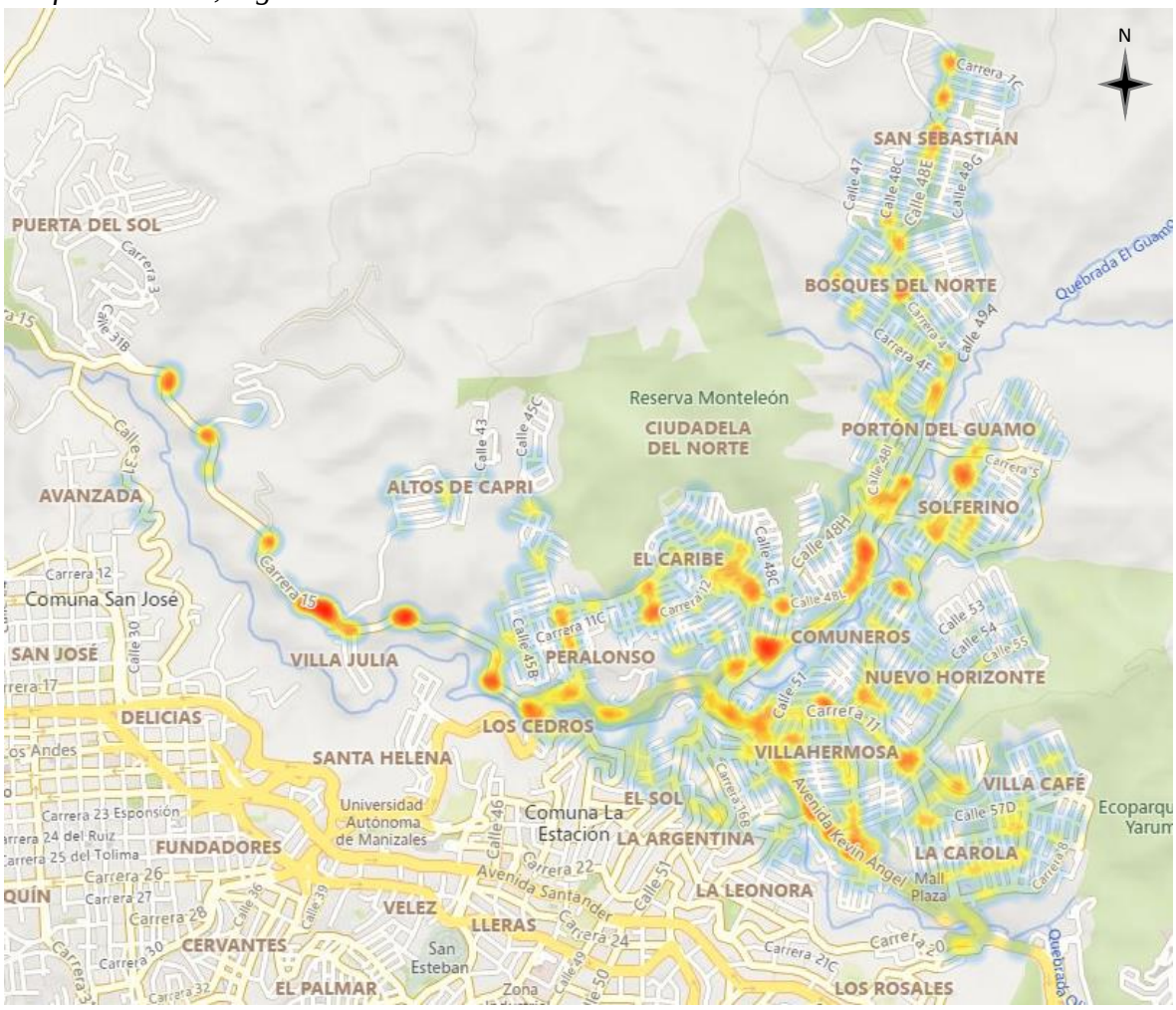
La Zona 6 comprende desde el sector conocido como La Estación Uribe al occidente, hasta el barrio Lusitania al sur oriente y limita al norte con la zona 5.

Esta zona se caracteriza por ser principalmente zona urbana y contiene en todo su trayecto la carretera panamericana cruza toda la ciudad.

En la imagen se puede apreciar una zona principal de accidentalidad la cual corresponde a la carretera panamericana, especialmente entre la zona de acceso al barrio al La Fuente y la entrada al municipio vecino de Villamaría.

Figura 16

Grafica de calor, según cantidad de accidentes zona 7



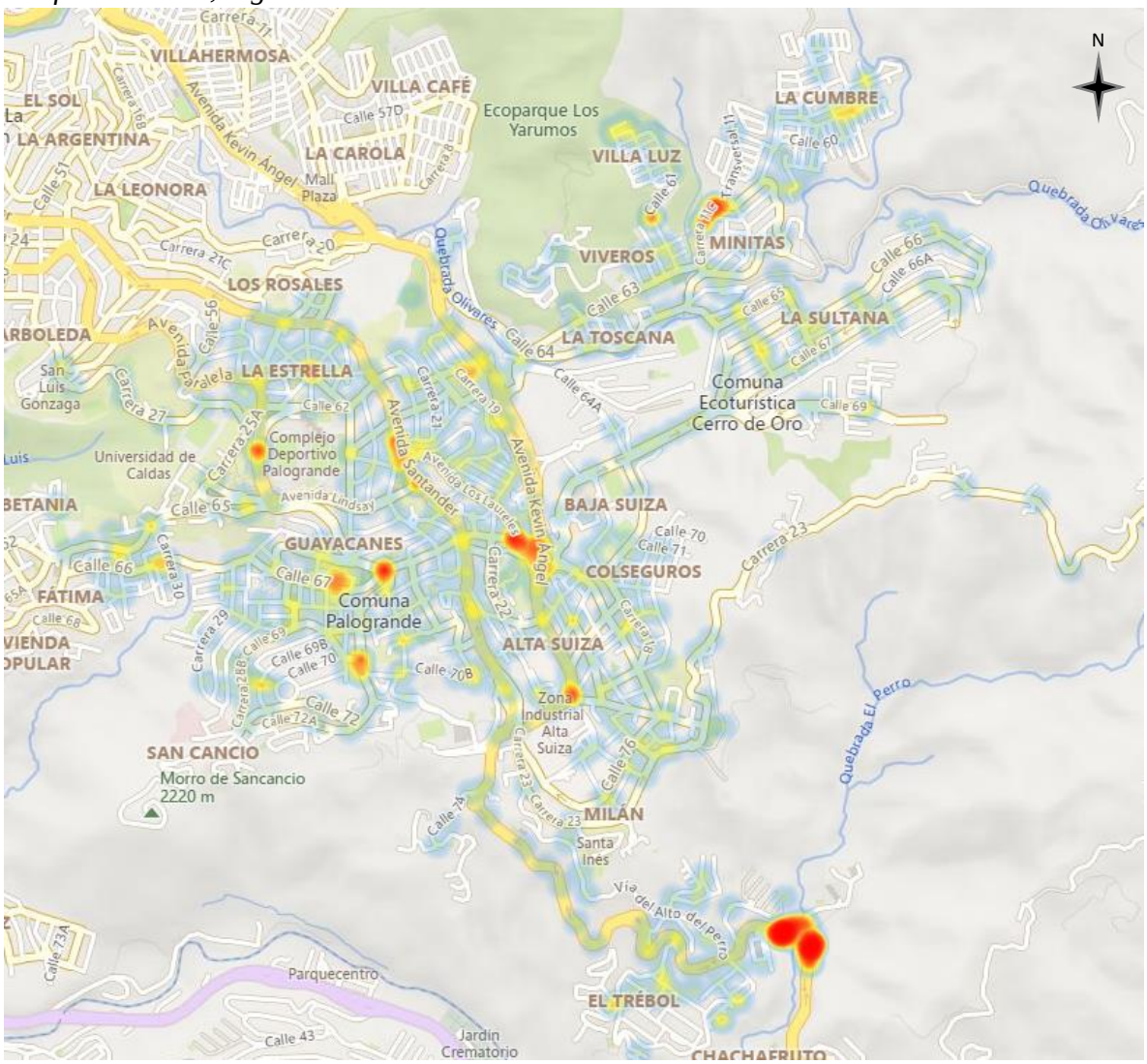
Elaboración propia.

La Zona 8 comprende principalmente la comuna norte de la ciudad va desde el barrio el Caribe al occidente pasando por toda la Avenida Kevin Ángel hasta el barrio la Carola al sur oriente y yendo al norte hasta el barrio San Sebastián.

Esta zona se caracteriza por ser Urbana y densamente poblada. Los puntos de accidentalidad en esta zona principalmente se dan en la avenida el Guamo, vía de acceso central a los barrios Bosques del Norte y San Sebastián, seguido de la Avenida Kevin Ángel, el Barrio Villa Julia y diferentes puntos los cuales son intersecciones viales a un mismo nivel en su mayoría.

Figura 17

Grafica de calor, según cantidad de accidentes zona 8



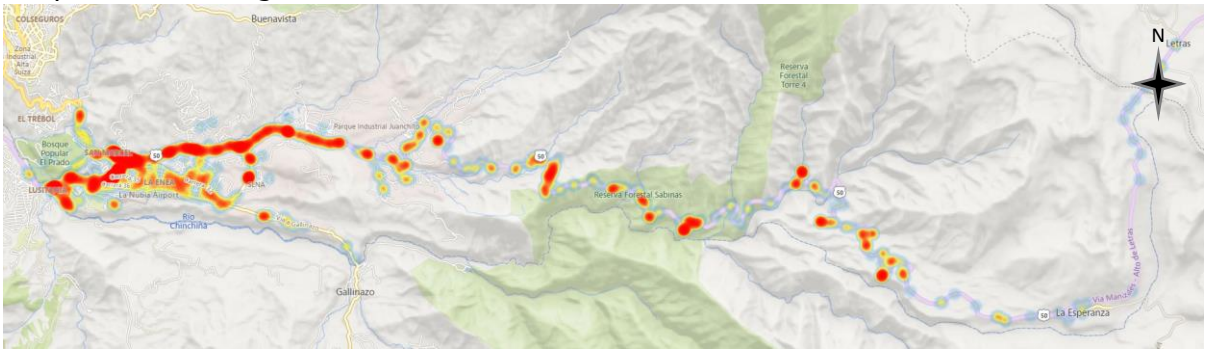
Elaboración propia.

La Zona 8 comprende la zona oriental de la ciudad va por el occidente desde el barrio La Estrella hasta Fátima, continuando al sur por el barrio Palermo hasta el sector de Expoferias al oriente y al norte va desde el sector de Milán pasando por la sultana hasta el barrio La Cumbre retornando a La Estrella.

Esta zona de la ciudad comprende zonas urbanas densamente pobladas en su mayoría. Los accidentes de esta zona se dan principalmente en el sector conocido como Expoferias, principalmente sobre el round point, seguido del sector de San Rafael y otros puntos de cruce ubicados a lo largo de esta zona.

Figura 18

Grafica de calor, según cantidad de accidentes zona 9



Elaboración propia.

La Zona 9 está comprendida entre el sector de San Marcel al occidente pando por el barrio la enea y Maltería hasta el límite con el municipio de Letras al oriente, Tolima.

Esta zona se caracteriza por ser mayormente rural y como una zona de acceso a la ciudad, sin embargo, contiene algunas zonas densamente pobladas.

De acuerdo a lo observado en la imagen la accidentalidad en esta zona se presenta principalmente en los sectores conocidos como San Marcel y la zona industrial Juanchito, y en la vía que comunica al municipio de Letras se observan varios puntos especialmente en las curvas con menor ángulo de apertura.

3.3 Descripción detallada del proceso de evaluación de técnicas de aprendizaje automático para la predicción de accidentes de tránsito en Manizales.

Para la predicción de accidentes de tránsito en la ciudad de Manizales, se implementaron diferentes técnicas de aprendizaje automático para entrenar varios modelos con diferentes arquitecturas y configuraciones. De esta manera, se evaluó el desempeño de los modelos construidos con cada técnica y se seleccionaron los mejores de cada una.

Posteriormente, se compararon los mejores modelos de cada técnica entre sí en términos de una métrica de error común, para mostrar el desempeño de todos los modelos dentro del experimento. Los resultados obtenidos para cada técnica se discuten en el capítulo 4 del informe, donde se muestran los valores de la métrica de error para cada modelo construido.

Este enfoque permitió identificar la técnica de aprendizaje automático más adecuada para la predicción de accidentes de tránsito en la ciudad de Manizales, y seleccionar el modelo con el mejor desempeño para la predicción de accidentes.

3.3.1 Implementación Facebook Prophet.

Prophet es una herramienta basada en un modelo aditivo que combina diversos componentes para abordar distintos aspectos de las series temporales. El modelo aditivo de Prophet consiste en una tendencia general, componentes de estacionalidad y efectos de días

festivos. La tendencia general puede capturar patrones de crecimiento no lineales y cambios en las tendencias, mientras que los componentes de estacionalidad permiten modelar patrones que se repiten a lo largo del tiempo, como ciclos diarios, semanales o anuales. Los efectos de días festivos son útiles para incorporar eventos especiales que afectan las observaciones en fechas específicas (Chaturvedi, Rajasekar, Natarajan, & McCullen, 2022).

La tendencia general se modela utilizando una función de crecimiento que puede ser lineal o logística. La función de crecimiento logística es especialmente útil cuando hay un límite superior en las observaciones. Los puntos de cambio de tendencia se identifican automáticamente en el modelo, lo que permite capturar cambios en la tendencia a lo largo del tiempo.

La estacionalidad se modela utilizando la descomposición de Fourier, que descompone las señales periódicas en una suma de funciones sinusoidales. Este enfoque permite capturar patrones de estacionalidad con diferentes formas y amplitudes. Además, Prophet permite modelar múltiples componentes de estacionalidad con diferentes periodos, como la estacionalidad diaria y la estacionalidad anual.

Los efectos de días festivos se modelan como componentes adicionales en el modelo aditivo, lo que permite capturar eventos específicos que no siguen patrones regulares de estacionalidad. Prophet permite agregar días festivos personalizados o incorporar días festivos de diferentes países.

Prophet utiliza inferencia Bayesiana para estimar los parámetros del modelo y realizar predicciones. En lugar de dar una única estimación puntual de los parámetros, la inferencia Bayesiana proporciona distribuciones de probabilidad completas, lo que permite cuantificar la incertidumbre en las predicciones. Para realizar la inferencia Bayesiana, Prophet utiliza un algoritmo de muestreo llamado Stan, que es eficiente y escalable para grandes conjuntos de datos y modelos complejos (Dommelen, 2021).

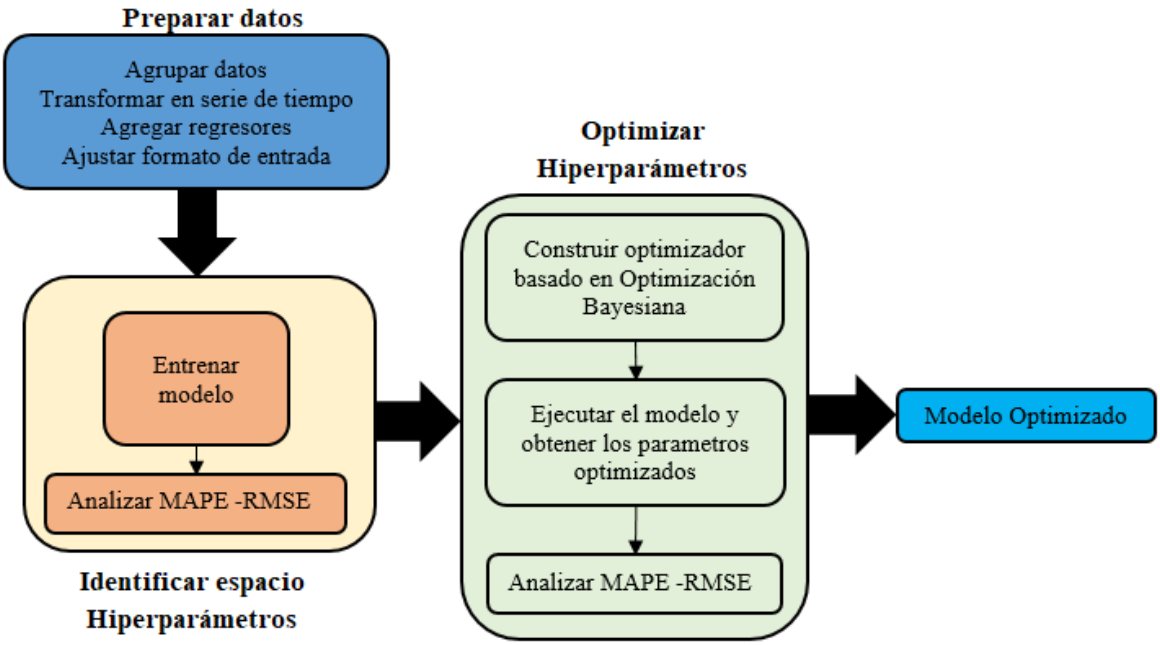
En particular, Prophet es conocido por su capacidad para abordar problemas de pronóstico de series de tiempo en diferentes dominios, como el comercio electrónico, la publicidad, la salud y muchos otros. Además, ofrece mediciones en tiempo real respecto a la importancia relativa de cada característica en la predicción, lo que permite una mejor comprensión de las tendencias y patrones en los datos.

La implementación de este algoritmo se lleva a cabo en cuatro fases, que comprenden la exploración y análisis de los datos históricos, la generación de modelos predictivos utilizando el algoritmo Prophet, la identificación de valores optimizados para los hiperparámetros y la evaluación de los modelos según la métrica de error cuadrático medio (RMSE, por sus siglas en inglés Root-mean-squared error).

En la primera fase, se exploran los datos históricos para identificar patrones y tendencias. En la segunda fase, se generan modelos predictivos utilizando el algoritmo Prophet y se ajustan a los datos históricos. En la tercera fase, se identifican valores óptimos para los hiperparámetros del modelo y se realizan pruebas para validar la precisión de las predicciones. En la figura 19 se pueden observar las fases de la implementación.

Figura 19

Etapas de implementación algoritmo Facebook Prophet



Elaboración propia.

Preparación de los datos: La preparación de los datos para el análisis de predicción de accidentes de tránsito en la ciudad de Manizales se llevó a cabo en dos fases principales:

La reestructuración y preprocesamiento de datos inicialmente, se obtuvieron datos de la Secretaría de Movilidad, en los que cada accidente contaba con un número de croquis asignado, así como una fecha y hora específicas. Estos datos necesitaban ser transformados en un formato apropiado para su utilización con el algoritmo de predicción Prophet. La primera etapa de dicha transformación se detalla en la sección de tratamiento de datos. Posteriormente, se incorporaron los regresores, es decir, las variables predictivas adicionales a la fecha del accidente, en el formato específico que requiere Prophet. Como requisito mínimo para el funcionamiento del algoritmo se requieren 2 columnas, “ds”: la cual contiene la información de las fechas de cada registro, y “y” la cual contiene la variable a predecir, en este caso la cantidad de accidentes, y adicionalmente se pueden agregar tantas columnas como regresores se tengan.

Identificación espacio de hiperparámetros: Prophet es un algoritmo de predicción de series temporales que se ejecuta utilizando una variedad de hiperparámetros ajustables. Estos hiperparámetros son fundamentales para mejorar la capacidad de predicción del algoritmo y optimizar el tiempo de ejecución del proceso de entrenamiento. La elección adecuada de estos hiperparámetros es esencial para garantizar un rendimiento óptimo en la resolución de problemas y en el análisis de datos.

Cabe destacar que el conjunto de hiperparámetros óptimos dependerá del problema específico que se desee resolver y de las características particulares de los datos con los que se esté trabajando. Por lo tanto, es crucial identificar y ajustar los hiperparámetros que

maximicen tanto la capacidad de predicción como la eficiencia en el tiempo de ejecución del entrenamiento.

En este caso se abordó un enfoque experimental el cual permite ejecutar diferentes combinaciones de hiperparámetros, para cada hiperparámetro se generó un rango de valores entre los cuales se puede hallar la combinación deseada para el rendimiento óptimo del algoritmo, este rango de valores se halló de manera experimental realizando diferentes pruebas las cuales tuvieron en cuenta la documentación del algoritmo para que los rangos escogidos de los hiperparámetros estuvieran dentro de los rangos permitidos por el algoritmo de Prophet. Como resultado se identificó el rango de valores de cada hiperparámetro, construyendo así el espacio de valores que se observa en la tabla 6.

Tabla 6

Espacio de hiperparámetros de Facebook Prophet

Hiperparámetro	Rango de Valores
changepoint_prior_scale	0.0001, 1.0
seasonality_mode	'additive', 'multiplicative'
seasonality_prior_scale	0.01, 50.0

Elaboración propia

Optimización de hiperparámetros: La optimización de hiperparámetros es una técnica muy útil en la construcción de modelos de aprendizaje automático, ya que permite ajustar los parámetros del modelo para maximizar el desempeño del mismo. En el contexto de la predicción de accidentes de tránsito en la ciudad de Manizales, la optimización de hiperparámetros permitió encontrar los valores óptimos de los parámetros que permiten maximizar la precisión del modelo de predicción.

En este caso, se utilizó el algoritmo de optimización bayesiana, el cual es una técnica de búsqueda que utiliza la información obtenida de evaluaciones anteriores para guiar la búsqueda hacia las regiones más prometedoras del espacio de hiperparámetros. En cada iteración del algoritmo, se evaluó el desempeño del modelo para un conjunto de valores de hiperparámetros utilizando la métrica de error MAPE. Con base en estas evaluaciones, el algoritmo ajustó los valores de los hiperparámetros para maximizar la precisión del modelo.

El espacio de hiperparámetros a explorar incluye 'changepoint_prior_scale', 'seasonality_mode' y 'seasonality_prior_scale'. *changepoint_prior_scale*: Este hiperparámetro controla la flexibilidad del modelo para adaptarse a los cambios en la tendencia. Un valor más alto permite que el modelo se ajuste más rápidamente a los cambios en la tendencia, mientras que un valor más bajo hace que el modelo sea más resistente a cambios repentinos. El valor predeterminado es 0.05 y se modifica para obtener mejores resultados. *seasonality_mode*: Este hiperparámetro determina cómo se modela la estacionalidad en la serie temporal. Hay dos opciones disponibles: 'aditiva' y 'multiplicativa'. En el modo 'aditiva', las componentes estacionales se suman a la tendencia; mientras que en el modo 'multiplicativa', las componentes estacionales se multiplican por la

tendencia. *seasonality_prior_scale*: Este hiperparámetro controla la cantidad de regularización aplicada a los componentes estacionales del modelo. Un valor más alto permite que el modelo ajuste los componentes estacionales más libremente, mientras que un valor más bajo aplica una regularización más fuerte y restringe la flexibilidad del modelo en el ajuste de la estacionalidad. El valor predeterminado es 10.

La función objetivo se define entrenando un modelo de Prophet con los hiperparámetros seleccionados y calculando el error porcentual absoluto medio (MAPE) en las predicciones del conjunto de prueba.

Se realiza la búsqueda de hiperparámetros óptimos mediante la función 'gp_minimize', que realiza 500 llamadas a la función objetivo con diferentes combinaciones de hiperparámetros. Al final, se reportan los mejores hiperparámetros encontrados y el MAPE correspondiente.

Esta implementación se enfoca en la optimización bayesiana, lo que permite una búsqueda más dirigida en el espacio de hiperparámetros y potencialmente una mayor eficiencia en la selección de los mejores modelos en base a la métrica de error elegida, en este caso el MAPE.

Una vez que se encontraron los valores óptimos de los hiperparámetros, se ajustó el modelo final utilizando Prophet y se realizó una evaluación exhaustiva del modelo para evaluar su precisión en la predicción de accidentes de tránsito en la ciudad de Manizales.

En resumen, la técnica de optimización bayesiana es una herramienta poderosa y eficiente para encontrar la combinación óptima de hiperparámetros en modelos de predicción basados en machine learning, y se utiliza en este caso para ajustar el modelo de predicción de accidentes de tránsito en la ciudad de Manizales y obtener la mejor precisión posible.

En la tabla 7, se presentan los parámetros obtenidos mediante la optimización bayesiana.

Tabla 7

Valores de hiperparámetros hallados en la etapa de optimización a través de Optimización Bayesiana

Periodicidad	changepoint_prior_scale	seasonality_mode	seasonality_prior_scale
Diario	0.983233	'additive'	42.998421
Semanal	0.02316	'multiplicative'	19.99905
Quincenal	0.02215	'multiplicative'	18.33367
Mensual	0.004152	'additive'	49.860838

Elaboración propia

A continuación, en función de los parámetros del mejor estimador obtenidos mediante la optimización bayesiana, se lleva a cabo el proceso de ejecución del algoritmo mediante la técnica de validación cruzada. El proceso se realiza en un bucle de validación cruzada, que itera sobre diferentes particiones de los datos para entrenar y probar el modelo con los hiperparámetros óptimos encontrados.

En cada iteración, se establecen los límites de los datos de entrenamiento y prueba, y se entrena un modelo Prophet con los hiperparámetros óptimos y un regresor adicional para los días festivos y otros eventos. Después de entrenar el modelo, se realizan predicciones en el conjunto de prueba y se calculan métricas de evaluación como el error porcentual absoluto medio (MAPE) y el error cuadrático medio (MSE.

Se almacenan los resultados de las predicciones y las métricas de evaluación en listas, y luego se convierten en un Dataframe que resume los resultados de la validación cruzada.

3.3.2 Implementación Light Gradient Boosting.

Light Gradient Boosting (LGB) es un algoritmo de aprendizaje automático basado en árboles de decisión que se utiliza para la clasificación y regresión en grandes conjuntos de datos. Este algoritmo se basa en la técnica de impulso, que combina varios modelos débiles para crear un modelo fuerte. LGB es conocido por su capacidad para trabajar con grandes conjuntos de datos y su eficiencia en términos de tiempo y recursos.

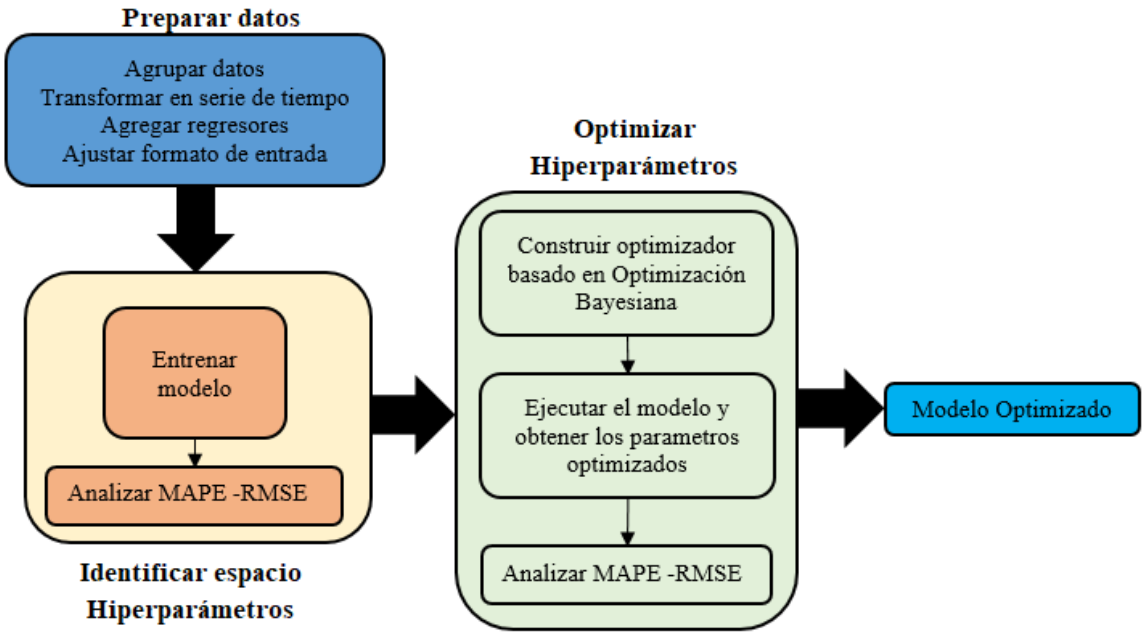
LGB utiliza un enfoque de construcción de árbol de decisión similar a XGBoost, pero utiliza una estrategia de crecimiento de hojas primero en lugar de crecimiento de profundidad primero. Esto permite una construcción de árbol más rápida y eficiente, lo que resulta en una velocidad de entrenamiento y predicción más rápida. Además, utiliza histogramas para discretizar características numéricas, lo que también contribuye a la velocidad del entrenamiento y predicción.

La implementación de LGB se realiza en varias etapas. En la primera etapa, se realiza una exploración de datos para identificar patrones y tendencias en los datos. En la segunda etapa, se crea un modelo base y se ajusta a los datos de entrenamiento. En la tercera etapa, se realiza una optimización de hiperparámetros para identificar los mejores valores para los hiperparámetros del modelo. Por último, se realiza una evaluación de la calidad del modelo mediante el cálculo de diferentes métricas de evaluación.

En general, LGB es un algoritmo de aprendizaje automático popular y efectivo que se utiliza para una amplia gama de aplicaciones de clasificación y regresión en grandes conjuntos de datos. Su eficiencia en términos de tiempo y recursos lo convierte en una herramienta valiosa.

Figura 20

Etapas de implementación algoritmo Light Gradient Boosting



Elaboración propia.

Preparación de los datos:

La preparación de los datos para el análisis de predicción de accidentes de tránsito en la ciudad de Manizales se llevó a cabo en dos fases principales:

La primera etapa de dicha transformación se detalla en la sección de tratamiento de datos. Posteriormente, se incorporaron los regresores, es decir, las variables predictivas adicionales a la fecha del accidente, en el formato específico que requiere el algoritmo Light Gradient Boosting, para el cual todos los regresores deben ser transformados en variables numéricas, puesto que el algoritmo sólo recibe parámetros numéricos.

Identificación del espacio de hiperparámetros:

Tabla 8
Espacio de hiperparámetros de Light Gradient Boosting

Hiperparámetro	Rango de Valores
num_leaves	10, 100
learning_rate	0.001, 0.1
feature_fraction	0.1, 1.0
bagging_fraction	0.1, 1.0
bagging_freq	1, 10

Elaboración propia.

El espacio de hiperparámetros se define a través de la siguiente configuración, basada en el código proporcionado:

num_leaves: Valores enteros entre 10 y 100, que representan el número de hojas en un árbol de decisión.
learning_rate: Valores reales entre 0.001 y 0.1, que representan la tasa de aprendizaje del modelo.
feature_fraction: Valores reales entre 0.1 y 1.0, que representan la fracción de características a utilizar en cada iteración.
bagging_fraction: Valores reales entre 0.1 y 1.0, que representan la fracción de datos a utilizar para el entrenamiento por cada árbol de decisión.
bagging_freq: Valores enteros entre 1 y 10, que representan la frecuencia con la que se realiza el muestreo de los datos.

El enfoque experimental con rangos para los hiperparámetros permite que la optimización bayesiana explore diferentes combinaciones de valores en busca de la mejor configuración de hiperparámetros que minimice el error de predicción en la función de pérdida. De esta manera, se logra una optimización más eficiente y efectiva del modelo LightGBM en comparación con la búsqueda exhaustiva o aleatoria.

Optimización de hiperparámetros: En este proceso, se lleva a cabo una optimización de hiperparámetros para un modelo LightGBM utilizando Optuna. Primero, se divide el conjunto de datos en entrenamiento y prueba, y se preparan los datos para el modelo LightGBM, separando las variables de entrada y las variables objetivo.

A continuación, se define una función objetivo que se desea minimizar, en este caso, el error porcentual absoluto medio (MAPE). Dentro de esta función, se especifican los hiperparámetros a optimizar, como el número de hojas, la tasa de aprendizaje, la fracción de características y la fracción y frecuencia de bagging. Luego, se crea un conjunto de datos LightGBM para el entrenamiento y otro para la evaluación.

Se entrena el modelo LightGBM utilizando los hiperparámetros especificados y un número de iteraciones de boosting. Se emplea un enfoque de detención temprana, deteniendo el entrenamiento si el error de validación no mejora significativamente durante un número específico de iteraciones. Después de entrenar el modelo, se realizan predicciones en el conjunto de prueba y se calcula el MAPE.

Para ejecutar la optimización de hiperparámetros, se crea un estudio de Optuna con la dirección de minimización y se optimiza la función objetivo mediante un número determinado de ensayos. Finalmente, se muestran los mejores hiperparámetros encontrados durante la optimización.

Se utilizó Optuna en este ejercicio debido a sus ventajas en la optimización de hiperparámetros para modelos de aprendizaje automático. Optuna es una biblioteca de Python diseñada específicamente para optimizar hiperparámetros de manera eficiente y flexible. A continuación, se detallan algunas de las razones por las que se eligió Optuna para este ejercicio:

Flexibilidad: Optuna permite definir fácilmente el espacio de búsqueda de hiperparámetros y las restricciones específicas del problema. Además, es compatible con una amplia variedad de algoritmos de aprendizaje automático, lo que facilita su integración con diferentes modelos, como LightGBM en este caso.
Eficiencia: Optuna utiliza algoritmos de optimización bayesiana y de árboles de búsqueda para explorar de manera eficiente el espacio de hiperparámetros. Estos algoritmos permiten a Optuna encontrar hiperparámetros óptimos con menos ensayos en comparación con otros métodos de búsqueda, como la búsqueda en cuadrícula o la búsqueda aleatoria.
Detección temprana de pruebas de baja calidad: Optuna ofrece la funcionalidad de detección temprana (pruning), que permite interrumpir pruebas de baja calidad antes de que finalicen, lo que ahorra tiempo y recursos computacionales.
Registro y visualización: Optuna incluye funciones de registro y visualización que facilitan el seguimiento del progreso de la optimización y la comprensión de los resultados.

En la tabla 9, se presentan los parámetros obtenidos mediante la optimización bayesiana.

Tabla 9
Valores de hiperparámetros hallados en la etapa de optimización a través de Optuna

Periodicidad	num_leaves	learning_rate	feature_fraction	bagging_fraction	bagging_freq
Diario	89	0.092452	0.881995	0.1698	3
Semanal	59	0.04226	0.919239	0.545604	5
Quincenal	45	0.041771	0.97136	0.16795	10
Mensual	62	0.099021	0.666184	0.829707	8

Elaboración propia

3.3.3 Implementación redes neuronales artificiales recurrentes - LSTM.

Las redes neuronales artificiales recurrentes (LSTM) son una técnica de aprendizaje profundo que se utiliza para el análisis y la predicción de datos de series de tiempo. A diferencia de otros modelos de predicción, como LGB o Prophet, LSTM utiliza una arquitectura de red neuronal que se enfoca en la memoria de los datos de entrada anteriores para hacer predicciones futuras. Esta arquitectura les permite a las LSTM modelar secuencias de datos complejas y capturar patrones a largo plazo en los datos de series de tiempo.

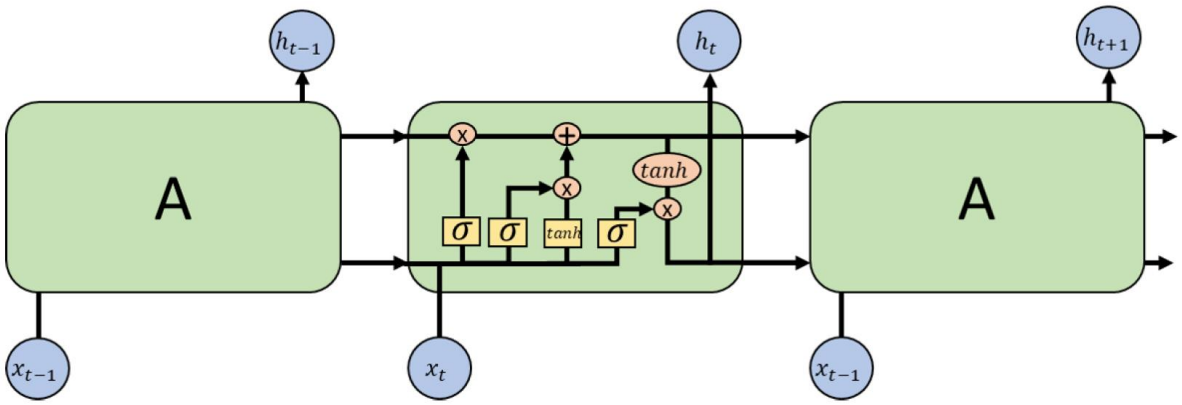
En una red LSTM, cada unidad de memoria está compuesta de una celda de memoria y tres puertas: la puerta de entrada, la puerta de salida y la puerta de olvido. Cada una de estas puertas tiene un propósito específico en la propagación de la información a través de la red. La puerta de entrada controla la cantidad de información nueva que se agregará a la celda de memoria, mientras que la puerta de olvido controla la cantidad de información que se mantendrá en la celda de memoria. Finalmente, la puerta de salida controla la cantidad de información que se propagará a la siguiente unidad de memoria o salida.

El proceso de entrenamiento de una red LSTM implica alimentar la red con secuencias de datos de entrada y luego ajustar los pesos de las conexiones neuronales para minimizar el error en la predicción de los datos de salida. A medida que se entrena la red, la celda de memoria y las puertas aprenden a modelar los patrones en los datos de entrada y predecir con precisión los valores futuros.

Una vez que se entrena la red, se puede utilizar para predecir futuros valores de una serie de tiempo. Al igual que con otros modelos de predicción de series de tiempo, se pueden utilizar métricas de evaluación, como RMSE o MAPE, para evaluar la precisión de las predicciones de la red LSTM. Las redes neuronales artificiales recurrentes (LSTM) son una técnica de aprendizaje profundo que se enfoca en modelar secuencias de datos de series de tiempo y capturar patrones a largo plazo. Su arquitectura única de memoria y puertas les permite modelar secuencias complejas de datos y hacer predicciones precisas.

Figura 21

Estructura de una red neuronal LSTM

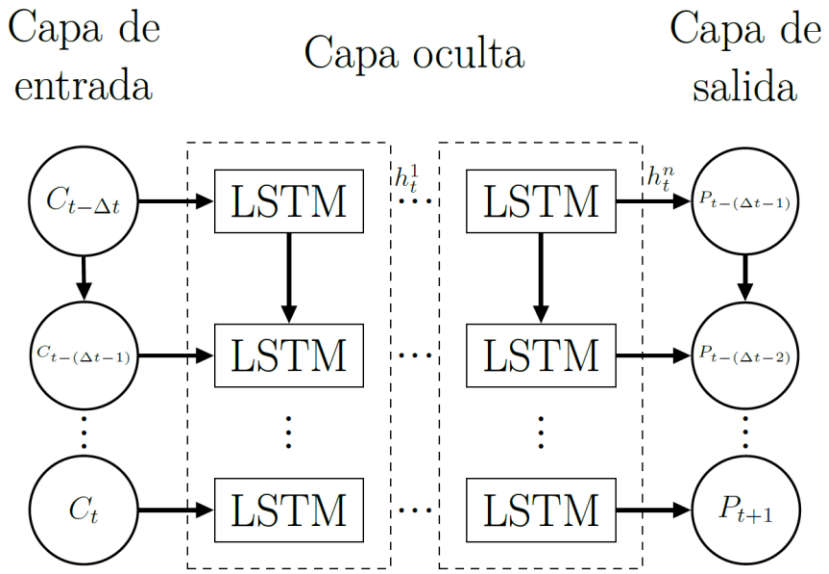


Adaptado de (Ferro, Celis Mayorga, & Casallas García, 2020)

En este estudio se planteó un modelo LSTM particular para predecir la cantidad de accidentes de tránsito.

Figura 22

Arquitectura general de una red neuronal de tipo LSTM



Elaboración propia.

Las siguientes expresiones determinan el vector de valores de salida h_t^2 y el vector de predicciones P_t, a partir del vector de valores de entrada C_t, que representa la cantidad de accidentes. En este caso, h_t^2 se refiere a los valores de salida intermedios, mientras que P_t, representa las predicciones finales basadas en la cantidad de accidentes (C_t).

$$h_t^1 = \sigma(Ct)$$

$$h_t^n = \sigma(h_t^{n-1}, h_{t-1}^n)$$

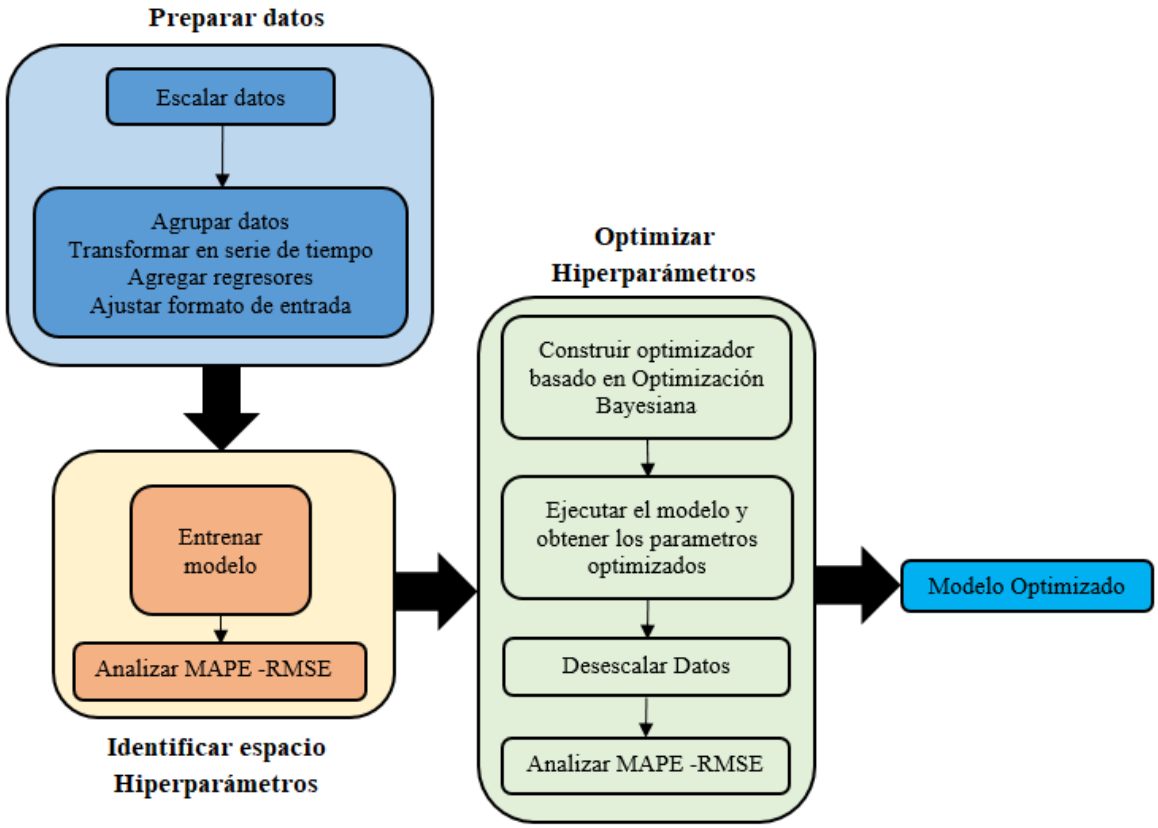
$$P_t = \Theta(W^{n,n+1} h_t^n + \alpha^{n+1})$$

Donde σ corresponde a la función de activación sigmoidal en la ANN LSTM, h_t^n denota el vector de valores de salida para la capa oculta n en el tiempo t. Θ corresponde a la función de activación tangencial, W denota la matriz de pesos de la capa oculta a la capa de salida y α describe el vector de sesgo a la capa de salida.

Chollet (2021) explica que los modelos de redes neuronales, como las redes LSTM, a menudo se caracterizan por un gran número de hiperparámetros que influyen en su arquitectura y proceso de aprendizaje. La optimización de estos hiperparámetros puede ser desafiante, ya que su ajuste adecuado es crucial para el rendimiento del modelo. Además, la complejidad de la estructura de la red está estrechamente relacionada con el tiempo necesario para entrenar el modelo, lo que puede aumentar significativamente a medida que la red se vuelve más profunda.

Figura 23

Etapas de implementación algoritmo LSTM



Elaboración propia.

Identificación espacio de hiperparámetros: Debido a la cantidad de hiperparámetros y configuraciones que requieren este tipo de modelos, el espacio de valores de estos hiperparámetros se halló de manera experimental entrenando un modelo con una arquitectura básica, variando los valores en diferentes magnitudes y analizando el impacto en el error obtenido y el tiempo de entrenamiento.

Tabla 10

Espacio de valores para los hiperparámetros de LSTM

Hiperparámetro	Valores
Unidades LSTM	32, 64, 128, 256, 512
Función de activación capas ocultas LSTM	tangencial, relu
Tasa de dropout	0, 0.01, 0.1, 0.2, 0.3, 0.5
Training Epochs	100
Tamaño de batch	10
Algoritmo de optimización	Adam, Adamax

Tasa de aprendizaje	0.01, 0.001, 0.0001
Patience	25

Elaboración propia

Según la Tabla 10 se seleccionaron las funciones de activación tangencial (LeCun, Bottou, Bengio, & Haffner, 1998) y unidad lineal rectificada (ReLU, por sus siglas en inglés). De igual forma se determinó que los algoritmos de optimización del modelo Adam y Adamax (Kingma & Ba, 2014) presentaron mejores resultados en general durante esta etapa, por lo tanto, se incluyeron dentro del espacio de hiperparámetros.

Optimización de hiperparámetros: En primer lugar, se realiza el preprocesamiento de los datos, escalando y creando conjuntos de entrenamiento y prueba. Luego, se crea una función para generar secuencias de datos de entrada y salida que se utilizarán en el modelo LSTM.

Para la optimización de hiperparámetros, se utiliza la biblioteca Optuna, que permite definir un espacio de búsqueda para explorar diversas combinaciones de hiperparámetros. Entre los hiperparámetros a optimizar se encuentran las unidades LSTM en dos capas, la función de activación, la tasa de abandono (dropout), el optimizador y la tasa de aprendizaje. Además, se implementa el mecanismo de detención temprana (EarlyStopping) basado en un parámetro de paciencia, evitando así el sobreajuste.

Figura 24

Arquitectura del modelo LSTM base

Elaboración propia.

La función objetivo toma los hiperparámetros del espacio de búsqueda y crea un modelo LSTM de varias capas utilizando la biblioteca Keras. Esta arquitectura incluye capas LSTM, capas de abandono (dropout) y una capa densa para la salida. El modelo se compila y ajusta utilizando los hiperparámetros proporcionados, junto con la detención temprana basada en el parámetro de paciencia, con el fin de evitar entrenamientos innecesariamente largos.

Una vez entrenado el modelo, se realizan predicciones en el conjunto de datos de prueba. Posteriormente, se desescalan los datos para obtener las predicciones y los valores reales en su escala original, lo cual es crucial para calcular el error porcentual absoluto medio (MAPE), que se utiliza como métrica de rendimiento del modelo.

La optimización se lleva a cabo utilizando el algoritmo Tree-structured Parzen Estimator (TPE) de Optuna, estableciendo un número máximo de evaluaciones y un tiempo máximo para la optimización. Al final del proceso, se obtienen los mejores hiperparámetros encontrados, que se pueden utilizar para construir un modelo LSTM optimizado y mejorar la precisión en la predicción de la variable objetivo. Se utilizó la optimización bayesiana para este ejercicio debido a sus ventajas en la búsqueda de hiperparámetros óptimos en comparación con otros métodos de optimización, como la búsqueda en cuadrícula o la

búsqueda aleatoria. En la tabla 11, se presentan los parámetros obtenidos mediante la optimización bayesiana.

Tabla 11

Valores de hiperparámetros hallados en la etapa de optimización a través de Optuna

Periodicidad	units_1	units_2	activation	dropout	optimizer	learning_rate
Diario	64	64	relu	0.1	Adam	1.001
Semanal	256	512	relu	0.5	Adamax	0.01
Quincenal	64	32	relu	0.5	Adamax	0.001
Mensual	512	512	relu	0.1	Adam	0.001

Elaboración propia

4. RESULTADOS.

La evaluación de las distintas implementaciones se llevó a cabo utilizando un conjunto de datos de prueba, que no se emplearon durante el entrenamiento. Estos datos representan los valores observados de la cantidad de accidentes en un periodo de 180 días perteneciente a la segunda mitad del año 2019.

En este capítulo, se presentan los resultados obtenidos para cada implementación en función del conjunto de datos de prueba, señalando el valor de la métrica de evaluación para cada modelo propuesto. Se eligieron dos métricas para esta evaluación: el RMSE y el MAPE. El RMSE mide la diferencia entre un valor observado y un valor predicho por un modelo, mientras que el MAPE evalúa el error porcentual absoluto medio.

La ecuación 3 describe la fórmula utilizada para calcular el RMSE:

Ecuación 3
Fórmula del RMSE

$$RMSE = \sqrt{\frac{\sum_{i=1}^{N}(y_i - \hat{y}_i)^2}{N}}$$

La ecuación 4 describe la fórmula utilizada para calcular el MAPE:

Ecuación 4
Fórmula del MAPE

$$MAPE = \frac{1}{N}\sum_{i=1}^{N}\frac{|y_i - \hat{y}_i|}{y_i}$$

Donde ‘y’ es el valor observado, ‘ŷ’ es el valor predicho y ‘N’ es el tamaño de la muestra.

Se optó por emplear estas métricas por diversas razones. En primer lugar, el RMSE permite calcular el error en la misma unidad de medida de los valores observados, lo que facilita la comparación con otros modelos y la interpretación del error en términos del problema analizado, en este caso, los accidentes de tránsito. Además, el RMSE otorga mayor peso al error, ya que la diferencia entre cada valor observado y el valor predicho se eleva al cuadrado antes de ponderar dichas diferencias.

Adicional a estas dos métricas se creó el RMSE porcentual (RMSE %) dado que si bien se presenta el RMSE su interpretación depende del valor promedio de los datos para medir la precisión del algoritmo, de esta forma se dividió el RMSE sobre el valor promedio de los datos predichos y se creó el RMSE % para interpretar los resultados independientemente de la escala de los datos predichos.

El MAPE (Mean Absolute Percentage Error) es una métrica de error ampliamente utilizada en la predicción de series temporales y en la evaluación de la precisión de los

modelos de pronóstico, presenta algunas limitaciones que pueden afectar su aplicabilidad debido a su sensibilidad a valores cercanos a cero, cuando los valores reales observados están cerca de cero, incluso pequeños errores absolutos pueden generar porcentajes de error muy grandes, lo que puede llevar a un MAPE inflado. Esta situación puede distorsionar la interpretación de la precisión del modelo (Hyndman & Koehler, 2006). Dado lo anterior se realizaron las interpretaciones del modelo basado en el RMSE % por causa de que varios de los dataset analizados por los algoritmos especialmente en zonas de menor cantidad de accidentes presentaban valores en cero, lo cual provocó un incremento significativo en el valor del MAPE.

Las predicciones se realizaron utilizando la estrategia de validación progresiva, que consiste en emplear los datos reales de accidentes disponibles a medida que avanza la predicción en el tiempo. Por lo tanto, para predecir un periodo especifico de accidentes, se utilizan los valores pasados reales, y luego, el valor real de la cantidad de accidentes para el momento predicho se conoce y se emplea para predecir el siguiente periodo.

4.1 Resultados Facebook Prophet.

En la tabla 12 se muestran los valores del error RMSE y MAPE para cada zona en cada tiempo modelado con el modelo de Prophet.

Tabla 12
Medición RMSE y MAPE de cada zona para los modelos basados en Facebook Prophet

Zona	Periodicidad	RMSE	PROMEDIO	RMSE %	MAPE
Total General	Diario	3.22	7.53	43%	76%
	Fin de Semana	7.88	53.25	15%	12%
	Entre Semana	7.21	38.31	19%	17%
	Semanal	4.20	14.94	28%	22%
	Quincenal	8.81	106.50	8%	6%
	Mensual	35.59	195.75	18%	17%
Zona 1	Diario	0.57	0.45	127%	256%
	Fin de Semana	0.85	1.10	77%	250%
	Entre Semana	1.33	1.63	82%	210%
	Semanal	1.73	2.69	64%	61%
	Quincenal	2.73	5.38	51%	39%
	Mensual	8.50	10.67	80%	64%
Zona 2	Diario	0.68	0.55	124%	304%
	Fin de Semana	0.79	1.25	63%	195%
	Entre Semana	1.59	2.17	73%	470%
	Semanal	2.00	3.39	59%	303%
	Quincenal	2.87	6.77	42%	34%

	Mensual	7.26	13.00	56%	42%
Zona 3	Diario	0.37	0.27	137%	149%
	Fin de Semana	0.47	0.48	98%	231%
	Entre Semana	1.02	0.97	104%	533%
	Semanal	1.10	1.37	80%	338%
	Quincenal	1.59	2.70	59%	280%
	Mensual	1.45	5.50	26%	24%
Zona 4	Diario	0.23	0.16	142%	73%
	Fin de Semana	0.27	0.20	135%	125%
	Entre Semana	0.53	0.42	127%	260%
	Semanal	0.53	0.52	102%	249%
	Quincenal	0.78	0.95	81%	279%
	Mensual	1.42	2.02	71%	358%
Zona 5	Diario	1.80	2.96	61%	237%
	Fin de Semana	2.33	5.08	46%	53%
	Entre Semana	3.90	15.62	25%	19%
	Semanal	4.64	20.69	22%	17%
	Quincenal	6.34	41.38	15%	13%
	Mensual	14.66	81.67	18%	16%
Zona 6	Diario	1.00	1.01	99%	369%
	Fin de Semana	1.41	1.68	84%	438%
	Entre Semana	2.48	5.12	48%	32%
	Semanal	2.70	6.77	40%	34%
	Quincenal	4.33	13.54	32%	27%
	Mensual	7.13	28.00	25%	15%
Zona 7	Diario	0.83	0.70	118%	354%
	Fin de Semana	1.30	1.63	80%	239%
	Entre Semana	1.75	2.93	60%	302%
	Semanal	2.54	4.54	56%	66%
	Quincenal	3.25	9.08	36%	36%
	Mensual	3.48	18.00	19%	17%
Zona 8	Diario	1.05	1.21	87%	385%
	Fin de Semana	1.29	1.89	68%	113%
	Entre Semana	2.82	6.35	44%	52%
	Semanal	3.33	8.23	40%	42%
	Quincenal	5.03	16.46	31%	25%
	Mensual	4.01	33.50	12%	11%
Zona 9	Diario	0.97	0.72	135%	371%
	Fin de Semana	1.57	1.53	103%	483%
	Entre Semana	2.88	3.12	92%	344%

	Semanal	3.59	4.62	78%	271%
	Quincenal	5.75	9.23	62%	62%
	Mensual	6.33	18.83	34%	29%
Zona 1+2	Diario	0.95	0.91	104%	382%
	Fin de Semana	1.50	2.32	65%	184%
	Entre Semana	2.40	3.77	64%	161%
	Semanal	2.97	6.08	49%	69%
	Quincenal	3.90	11.62	34%	30%
	Mensual	6.02	22.83	26%	24%
Zona 3+4+7	Diario	0.97	0.95	102%	373%
	Fin de Semana	1.25	2.16	58%	107%
	Entre Semana	2.34	4.20	56%	246%
	Semanal	2.78	6.35	44%	42%
	Quincenal	3.95	12.69	31%	34%
	Mensual	3.62	25.50	14%	14%

Elaboración propia

El modelo se entrenó y evaluó en diferentes zonas y para diferentes periodicidades, utilizando las métricas RMSE (Root Mean Squared Error) y MAPE (Mean Absolute Percentage Error).

El análisis muestra que el modelo tiene un desempeño variable en función de la zona y la periodicidad. En general, el modelo tiene un mejor desempeño (RMSE más bajo y MAPE más bajo) en la predicción de accidentes para periodicidades mayores (semanal, quincenal y mensual), en comparación con las periodicidades diarias y de fin de semana.

Para la mayoría de las zonas, el MAPE es más bajo en las predicciones quincenales y mensuales, lo que indica que el modelo es más preciso en esas escalas de tiempo.

El modelo parece tener dificultades para predecir accidentes de tránsito en días específicos, ya que el error es generalmente más alto para las predicciones diarias en todas las zonas. En términos generales, el modelo parece tener un mejor desempeño en la predicción de accidentes en la ciudad en su conjunto, con un RMSE % de 46% y un MAPE de 76% para la periodicidad diaria. El modelo también tiene un buen desempeño en la predicción de accidentes en la Zona 1+2 (combinación de Zona 1 y Zona 2) y en la Zona 3+4+7 (combinación de Zona 3, Zona 4 y Zona 7).

El modelo tiende a tener un mejor desempeño en periodos de tiempo más largos, como quincenal y mensual. Esto podría deberse a que, en periodos más largos, la variabilidad y ruido en los datos se suavizan, permitiendo que el modelo capte patrones subyacentes más fácilmente.

En general, las zonas con menores promedios de accidentes de tránsito parecen presentar valores más altos de RMSE % y MAPE. Esto sugiere que el modelo tiene dificultades para predecir con precisión en zonas y periodos con bajos promedios de accidentes. En estos casos, incluso errores pequeños en las predicciones pueden generar porcentajes de error relativamente altos.

El MAPE muestra valores altos en varios casos, lo que indica que el modelo puede estar teniendo dificultades para predecir con precisión en ciertas zonas y periodos de tiempo. Es importante tener en cuenta que el MAPE es una métrica sensible a errores en casos donde los valores reales son cercanos a cero, por lo que debe ser interpretado con precaución.

En resumen, el modelo Prophet muestra resultados prometedores en la predicción de accidentes de tránsito en la ciudad de Manizales, especialmente para periodicidades mayores. Sin embargo, hay variaciones en el desempeño del modelo en función de la zona y la periodicidad, lo que indica la necesidad de ajustar o mejorar el modelo para cada caso específico, tal como fue realizado durante este trabajo.

4.2 Resultados Light Gradient Boosting.

En la tabla 13 se muestran los valores del error RMSE y MAPE para cada zona en cada tiempo modelado con el modelo de Light Gradient Boosting.

Tabla 13
Medición RMSE y MAPE de cada zona para los modelos basados en Light Gradient Boosting

Zona	Periodicidad	RMSE	PROMEDIO	RMSE %	MAPE
Total General	Diario	2.93	7.61	38%	41%
	Fin de Semana	13.28	53.25	25%	23%
	Entre Semana	11.84	38.31	31%	30%
	Semanal	3.90	14.94	26%	26%
	Quincenal	22.72	106.50	21%	20%
	Mensual	71.75	195.75	37%	38%
Zona 1	Diario	0.57	0.45	125%	127%
	Fin de Semana	1.14	1.10	104%	93%
	Entre Semana	1.66	1.63	102%	83%
	Semanal	2.04	2.69	76%	46%
	Quincenal	3.83	5.38	71%	65%
	Mensual	7.08	10.67	66%	55%
Zona 2	Diario	0.66	0.55	121%	195%
	Fin de Semana	1.21	1.25	96%	101%
	Entre Semana	2.12	2.17	98%	131%
	Semanal	2.37	3.39	70%	178%
	Quincenal	5.10	6.77	75%	75%
	Mensual	6.61	13.00	51%	32%
Zona 3	Diario	0.36	0.27	130%	112%
	Fin de Semana	0.50	0.48	105%	108%
	Entre Semana	1.06	0.97	109%	176%

	Semanal	1.03	1.37	75%	293%
	Quincenal	2.28	2.70	84%	155%
	Mensual	1.15	5.50	21%	16%
Zona 4	Diario	0.21	0.16	134%	39%
	Fin de Semana	0.29	0.20	144%	42%
	Entre Semana	0.55	0.42	131%	83%
	Semanal	0.49	0.52	94%	213%
	Quincenal	0.76	0.95	80%	91%
	Mensual	0.97	2.02	48%	212%
Zona 5	Diario	1.75	2.96	59%	222%
	Fin de Semana	3.54	5.08	70%	53%
	Entre Semana	8.41	15.62	54%	42%
	Semanal	5.47	20.69	26%	26%
	Quincenal	30.96	41.38	75%	73%
	Mensual	17.77	81.67	22%	21%
Zona 6	Diario	0.98	1.01	97%	290%
	Fin de Semana	1.74	1.68	104%	201%
	Entre Semana	3.74	5.12	73%	52%
	Semanal	2.54	6.77	38%	35%
	Quincenal	11.63	13.54	86%	77%
	Mensual	6.64	28.00	24%	15%
Zona 7	Diario	0.79	0.70	112%	309%
	Fin de Semana	1.59	1.63	97%	126%
	Entre Semana	2.05	2.93	70%	133%
	Semanal	2.47	4.54	54%	70%
	Quincenal	7.11	9.08	78%	64%
	Mensual	4.09	18.00	23%	20%
Zona 8	Diario	1.00	1.21	83%	297%
	Fin de Semana	1.54	1.89	82%	71%
	Entre Semana	3.92	6.35	62%	39%
	Semanal	3.25	8.23	39%	47%
	Quincenal	12.35	16.46	75%	68%
	Mensual	9.06	33.50	27%	27%
Zona 9	Diario	0.89	0.72	125%	286%
	Fin de Semana	1.80	1.53	118%	147%
	Entre Semana	3.19	3.12	102%	125%
	Semanal	3.38	4.62	73%	187%
	Quincenal	7.00	9.23	76%	65%
	Mensual	4.27	18.83	23%	13%
	Diario	0.93	0.91	103%	229%

Zona 1+2	Fin de Semana	1.85	2.32	80%	113%
	Entre Semana	2.76	3.77	73%	109%
	Semanal	3.36	6.08	55%	45%
	Quincenal	5.30	11.62	46%	30%
	Mensual	15.78	22.83	69%	75%
Zona 3+4+7	Diario	0.91	0.95	96%	317%
	Fin de Semana	1.22	2.16	57%	110%
	Entre Semana	2.32	4.20	55%	253%
	Semanal	2.71	6.35	43%	43%
	Quincenal	10.11	12.69	80%	68%
	Mensual	3.81	25.50	15%	12%

Elaboración propia

En términos generales, el modelo Light Gradient Boosting parece ofrecer un mejor desempeño en comparación con el modelo Prophet, con menores errores en términos de RMSE y MAPE en la mayoría de los casos.

Las predicciones diarias tienen un desempeño mixto, con algunas zonas mostrando un mayor error porcentual en comparación con otras, lo que indica una mayor variabilidad en la precisión de las predicciones a nivel diario.

En general, las predicciones para fines de semana y entre semana tienen un menor error porcentual en comparación con las predicciones diarias, lo que indica una mayor precisión en la predicción de accidentes en estos periodos.

Las predicciones a nivel semanal, quincenal y mensual tienden a tener un menor error porcentual en comparación con las predicciones diarias, lo que indica una mayor precisión en la predicción de accidentes a medida que aumenta el periodo de tiempo.

Para el total general, el RMSE % en las predicciones diarias es del 38.50%, lo que indica cierta variabilidad en la precisión de las predicciones en este nivel de granularidad. Sin embargo, el RMSE % disminuye a medida que aumenta el periodo de tiempo, siendo del 24.94% para fines de semana, 30.92% para entre semana, 26.08% para semanal, 21.33% para quincenal y 36.66% para mensual.

Al analizar los resultados por zonas, se observa que el RMSE % varía significativamente entre ellas en las predicciones diarias, lo que indica diferencias en la precisión de las predicciones según la zona. Algunas zonas, como la Zona 1 y la Zona 2, tienen un RMSE % mayor al 100% en las predicciones diarias, lo que sugiere que el modelo puede estar teniendo dificultades para predecir accidentes en esas áreas específicas.

A medida que se aumenta el periodo de tiempo, se observa una disminución general en el RMSE % en la mayoría de las zonas. Esto sugiere que el modelo Light Gradient Boosting es más preciso en la predicción de accidentes a medida que se consideran periodos de tiempo más amplios.

Para algunas zonas, como la Zona 3 y la Zona 4, el RMSE % en las predicciones mensuales es significativamente más bajo que en otros periodos, lo que indica una mayor precisión en la predicción de accidentes a nivel mensual en estas áreas.

El modelo LGB parece tener dificultades para predecir accidentes de tránsito en días específicos, ya que el MAPE es generalmente más alto para las predicciones diarias en todas las zonas. Esto también es similar a lo que se observó en el modelo Prophet.

En resumen, el modelo LGB presenta un desempeño variado en las distintas zonas y periodicidades en el dataset de accidentes de tránsito en Manizales. En general, el modelo es más preciso en predicciones quincenales y mensuales, y su precisión es mayor en áreas más amplias. Sin embargo, al igual que el modelo Prophet, el modelo LGB enfrenta dificultades para predecir accidentes de tránsito en días específicos.

4.3 Resultados redes neuronales artificiales recurrentes - LSTM.

En la tabla 14 se muestran los valores del error RMSE y MAPE para cada zona en cada tiempo modelado con el modelo de Light Gradient Boosting.

Tabla 14

Medición RMSE y MAPE de cada zona para los modelos basados en LSTM

Zona	Periodicidad	RMSE	PROMEDIO	RMSE %	MAPE
Total General	Diario	3.39	7.65	44%	52%
	Fin de Semana	10.34	53.39	19%	17%
	Entre Semana	10.16	38.05	27%	26%
	Semanal	4.08	15.34	27%	22%
	Quincenal	23.08	107.50	21%	21%
	Mensual	86.09	165.50	52%	67%
Zona 1	Diario	0.58	0.46	129%	178%
	Fin de Semana	0.96	1.16	83%	189%
	Entre Semana	1.32	1.56	85%	176%
	Semanal	1.98	2.68	74%	50%
	Quincenal	4.13	5.67	73%	62%
	Mensual	4.68	10.50	45%	53%
Zona 2	Diario	0.73	0.54	134%	325%
	Fin de Semana	1.08	1.20	90%	112%
	Entre Semana	1.60	1.93	83%	260%
	Semanal	1.90	3.10	61%	327%
	Quincenal	4.29	6.11	70%	49%
	Mensual	3.82	9.50	40%	39%
Zona 3	Diario	0.41	0.28	147%	214%
	Fin de Semana	0.51	0.55	93%	204%
	Entre Semana	0.97	0.87	111%	501%
	Semanal	1.12	1.34	84%	298%
	Quincenal	1.74	2.68	65%	261%

	Mensual	1.85	6.00	31%	30%
Zona 4	Diario	0.23	0.16	146%	67%
	Fin de Semana	0.32	0.22	145%	67%
	Entre Semana	0.58	0.43	133%	151%
	Semanal	0.55	0.55	99%	229%
	Quincenal	0.65	1.02	64%	168%
	Mensual	0.91	2.50	36%	25%
Zona 5	Diario	1.82	2.98	61%	247%
	Fin de Semana	2.29	5.05	45%	65%
	Entre Semana	5.52	15.59	35%	36%
	Semanal	6.28	20.64	30%	29%
	Quincenal	7.76	42.33	18%	15%
	Mensual	8.66	91.00	10%	10%
Zona 6	Diario	1.03	1.01	102%	284%
	Fin de Semana	1.63	1.89	86%	362%
	Entre Semana	2.44	5.45	45%	33%
	Semanal	2.74	7.32	37%	33%
	Quincenal	5.10	14.67	35%	27%
	Mensual	3.98	25.00	16%	16%
Zona 7	Diario	0.86	0.71	121%	421%
	Fin de Semana	1.15	1.56	74%	209%
	Entre Semana	1.91	2.92	65%	383%
	Semanal	2.79	4.45	63%	89%
	Quincenal	3.31	8.89	37%	39%
	Mensual	3.40	20.50	17%	16%
Zona 8	Diario	1.08	1.22	89%	325%
	Fin de Semana	1.28	1.87	69%	161%
	Entre Semana	2.91	6.09	48%	58%
	Semanal	3.78	7.95	48%	56%
	Quincenal	6.29	15.89	40%	40%
	Mensual	11.54	29.50	39%	39%
Zona 9	Diario	0.94	0.71	132%	414%
	Fin de Semana	1.70	1.66	102%	245%
	Entre Semana	3.15	3.19	99%	281%
	Semanal	3.93	4.82	82%	222%
	Quincenal	4.12	8.78	47%	39%
	Mensual	5.42	21.00	26%	24%
Zona 1+2	Diario	0.98	0.91	108%	319%
	Fin de Semana	1.52	2.33	65%	154%
	Entre Semana	2.40	3.46	69%	167%

	Semanal	2.81	5.77	49%	51%
	Quincenal	6.54	11.00	59%	47%
	Mensual	4.77	17.50	27%	27%
Zona 3+4+7	Diario	0.97	0.96	101%	360%
	Fin de Semana	1.21	2.18	55%	44%
	Entre Semana	2.57	4.10	63%	310%
	Semanal	3.01	6.27	48%	51%
	Quincenal	4.45	12.56	35%	39%
	Mensual	6.46	29.00	22%	22%

Elaboración propia

Al igual que en los dos modelos anteriores se puede observar que el modelo LSTM tiende a tener un mejor rendimiento en la predicción de accidentes para periodos más largos. Específicamente, el RMSE % y MAPE disminuyen a medida que se pasa de periodos diarios y de fin de semana a periodos entre semana, semanales, quincenales y mensuales. Esto sugiere que el modelo captura mejor las tendencias y patrones en los datos cuando se consideran periodos más extensos.

Para el total general, el modelo LSTM muestra una mayor variabilidad en la precisión de las predicciones en periodos cortos, como diario y fin de semana, con un RMSE % de 44.28% y 19.37%, respectivamente, y un MAPE de 52.32% y 17.06%, respectivamente. Esto sugiere que el modelo puede tener dificultades para predecir accidentes con precisión en estos periodos de tiempo.

El modelo LSTM parece tener dificultades para predecir accidentes de tránsito en días específicos, ya que el MAPE es generalmente más alto para las predicciones diarias en todas las zonas. Esto es similar a lo observado en los modelos Prophet y LGB.

Las zonas combinadas (Zona 1+2 y Zona 3+4+7) presentan un MAPE más bajo en comparación con las zonas individuales, lo que sugiere que el modelo LSTM también es más preciso al predecir accidentes de tránsito para áreas más amplias.

Para la mayoría de las zonas, el MAPE es más bajo en las predicciones quincenales y mensuales, lo que indica que el modelo es más preciso en esas escalas de tiempo. Esto podría darse debido a que el número de accidentes aumenta en mayores escalas de tiempo, y el modelo captura más patrones e información, lo cual utiliza en las predicciones.

La predicción del total general muestra un mejor desempeño en la periodicidad diaria y de fin de semana, lo que indica que el modelo es más preciso en esas escalas de tiempo para esa zona en particular.

En resumen, el modelo LSTM presenta un desempeño variado en las distintas zonas y periodicidades en el dataset de accidentes de tránsito en Manizales. En general, el modelo es más preciso en predicciones semanal, quincenal y mensual, y su precisión es mayor en áreas más amplias. Sin embargo, al igual que los modelos Prophet y LGB, el modelo LSTM enfrenta dificultades para predecir accidentes de tránsito en días específicos.

4.4 Comparativo de resultados de los modelos

En la tabla 15 se pueden observar los resultados de los tres algoritmos aplicados, para cada zona y periodo específico se resaltó en color gris el resultado del algoritmo que obtuvo el menor error en términos de la métrica RMSE Porcentual.

Tabla 15

Comparativo de resultados de los modelos

Zona	Periodicidad	RMSE % (Prophet)	RMSE % (LGB)	RMSE % (LSTM)
Total General	Diario	43%	38%	44%
	Fin de Semana	15%	25%	19%
	Entre Semana	19%	31%	27%
	Semanal	28%	26%	27%
	Quincenal	8%	21%	21%
	Mensual	18%	37%	52%
Zona 1	Diario	127%	125%	129%
	Fin de Semana	77%	104%	83%
	Entre Semana	82%	102%	85%
	Semanal	64%	76%	74%
	Quincenal	51%	71%	73%
	Mensual	80%	66%	45%
Zona 2	Diario	124%	121%	134%
	Fin de Semana	63%	96%	90%
	Entre Semana	73%	98%	83%
	Semanal	59%	70%	61%
	Quincenal	42%	75%	70%
	Mensual	56%	51%	40%
Zona 3	Diario	137%	130%	147%
	Fin de Semana	98%	105%	93%
	Entre Semana	104%	109%	111%
	Semanal	80%	75%	84%
	Quincenal	59%	84%	65%
	Mensual	26%	21%	31%
Zona 4	Diario	142%	134%	146%
	Fin de Semana	135%	144%	145%
	Entre Semana	127%	131%	133%
	Semanal	102%	94%	99%

	Quincenal	81%	80%	64%
	Mensual	71%	48%	36%
Zona 5	Diario	61%	59%	61%
	Fin de Semana	46%	70%	45%
	Entre Semana	25%	54%	35%
	Semanal	22%	26%	30%
	Quincenal	15%	75%	18%
	Mensual	18%	22%	10%
Zona 6	Diario	99%	97%	102%
	Fin de Semana	84%	104%	86%
	Entre Semana	48%	73%	45%
	Semanal	40%	38%	37%
	Quincenal	32%	86%	35%
	Mensual	25%	24%	16%
Zona 7	Diario	118%	112%	121%
	Fin de Semana	80%	97%	74%
	Entre Semana	60%	70%	65%
	Semanal	56%	54%	63%
	Quincenal	36%	78%	37%
	Mensual	19%	23%	17%
Zona 8	Diario	87%	83%	89%
	Fin de Semana	68%	82%	69%
	Entre Semana	44%	62%	48%
	Semanal	40%	39%	48%
	Quincenal	31%	75%	40%
	Mensual	12%	27%	39%
Zona 9	Diario	135%	125%	132%
	Fin de Semana	103%	118%	102%
	Entre Semana	92%	102%	99%
	Semanal	78%	73%	82%
	Quincenal	62%	76%	47%
	Mensual	34%	23%	26%
Zona 1+2	Diario	104%	103%	108%
	Fin de Semana	65%	80%	65%
	Entre Semana	64%	73%	69%
	Semanal	49%	55%	49%

	Quincenal	34%	46%	59%
	Mensual	26%	69%	27%
Zona 3+4+7	Diario	102%	96%	101%
	Fin de Semana	58%	57%	55%
	Entre Semana	56%	55%	63%
	Semanal	44%	43%	48%
	Quincenal	31%	80%	35%
	Mensual	14%	15%	22%

Elaboración propia

Según los resultados observados en la tabla 15, podemos ver que no existe un modelo único el cual genere el menor error para todas las zonas y periodicidades, dado esto se pueden utilizar diferentes modelos en función de los mejores resultados para las predicciones.

4.5 Dashboard Accidentes en la ciudad de Manizales

Como parte de los resultados obtenidos en este proyecto, se ha desarrollado un Dashboard de Accidentes para la ciudad de Manizales, que constituye una herramienta valiosa en el análisis y seguimiento de la accidentalidad en la región. Este tablero ha sido creado utilizando el software de visualización MICROSOFT POWER BI, el cual permite la presentación de un análisis descriptivo detallado de los accidentes en la ciudad, así como un análisis predictivo basado en los resultados generados por los algoritmos de predicción empleados en esta investigación.

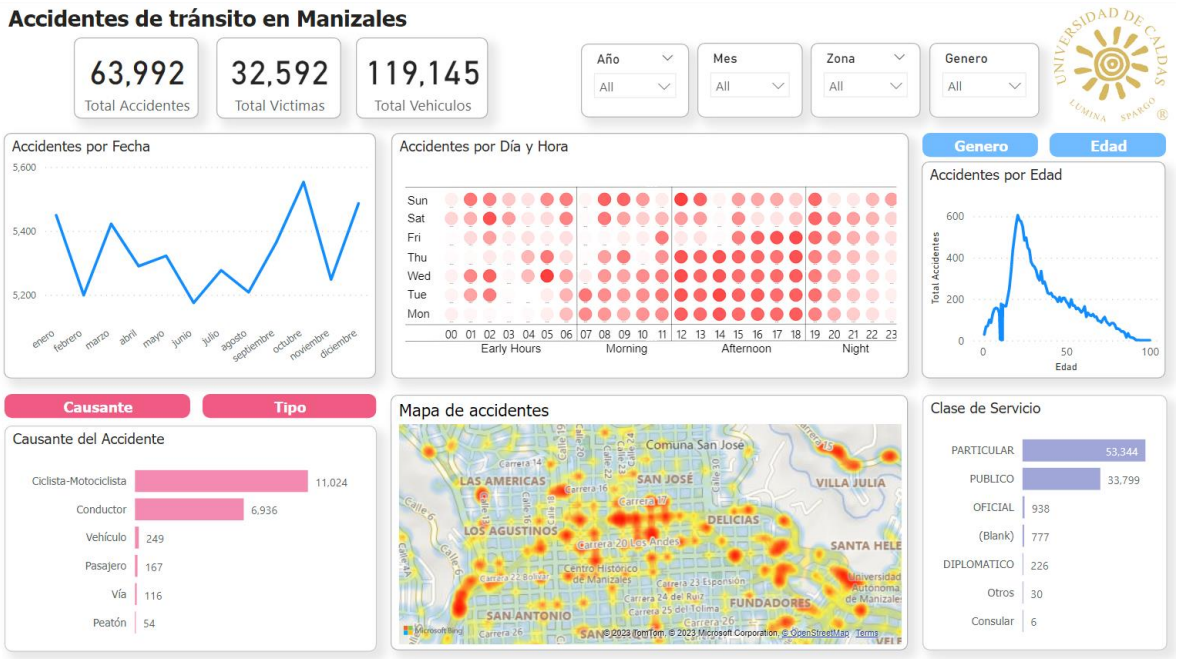
El objetivo principal de este producto es proporcionar una herramienta lista para su implementación, facilitando la aplicación de los resultados de esta investigación por parte del personal de la Secretaría de Movilidad de Manizales. De esta manera, se busca aprovechar al máximo la fase descriptiva de los datos analizados durante el estudio, evitando que estos se conviertan en un análisis meramente retrospectivo. En lugar de ello, se busca crear un recurso que pueda ser constantemente actualizado con datos más recientes y que permita la identificación de nuevos patrones y tendencias en la información.

Este Dashboard de Accidentes se concibe como un instrumento de apoyo a la toma de decisiones y al diseño de políticas públicas enfocadas en la prevención y reducción de accidentes de tránsito. Al ofrecer una herramienta fácilmente accesible y actualizable, se espera que los responsables de la movilidad en la ciudad de Manizales puedan realizar un seguimiento más eficiente y eficaz de la situación de accidentalidad, identificando áreas problemáticas y evaluando el impacto de las intervenciones implementadas.

Inicialmente se creó una interfaz para hacer un análisis descriptivo de los datos, en la figura 25 se observa la primera página del tablero de accidentes.

Figura 25

Resumen accidentes de tránsito en Manizales - Dashboard

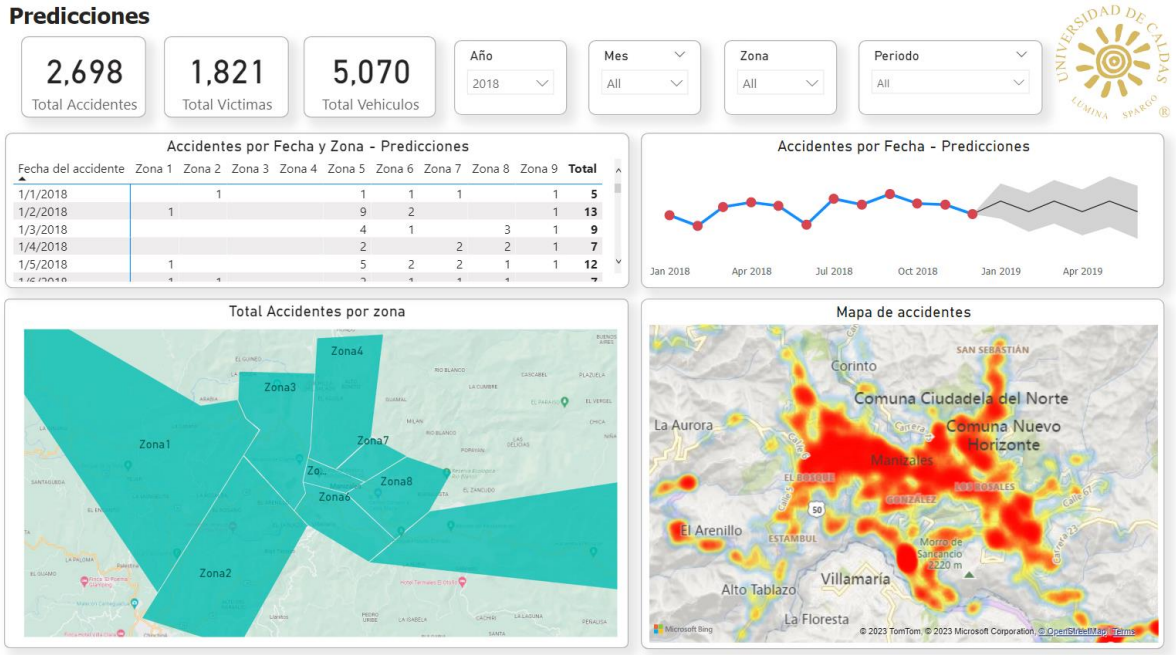


Elaboración propia.

En la Figura 25 se puede llevar a cabo un análisis descriptivo detallado de los accidentes ocurridos en la ciudad de Manizales, el cual nos brinda la posibilidad de filtrar por área geográfica, género o periodo específico. A través de diversas visualizaciones, podemos discernir patrones en la incidencia de dichos accidentes, como la cantidad de accidente ocurridos cada mes, la identificación de las horas y días de la semana con mayor y menor accidentalidad, la distribución de accidentes por género y edad, las causas y tipos de accidentes más comunes y el mapa de calor que permite visualizar las zonas con mayor concentración de accidentes en la ciudad.

Figura 26

Predicción accidentes de tránsito en Manizales - Dashboard



Elaboración propia.

En la Figura 26 se puede observar una vista predictiva de los accidentes, el cual se alimenta de la información extraída de los modelos de predicción creados en esta investigación, la información de las predicciones se muestra en los cuadros superiores tanto tabular como gráficamente junto a los datos reales.

5. CONCLUSIONES Y RECOMENDACIONES.

5.1 Conclusiones.

En este estudio, se exploraron múltiples técnicas de aprendizaje automático para predecir accidentes de tráfico en Manizales, Colombia, utilizando enfoques basados en series temporales como Prophet, algoritmos avanzados como Light Gradient Boosting (LGB) y redes neuronales recurrentes, específicamente Long Short Term Memory (LSTM). Los modelos se construyeron para toda la ciudad y cada zona, utilizando datos de 18 años en general y de 7 años por zonas. Las métricas MAPE y RMSE se emplearon para evaluar el rendimiento de los modelos y la precisión de las predicciones.

Con los modelos desarrollados en esta investigación, es posible realizar predicciones de la cantidad de accidentes en diversos intervalos de tiempo, como diariamente, durante los fines de semana, entre semana, semanalmente, quincenalmente y mensualmente. Estas predicciones abarcan tanto la zona general correspondiente a la jurisdicción del municipio de Manizales como las zonas específicas establecidas. Además, las predicciones generadas pueden complementarse con los análisis espaciales y temporales proporcionados por la herramienta Power BI desarrollada con el propósito de comprender cómo se distribuyen los accidentes en una zona específica y en diferentes días de la semana y horas del día.

En este estudio, se realizaron predicciones para un horizonte de 180 días, centrándose específicamente en el segundo semestre del año 2019. Para el conjunto de datos general, se entrenó el modelo utilizando datos desde el 1 de enero de 2002, lo que resultó en un conjunto de entrenamiento inicial de 6.389 días. Por otro lado, para las distintas zonas, se entrenaron modelos utilizando datos a partir del 1 de enero de 2013, lo que generó un conjunto de entrenamiento inicial de 2.371 días.

El término "entrenamiento inicial" se utiliza porque, durante el primer período de predicción, se emplea esta cantidad de datos. Sin embargo, a medida que el modelo avanza y realiza predicciones para nuevos períodos, se actualiza el entrenamiento utilizando datos reales hasta el período inmediatamente anterior. De esta manera, el conjunto de datos de entrenamiento se amplía progresivamente a lo largo del tiempo.

Los resultados mostraron que el algoritmo Prophet tuvo el mejor rendimiento en general para predecir accidentes de tráfico en Manizales, seguido por LSTM y LGB. El análisis de los resultados sugiere que la combinación de diferentes técnicas de aprendizaje automático puede mejorar la precisión de las predicciones y proporcionar información valiosa para las autoridades en la toma de decisiones, esto se da debido a que hay modelos que tienen mejor desempeño en una zonas o dimensiones temporales que otros, pero en otras zonas o dimensiones temporales son otros los modelos que tienen mejor desempeño, además, se observó que el rendimiento de los modelos variaba según la zona y el horizonte temporal, lo que indica que se pueden utilizar simultáneamente diferentes modelos en función de la zona y horizonte temporal en la que tienen mejor desempeño.

Los modelos de predicción de accidentes de tráfico, como Prophet, LGB y LSTM, muestran un mejor desempeño en horizontes temporales más amplios, siendo más efectivos en predicciones a medio y largo plazo. Estos modelos capturan tendencias y patrones

estacionales, además la precisión es mayor en zonas con más accidentes debido a patrones y factores más consistentes y predecibles, mientras que, en áreas con menos accidentes, los eventos esporádicos dificultan la precisión de las predicciones.

Otro aspecto clave a considerar en la aplicación de los resultados de este estudio es que no es necesario limitarse a un único modelo para todas las zonas y periodos de tiempo. En cambio, es posible combinar los modelos de predicción de accidentes de tráfico, aprovechando el mejor desempeño de cada uno en las zonas y periodos específicos en los que han demostrado ser más efectivos. Esta estrategia de combinación de modelos permitiría obtener predicciones más precisas y adaptadas a las características particulares de cada área y periodo de tiempo. Por ejemplo, se podría utilizar el algoritmo Prophet en zonas y periodos donde haya demostrado un mejor rendimiento en comparación con los otros modelos, mientras que, en otras áreas o periodos, se podría aplicar el LGB o LSTM según su desempeño. De esta manera, se estaría aprovechando al máximo las fortalezas de cada modelo en función de las características específicas de la zona y el horizonte temporal.

Es importante mencionar que la naturaleza de los accidentes de tránsito depende de muchas variables, algunas de las cuales pueden ser difíciles de predecir o incluir en un modelo computacional. Estas variables incluyen factores externos en las vías y factores no predecibles relacionados con los conductores. Algunos de estos factores relevantes son las condiciones climáticas, distracciones del conductor, comportamiento del conductor, condiciones de la vía, volumen de tráfico y eventos especiales variables que no fueron tenidas en cuenta en este estudio. Teniendo en cuenta todos estos factores, es fundamental reconocer que el desempeño de los modelos podría verse afectado por su incapacidad para considerar todas estas variables en la predicción de accidentes de tránsito en la ciudad de Manizales. Por lo tanto, podría ser beneficioso explorar enfoques alternativos o complementarios que puedan tener en cuenta una mayor cantidad de variables relevantes para mejorar la precisión y fiabilidad de las predicciones.

Además de las conclusiones ya mencionadas, este estudio también proporciona una base para futuras investigaciones y aplicaciones en la predicción de accidentes de tráfico en otros entornos urbanos, especialmente en países en desarrollo donde la disponibilidad de datos y la infraestructura pueden ser limitadas, dado que otras investigaciones pueden abordar aspectos no desarrollados en esta investigación y complementarla, así como añadir nuevas variables disponibles a los modelos construidos en esta investigación, lo cuales brinden nuevas perspectivas que podrían mejorar el desempeño de las predicciones.

La incorporación de datos adicionales, como información sobre el clima, eventos especiales, condiciones de la infraestructura y datos socioeconómicos, podría mejorar aún más la precisión y la aplicabilidad de los modelos de predicción. A medida que se recopilan y se vuelven accesibles más datos, se podrán desarrollar modelos más precisos y eficientes, lo que mejorará la capacidad de las autoridades para tomar decisiones informadas y abordar el problema de los accidentes de tráfico.

En el futuro, se podrían explorar enfoques de aprendizaje automático más avanzados y en tiempo real para predecir accidentes de tráfico y proporcionar alertas tempranas a las autoridades y conductores. Estas innovaciones podrían contribuir a reducir aún más el número de accidentes y mejorar la seguridad vial en general, estos enfoques podrían incluir

la integración de datos múltiples, como datos del flujo del tráfico en tiempo real, así como información demográfica, datos climáticos y datos de redes sociales, para obtener una visión más completa y precisa de los factores que contribuyen a los accidentes de tránsito.

En el marco de este estudio, se creó un tablero en Power BI que contiene un análisis descriptivo de los accidentes de tráfico y permite visualizar los resultados de las predicciones realizadas con los modelos Prophet, LGB y LSTM. Este tablero puede ser de gran utilidad para las autoridades y otros interesados en la seguridad vial, ya que facilita la comprensión de los patrones de accidentes de tráfico y las tendencias en la ciudad. Además, el tablero puede ser actualizado con nuevos datos y predicciones a medida que estén disponibles, lo que permite monitorear la evolución de los accidentes de tráfico y la efectividad de las intervenciones implementadas en tiempo real.

Las conclusiones de este estudio tienen implicaciones importantes para la planificación urbana y la gestión del tráfico en Manizales. En primer lugar, el uso de técnicas de aprendizaje automático para predecir accidentes de tráfico puede ayudar a las autoridades a identificar áreas de alto riesgo y a asignar recursos de manera más eficiente para prevenir accidentes y salvar vidas. Además, los resultados sugieren que es fundamental considerar las diferencias en las características y dinámicas de cada zona al desarrollar modelos de predicción, ya que los enfoques generalizados pueden no ser adecuados para todas las áreas.

5.2 Recomendaciones.

- Como recomendación se propone desarrollar un enfoque de recolección de datos más amplio y completo específico para la ciudad de Manizales, considerando factores locales como el clima, la infraestructura vial y las zonas de mayor congestión vehicular. Esta información permitirá crear perfiles o clústeres de áreas propensas a accidentes y facilitar la implementación de sistemas de predicción. Incluir características adicionales en los conjuntos de datos de entrenamiento también mejorará la precisión de los modelos resultantes en el contexto de Manizales.
- Investigar y aplicar técnicas de optimización alternativas, como los algoritmos genéticos, optimización por enjambre de partículas y de colonia de hormigas entre otros, en la búsqueda de hiperparámetros óptimos para mejorar la precisión y el rendimiento de los modelos de predicción de accidentes de tráfico en Manizales, dado que la fase de optimización de hiperparámetros constituyo una parte importante del tiempo dedicado a este proyecto, y es probable que se puedan obtener mejores hiperparámetros con otras técnicas.
- Fomentar la colaboración entre diferentes actores, como las autoridades de tráfico, la policía y las organizaciones de transporte público, para compartir y aprovechar las predicciones de accidentes de tráfico en Manizales. Esto ayudará a maximizar la eficiencia, mejorar la seguridad vial y disminuir el impacto ambiental en la ciudad, en este caso entre la secretaria de movilidad de Manizales, el ministerio de transporte y la policía de carreteras.

- Desarrollar y evaluar estrategias de intervención basadas en las predicciones de accidentes de tráfico en Manizales, como la implementación de campañas de educación vial, mejoras en la señalización y el diseño de infraestructuras, y la promoción del uso de transporte público o medios alternativos de transporte. Estas estrategias pueden contribuir a reducir la incidencia de accidentes en la ciudad.

6. BIBLIOGRAFIA.

Alver, Y., Onelcin, P., Cicekli, A., & Abdel-Aty, M. (2021). Evaluation of pedestrian critical gap and crossing speed at midblock crossing using image processing. *Accident Analysis & Prevention, 156*, 106127. doi:https://doi.org/10.1016/j.aap.2021.106127

Amiri, A. M., Nadimi, N., Khalifeh, V., & Shams, M. (2021). GIS-based crash hotspot identification: a comparison among mapping clusters and spatial analysis techniques. *International journal of injury control and safety promotion*, 1-14.

Banker, S. (2016). A New Big Data Predictive Analytics Solution For Ocean Carriers.

Bao, J., Liu, P., & Ukkusuri, S. V. (2019). A spatiotemporal deep learning approach for citywide short-term crash risk prediction with multi-source data. *Accident Analysis & Prevention, 122*, 239-254.

Bao, W., Yu, Q., & Kong, Y. (2020). *Uncertainty-based traffic accident anticipation with spatio-temporal relational learning.* Paper presented at the Proceedings of the 28th ACM International Conference on Multimedia.

Betancourt, G. A. (2005). Las máquinas de soporte vectorial (SVMs). *Scientia et technica, 1*(27).

BM. (2018). *Las muertes y lesiones causadas por accidentes de tránsito frenan el crecimiento económico de los países en desarrollo*. Retrieved from https://www.bancomundial.org/es/news/press-release/2018/01/09/road-deaths-and-injuries-hold-back-economic-growth-in-developing-countries

BM. (2021). Ingreso mediano alto. Retrieved from https://datos.bancomundial.org/nivel-de-ingresos/ingreso-mediano-alto

Brownlee, J. (2020). *Data preparation for machine learning: data cleaning, feature selection, and data transforms in Python*: Machine Learning Mastery.

Caparrós, A. E. (1999). El comportamiento humano en conducción: factores perceptivos, cognitivos y de respuesta. *Cognición y Psicología Aplicada a la conducción de vehículos*.

Cavanillas, J. M., Curry, E., & Wahlster, W. (2016). *New horizons for a data-driven economy: a roadmap for usage and exploitation of big data in Europe*: Springer Nature.

Chaturvedi, S., Rajasekar, E., Natarajan, S., & McCullen, N. (2022). A comparative assessment of SARIMA, LSTM RNN and Fb Prophet models to forecast total and peak monthly energy demand for India. *Energy Policy, 168*, 113097.

Chen, L., Jia, Y., Sellis, T., & Liu, G. (2014). *Big Data Cleaning. En: Web Technologies and Applications: 16th Asia-Pacific Web Conference, APWeb 2014, Changsha, China, September 5-7, 2014. Proceedings* (Vol. 8709): Springer.

Chen, M., Mao, S., & Liu, Y. (2014). Big Data: A Survey. *Mobile Networks and Applications, 19*(2), 171-209. doi:10.1007/s11036-013-0489-0

Chollet, F. (2021). *Deep learning with Python*: Simon and Schuster.

Clifton, C., & Thuraisingham, B. (2001). Emerging standards for data mining. *Computer Standards & Interfaces, 23*(3), 187-193.

Cukier, K. (2010). Data, data everywhere. *Economist, 394*(8671), 3-5.

Dabiri, S., & Heaslip, K. (2018). Inferring transportation modes from GPS trajectories using a convolutional neural network. *Transportation research part C: emerging technologies, 86*, 360-371.

DANE. (2019). Resultados Censo Nacional de Población y Vivienda 2018. Retrieved from https://www.dane.gov.co/files/censo2018/informacion-tecnica/presentaciones-territorio/191019-CNPV-presentacion-Caldas-Manizales.pdf

de Medrano, R., & Aznarte, J. L. (2021). A New Spatio-Temporal Neural Network Approach for Traffic Accident Forecasting. *Applied Artificial Intelligence*, 1-20.

Díaz Fuentes, D. (2014). Transporte y logística en la economía mundial.

Dominguez Márquez, C., López Castro, A., & González, J. O. (2004). Lesiones anatómicas y factores

relacionados con muertes de ciclistas en

accidentes de transporte. Bogotá, 2004. *Revista de la Facultad de Medicina, 55*(1), 14-23.

Dommelen, T. v. (2021). *Hotel room demand forecasting: time series forecasting using SARIMA, LSTM and Prophet.* ETSI_Informatica.

Ferro, C., Celis Mayorga, N., & Casallas García, A. (2020). *Llenado de series de datos de 2014 a 2019 de PM2.5 por medio de una red neuronal y una regresión lineal*.

Franco-Bedoya, O., Ameller, D., Costal, D., & Franch, X. (2017). Open source software ecosystems: A Systematic mapping. *Information and Software Technology, 91*, 160-185.

Gandomi, A., & Haider, M. (2015). Beyond the hype: Big data concepts, methods, and analytics. *International journal of information management, 35*(2), 137-144.

Gartner. (2012). Gartner Glossary. Retrieved from https://www.gartner.com/en/information-technology/glossary/big-data

Howarth, B. (2014). Big data: how predictive analytics is taking over the public sector. *The Guardian, 13*, 2014.

Hyndman, R. J., & Koehler, A. B. (2006). Another look at measures of forecast accuracy. *International journal of forecasting, 22*(4), 679-688.

IBM. (2010). Bancolombia strengthens anti-money-laundering capabilities with Predictive Analytics.

IBM. (2020). IBM netfinity predictive failure analysis. Retrieved from http://ps-2.kev009.com/pccbbs/pc_servers/pfaf.pdf

INMLCF. (2019). Versión Web de Cifras de Lesiones de Causa Externa en Colombia 2019. Retrieved from https://www.medicinalegal.gov.co/cifras-estadisticas/forensis

Integra. (2021). Ventajas y Desventajas de la Inteligencia Artificial en Empresas. Retrieved from https://nexusintegra.io/es/ventajas-y-desventajas-de-la-inteligencia-artificial/

Joel R. Spiegel, M. T. M., Girish S. Lakshman,Paul G. Nordstrom. (2004). United States Patent No.: A. T. Inc.

Kingma, D. P., & Ba, J. (2014). Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Laney, D. (2001). 3D data management: Controlling data volume, velocity and variety. *META group research note, 6*(70), 1.

LeCun, Y., Bottou, L., Bengio, Y., & Haffner, P. (1998). Gradient-based learning applied to document recognition. *Proceedings of the IEEE, 86*(11), 2278-2324.

Li, Z., Wang, W., Liu, P., Bigham, J. M., & Ragland, D. R. (2013). Using geographically weighted Poisson regression for county-level crash modeling in California. *Safety science, 58*, 89-97.

Liu, M., Wu, J., Wang, Y., & He, L. (2018). Traffic flow prediction based on deep learning. *Journal of System Simulation, 30*(11), 4100.

Mills, S., Lucas, S., Irakliotis, L., Rappa, M., Carlson, T., & Perlowitz, B. (2012). Demystifying big data: a practical guide to transforming the business of government. *TechAmerica Foundation, Washington*.

Moher, D., Liberati, A., Tetzlaff, J., Altman, D. G., & Group, P. (2009). Preferred reporting items for systematic reviews and meta-analyses: the PRISMA statement. *PLoS medicine, 6*(7), e1000097.

Nyce, C. (2007). Predictive Analytics White Paper, sl: American Institute for Chartered Property Casualty Underwriters. *Insurance Institute of America*, 1.

Ongsulee, P., Chotchaung, V., Bamrungsi, E., & Rodcheewit, T. (2018). *Big data, predictive analytics and machine learning.* Paper presented at the 2018 16th International Conference on ICT and Knowledge Engineering (ICT&KE).

Orea, S. V., Vargas, A. S., & Alonso, M. G. (2005). Minería de datos: predicción de la deserción escolar mediante el algoritmo de árboles de decisión y el algoritmo de los k vecinos más cercanos. *Ene, 779*(73), 33.

Oussous, A., Benjelloun, F., Ait, A., & Belfkih, S. (2017). Big data technologies: A survey Journal of King Saud University-Computer and Information Sciences. *em></p*, 01-18.

Pavlyuchenko, K., Panfilov, P., & Gorshkov, G. (2021). EXPLORING THE CAPABILITIES OF PREDICTIVE DATA ANALYTICS IN FMCG INDUSTRY. *Infokommunikacionnye tehnologii, 19*(4), 439-454.

Pérez, M. (2015). *BIG DATA-Técnicas, herramientas y aplicaciones*: Alfaomega Grupo Editor.

Portafolio. (2020). Siniestros viales le cuestan al país 23,9 billones de pesos al año. *Portafolio*. Retrieved from https://www.portafolio.co/economia/a-octubre-en-colombia-fallecieron-4-156-personas-en-siniestros-viales-546657

Pusala, M. K., Salehi, M. A., Katukuri, J. R., Xie, Y., & Raghavan, V. (2016). Massive data analysis: tasks, tools, applications, and challenges *Big Data Analytics* (pp. 11-40): Springer.

Quintero-González, J.-R. (2017). Del concepto de ingeniería de tránsito al de movilidad urbana sostenible. *Ambiente y Desarrollo, 21*(40), 57-72.

Ren, H., Song, Y., Wang, J., Hu, Y., & Lei, J. (2018). *A deep learning approach to the citywide traffic accident risk prediction.* Paper presented at the 2018 21st International Conference on Intelligent Transportation Systems (ITSC).

Rijmenana, M. (2013). Purdue University Achieves Remarkable Results With Big Data. Retrieved from https://datafloq.com/read/purdue-university-achieves-remarkable-results-data/489

Robles Aranda, Y., Trujillo Rasúa, R. A., & Sotolongo León, A. R. (2013). *Algoritmos de minería de datos: Árboles de decisión y Reglas de inducción integrados a PostgreSQL.*

Semana, R. (2021). Las motos representan el 59 % del parque automotor de Colombia. Retrieved from https://www.semana.com/economia/articulo/las-motos-representan-el-59-del-parque-automotor-de-colombia/202157/#:~:text=El%20parque%20automotor%20de%20Colombia,336%20veh%C3%ADculos%20seg%C3%BAn%20el%20RUNT.

Shafabakhsh, G. A., Famili, A., & Bahadori, M. S. (2017). GIS-based spatial analysis of urban traffic accidents: Case study in Mashhad, Iran. *Journal of traffic and transportation engineering (English edition), 4*(3), 290-299.

Strohbach M., D. J., Ravkin H., Lischka M. (2016). *Big Data Storage. En:New horizons for a data-driven economy: a roadmap for usage and exploitation of big data in Europe*: Springer Nature.

Tan, A. E. H. (2018). Planar and network spatial analyses of road traffic accidents-a review of methods and tools.

LEY 769 DE 2002, (2002).

Tiempo, E. (2021). Motociclistas y peatones ponen casi el 80 % de los muertos en las vías. *EL TIEMPO*.

Villavicencio, J. (2010). Introducción a series de tiempo. *Puerto Rico*.

Wassouf, W. N., Alkhatib, R., Salloum, K., & Balloul, S. (2020). Predictive analytics using big data for increased customer loyalty: Syriatel Telecom Company case study. *Journal of Big Data, 7*, 1-24.

Wen, J., Li, S., Lin, Z., Hu, Y., & Huang, C. (2012). Systematic literature review of machine learning based software development effort estimation models. *Information and Software Technology, 54*(1), 41-59.

WHO. (2018). 10 Facts about road safety. Retrieved from https://www.who.int/news-room/facts-in-pictures/detail/road-safety

WHO. (2021). Road traffic injuries. Retrieved from https://www.who.int/news-room/fact-sheets/detail/road-traffic-injuries

Yu, L., Du, B., Hu, X., Sun, L., Han, L., & Lv, W. (2021). Deep spatio-temporal graph convolutional network for traffic accident prediction. *Neurocomputing, 423*, 135-147.

Yuan, Z., Zhou, X., & Yang, T. (2018). *Hetero-convlstm: A deep learning approach to traffic accident prediction on heterogeneous spatio-temporal data.* Paper presented at the Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining.

Zhou, Z. (2019). *Attention based stack resnet for citywide traffic accident prediction.* Paper presented at the 2019 20th IEEE International Conference on Mobile Data Management (MDM).

Zhou, Z., Wang, Y., Xie, X., Chen, L., & Liu, H. (2020). *RiskOracle: A Minute-Level Citywide Traffic Accident Forecasting Framework.* Paper presented at the Proceedings of the AAAI Conference on Artificial Intelligence.

Zong, F., Chen, X., Tang, J., Yu, P., & Wu, T. (2019). Analyzing traffic crash severity with combination of information entropy and Bayesian network. *IEEE Access, 7*, 63288-63302.

7. ANEXOS.

7.1 Modelo construido.

7.1.1 Entorno y librerías

Los modelos presentados en esta investigación fueron construidos y ejecutados en el entorno de Google Colab: https://colab.research.google.com/?hl=es , con el leguaje de programación Python.

Primero se importan las librerias usadas.

```
# Importar librerias
!pip install tensorflow==2.7.0
```

```
!pip install prophet
!pip install pmdarima
!pip install scikit-optimize
!pip install optuna
!pip install keras-tuner
import matplotlib.pyplot as plt
import matplotlib
import pandas as pd
import numpy as np
import plotly.express as px
import lightgbm as lgb
import optuna
import math
import tensorflow as tf
import time
tf.config.run_functions_eagerly(True)
from sklearn.metrics import mean_absolute_percentage_error
from sklearn.metrics import mean_squared_error
from sklearn.preprocessing import MinMaxScaler
from sklearn.metrics import mean_absolute_percentage_error, mean_squared_error
from sklearn.model_selection import RandomizedSearchCV, GridSearchCV,
        train_test_split, TimeSeriesSplit
from hyperopt import fmin, tpe, hp, STATUS_OK, Trials
from functools import partial
from tensorflow.keras.wrappers.scikit_learn import KerasRegressor
from tensorflow.keras.callbacks import EarlyStopping
from tensorflow.keras.optimizers import Adam, Adamax, RMSprop
from itertools import product
from math import sqrt
from pmdarima import ARIMA, auto_arima
from statsmodels.tsa.arima.model import ARIMA
from statsmodels.tsa.statespace.sarimax import SARIMAX
from statsmodels.graphics.tsaplots import plot_acf, plot_pacf
from statsmodels.stats.diagnostic import acorr_ljungbox
from statsmodels.tsa.stattools import adfuller
from scipy.stats import jarque_bera
from scipy.stats import randint as sp_randint
from keras.models import Sequential
from keras.layers import Dense, LSTM, Dropout
from kerastuner.tuners import BayesianOptimization
from optuna import Trial
from optuna.samplers import TPESampler
from prophet import Prophet
from prophet.diagnostics import performance_metrics
from prophet.diagnostics import cross_validation
from skopt import gp_minimize
from skopt.space import Real, Categorical, Integer
```

```
from skopt.utils import use_named_args
from shapely.geometry import Point, Polygon
from google.colab import files
from prettytable import PrettyTable
from io import StringIO
```

7.1.2 Preprocesamiento de los datos

Posteriormente se procede a cargar los datos suministrados por la secretaria de movilidad los cuales se encuentran en archivo (.CSV) y se procede a transformar la información en un formato listo para que se ejecute cada modelo:

```
# Importar datos de accidentes
data = pd.read_csv('/content/drive/Othercomputers/Mi Ordenador/Google Drive
        (flitscol@gmail.com)/Tesis de Maestria/3. Modelos IA/Facebook
        Prophet/Datos/Accidentes.csv', parse_dates=['ds'])
data['ds'] = pd.to_datetime(data['ds'])

# Crear DataFrame con todas las fechas desde 2000-01-01 hasta 2021-12-31
date_range = pd.date_range(start='2000-01-01', end='2021-12-31')
all_dates = pd.DataFrame({'ds': date_range})
all_dates['ds'] = pd.to_datetime(all_dates['ds'])

# Fusionar DataFrames y rellenar valores faltantes con cero con el fin de completar la
        serie en los valores faltantes
data = all_dates.merge(data, on='ds', how='left').fillna(0)

# Filtrar por los años de 2016 a 2019
start_date = start_date#'2002-01-01'
end_date = end_date#'2019-12-31'
df = data.loc[(data['ds'] >= start_date) & (data['ds'] <= end_date)]

# Realizar la interpolación lineal y reemplazar los valores faltantes con los interpolados
        en caso de que no se pueda hacer el calculo se pone un 8 que es promedio de los
        ultimos 8 años
df['y'] = df['y'].replace(0, pd.np.nan).interpolate().fillna(8)

# Importar dias festivos
holidays = pd.read_csv('/content/drive/Othercomputers/Mi Ordenador/Google Drive
        (flitscol@gmail.com)/Tesis de Maestria/3. Modelos IA/Facebook
        Prophet/Datos/Holidays.csv', encoding='latin-1')
holidays['ds'] = pd.to_datetime(holidays['ds'])
# Convierte los festivos en 1
holidays['holiday'] = holidays['holiday'].apply(lambda x: 1 if x!=0 else x)

# Unir información de accidentes con información de festivos
```

```
df = df.merge(holidays, on='ds', how='left').fillna(0)

# agregar columnas de información adicional como dia, mes, año y dia de la semana
df['año'] = df['ds'].dt.year
df['mes'] = df['ds'].dt.month
df['dia'] = df['ds'].dt.day
df['diasem'] = df['ds'].dt.dayofweek
df['fin_de_semana'] = df['ds'].apply(lambda x: 1 if x.dayofweek in [5, 6] else 0)
```

Hasta el momento, el conjunto de datos inicial se encuentra en una escala diaria. A continuación, se llevan a cabo transformaciones para representar esta información en distintos intervalos temporales, incluidos días entre semana, fines de semana, semanas, quincenas y meses. Estos nuevos formatos temporales se aplican tanto al conjunto de datos general como a cada una de las zonas específicas.

7.1.2.1 Creación de zonas

Para realizar la creación de las zonas se extrae la información de las coordenadas generadas con la herramienta google my maps la cual permite exportar la información de las capas creadas para cada zona, cada poligono contiene todas las coordenadas que delimitan el poligono que define cada zona.

```
# Crear polígonos que representan las zonas geográficas de la ciudad de Manizales
Zona1 = Polygon([(5.1117706779791865, -75.70565047972218),(5.186793388711815,
        -75.66434232597818),(5.099274302438001, -
        75.59666825984772),(5.107721836444623, -
        75.56025002480632),(5.0965908310945345, -
        75.55523999780154),(5.078201356546856, -
        75.55953977324218),(5.011366532842681, -75.59917118206164)])
Zona2 = Polygon([(4.985203426208803, -75.61416485198585),(5.078201356546856, -
        75.55953977324218),(5.042286805717766, -
        75.52202838503497),(4.971908930929249, -75.56596887728409)])
Zona3 = Polygon([(5.142658138092109, -75.62990289505488),(5.141436106283816, -
        75.52305297577209),(5.0818568858779125, -
        75.5238705821585),(5.081042604944235, -
        75.52809421555746),(5.075342713053607, -
        75.52932039544865),(5.068692829307098, -
        75.52427931705937),(5.05783561702668, -
        75.52863909804046),(5.053763983023885, -
        75.53422506053288),(5.078201356546856, -
        75.55953977324218),(5.0965908310945345, -
        75.55523999780154),(5.107721836444623, -
        75.56025002480632),(5.099274302438001, -75.59666825984772)])
Zona4 = Polygon([(5.141436106283816, -75.52305297577209),(5.142294462768994, -
        75.40081827136721),(5.077873823668893, -
        75.51645021257515),(5.0818568858779125, -75.5238705821585)])
```

```
Zona5 = Polygon([(5.0818568858779125, -75.5238705821585),(5.081042604944235, -
        75.52809421555746),(5.075342713053607, -
        75.52932039544865),(5.068692829307098, -
        75.52427931705937),(5.064103072936762, -
        75.5261303742918),(5.058553674200892, -
        75.50137292609838),(5.062486416552631, -75.48696362874686)])
Zona6 = Polygon([(5.064103072936762, -75.5261303742918),(5.05783561702668, -
        75.52863909804046),(5.053763983023885, -
        75.53422506053288),(5.042286805717766, -
        75.52202838503497),(5.022749206343955, -
        75.49112408555027),(5.036912596175837, -
        75.48029767371789),(5.058553674200892, -75.50137292609838)])
Zona7 = Polygon([(5.0818568858779125, -75.5238705821585),(5.062486416552631, -
        75.48696362874686),(5.094100251202555, -
        75.45704085971347),(5.096089218457559, -75.48414930220203)])
Zona8 = Polygon([(5.094100251202555, -75.45704085971347),(5.062486416552631, -
        75.48696362874686),(5.058553674200892, -
        75.50137292609838),(5.036912596175837, -
        75.48029767371789),(5.075373891912557, -75.42415375258541)])
Zona9 = Polygon([(5.075373891912557, -75.42415375258541),(5.036912596175837, -
        75.48029767371789),(5.022749206343955, -
        75.49112408555027),(4.983276647652719, -
        75.30495100493002),(5.084527019791259, -75.30836670458741)])
```

Posteriormente se importan los datos de los accidentes con las coordenadas correspondientes a cada accidente.

```
# Importar datos de accidentes
coord = pd.read_csv('/content/drive/Othercomputers/Mi Ordenador/Google Drive
(flitscol@gmail.com)/Tesis de Maestria/3. Modelos IA/Facebook
Prophet/Datos/Coordenadas.csv', parse_dates=['Fecha del accidente'],
encoding='ISO-8859-1')
coord['Fecha del accidente'] = pd.to_datetime(coord['Fecha del accidente'])

# Filtrar datos por los que tienen la ubicación (se eliminan los que no tienen una
ubicación)
coord_ok = coord.dropna(subset=['Latitud'])

# Cargar las coordenadas en un DataFrame de Pandas
df_accidentes = pd.DataFrame({
   'x': coord_ok['Latitud'],
   'y': coord_ok['Longitud'],
   'NRO_CROQUIS': coord_ok['NRO_CROQUIS']
})
```

Una vez cargada la información de las coordenadas que le corresponden a cada accidente, se crea una funciona que asigne cada accidente a la zona correspondiente.

```
# Función para asignar cada accidente a la zona correspondiente
def asignar_zona(x, y):
    punto = Point(x, y)
    if punto.within(Zona1):
        return 'Zona 1'
    elif punto.within(Zona2):
        return 'Zona 2'
    elif punto.within(Zona3):
        return 'Zona 3'
    elif punto.within(Zona4):
        return 'Zona 4'
    elif punto.within(Zona5):
        return 'Zona 5'
    elif punto.within(Zona6):
        return 'Zona 6'
    elif punto.within(Zona7):
        return 'Zona 7'
    elif punto.within(Zona8):
        return 'Zona 8'
    elif punto.within(Zona9):
        return 'Zona 9'
    else:
        return 'Desconocido'

# Aplicar la función a cada punto de accidente
df_accidentes['zona'] = df_accidentes.apply(lambda row: asignar_zona(row['x'],
row['y']), axis=1)
```

Luego se añade la informacion de la zona al dataset

```
# Añadir la información de zona a cada accidente
df_accidentes = df_accidentes[['NRO_CROQUIS','zona']]
coord_ok = coord_ok.merge(df_accidentes, how = str('left'), on='NRO_CROQUIS')
```

Se agrupa la información de manera que para cada dia muestre la cantidad de accidentes por cada zona y en general, para esto se creo una columna por cada zona.

```
# Tabla resumida con la cantidad de accidentes por dia en cada zona
coord_ok = coord_ok.rename(columns={'Fecha del accidente': 'ds'})
coord_resumen = pd.pivot_table(coord_ok, values='Latitud', index='ds',
columns='zona', aggfunc=len)
coord_resumen = coord_resumen.fillna(0)

# Fusionar DataFrames y rellenar valores faltantes con cero
df_coord = df.merge(coord_resumen, on='ds', how='left').fillna(0)
```

7.1.2.2 Creación de datasets en diferentes escalas temporales

Una vez que se tiene la información por zonas y general se procede a realizar la creación de los dataset en escalas de tiempo diaria, entre semana, fin de semana, semanal, quincenal y mensual, para todas las zonas y para cada zona.

Importación de regresores adicionales:

Primero se importan los regresores adicionales, estos regresores contienen la información de ocurrencia de diferentes eventos que inciden en la cantidad de accidentes de tránsito y comprenden diversos eventos como inicio de clases y periodos de vacaciones en colegios y universidades hasta eventos de ferias o fechas especiales.

```
# Importar regresores en periodos semanales
regresor_sem = pd.read_csv('/content/drive/Othercomputers/Mi Ordenador/Google Drive (flitscol@gmail.com)/Tesis de Maestria/3. Modelos IA/Facebook Prophet/Datos/Regresores_entre_semana.csv', parse_dates=['ds'], encoding='ISO-8859-1')
regresor_sem['ds'] = pd.to_datetime(data['ds'])
regresor_sem.fillna(value=0, inplace=True)
regresor_sem = regresor_sem[['ds','SUBE','BAJA']]

# Importar regresores en periodos de fines de semana
regresor_finde = pd.read_csv('/content/drive/Othercomputers/Mi Ordenador/Google Drive (flitscol@gmail.com)/Tesis de Maestria/3. Modelos IA/Facebook Prophet/Datos/Regresores_finde_semana.csv', parse_dates=['ds'], encoding='ISO-8859-1')
regresor_finde['ds'] = pd.to_datetime(data['ds'])
regresor_finde.fillna(value=0, inplace=True)
regresor_finde = regresor_finde[['ds','SUBE','BAJA']]

# Unir los regresores a los dataset completos y por fin de semana
df_semanal_regresor = df_semanal.merge(regresor_sem, on='ds', how='left')
df_entre_sem_regresor = df_entre_semana.merge(regresor_sem, on='ds', how='left')
df_finde_sem_regresor = df_finde_semana.merge(regresor_finde, on='ds', how='left')
```

Creación de datasets por zona y periodo:

Posteriormente se realiza la creacion de las tablas base para cada periodo con la información de todas las zonas, es decir que se deja una tabla para el periodo diario que contiene la información de los accidentes diariamente y contiene la información de todas las zonas y asi con cada periodo de tiempo

```
# Se crea la tabla que almacena la información en periodo diario
df_zona = df_coord
```

```
# Se añaden los regresores
df_entre_semana_regresor = df_entre_semana.merge(regresor_sem, on='ds',
how='left')
df_finde_semana_regresor = df_finde_semana.merge(regresor_finde, on='ds',
how='left')

# Añadir regresores a las zonas con periodicidad semanal
df_sem_zona_regresor = df_sem_zona.merge(regresor_sem, on='ds', how='left')

# TablaS base para entre semana y fin de semana
df1 = df_coord
filter_entre = df1['diasem']<=4
filter_finde = df1['diasem']>4
df_entresem_zona = df1[filter_entre]
df_findesem_zona = df1[filter_finde]
# Cambiar la periodicidad de los datos a semanal
df_entresem_zona['semana'] = df_entresem_zona['ds'].dt.to_period('W')
df_findesem_zona['semana'] = df_findesem_zona['ds'].dt.to_period('W')

# Generar dataset Entre semana para todas las zonas
df_entresem_zona = df_entresem_zona.groupby('semana')['y','holiday','Zona
1','Zona 2','Zona 3','Zona 4','Zona 5','Zona 6','Zona 7','Zona 8','Zona
9'].sum().reset_index()
df_entresem_zona['semana'] = df_entresem_zona['semana'].astype(str)
df_entresem_zona[['inicio', 'fin']] = df_entresem_zona['semana'].str.split('/',
expand=True)
df_entresem_zona['inicio'] = pd.to_datetime(df_entresem_zona['inicio'])
df_entresem_zona['fin'] = pd.to_datetime(df_entresem_zona['fin'])
df_entresem_zona.drop(['semana','fin'], axis=1, inplace=True)
df_entresem_zona.rename(columns={'inicio': 'ds'}, inplace=True)
df_entresem_zona = df_entresem_zona.merge(regresor_sem, on='ds', how='left')

# Generar dataset Fin de semana para todas las zonas
df_findesem_zona = df_findesem_zona.groupby('semana')['y','holiday','Zona
1','Zona 2','Zona 3','Zona 4','Zona 5','Zona 6','Zona 7','Zona 8','Zona
9'].sum().reset_index()
df_findesem_zona['semana'] = df_findesem_zona['semana'].astype(str)
df_findesem_zona[['inicio', 'fin']] = df_findesem_zona['semana'].str.split('/',
expand=True)
df_findesem_zona['inicio'] = pd.to_datetime(df_findesem_zona['inicio'])
df_findesem_zona['fin'] = pd.to_datetime(df_findesem_zona['fin'])
df_findesem_zona.drop(['semana','fin'], axis=1, inplace=True)
df_findesem_zona.rename(columns={'inicio': 'ds'}, inplace=True)
df_findesem_zona = df_findesem_zona.merge(regresor_finde, on='ds', how='left')
```

Creación de datasets por zona y periodo:

Posteriormente se crea una función para que itere por cada zona y genere los datasets para cada zona en todos los periodos:

```
# Funcion para generar tablas por zonas

generar_tablas_zonas = """

filter1 = df_entresem_zona['ds']>='2012-12-31'
filter2 = df_findesem_zona['ds']>='2012-12-31'
filter3 = df_zona['ds']>='2012-12-31'

Z1 = df_sem_zona_regresor[filter]
Z1 = Z1.drop(columns=DROP)
Z1['Participacion'] = Z1[SELECCION]/Z1['y']
#Z1['Participacion_media'] =
Z1['Participacion'].expanding().mean().shift(fill_value=0)
#Z1['Participacion_media_52'] =
Z1['Participacion'].rolling(52).mean().shift(fill_value=0)
#Z1['Participacion_media_26'] =
Z1['Participacion'].rolling(26).mean().shift(fill_value=0)
#Z1['Participacion_media_13'] =
Z1['Participacion'].rolling(13).mean().shift(fill_value=0)
#Z1['Participacion_media_6'] =
Z1['Participacion'].rolling(6).mean().shift(fill_value=0)
#Z1['Participacion_media_3'] =
Z1['Participacion'].rolling(3).mean().shift(fill_value=0)
#Z1['Participacion_media_1'] =
Z1['Participacion'].rolling(1).mean().shift(fill_value=0)

# Renombrar columnas 'A' y 'B'
Z1 = Z1.rename(columns={'y': 'Total', SELECCION: 'y'})
Z1 = Z1.dropna()
Z1['y'] = Z1['y'].replace(0, 0.1)

# 2 Semanas
S2 = df_sem_zona_regresor[filter]
S2 = S2.resample("2W", on="ds").sum()
S2 = S2.drop(columns=DROP)
S2['Participacion'] = S2[SELECCION]/S2['y']
S2 = S2.rename(columns={'y': 'Total', SELECCION: 'y'})
S2.reset_index(inplace=True)
S2['y'] = S2['y'].replace(0, 0.1)

# 4 Semanas
S4 = df_sem_zona_regresor[filter]
```

```
S4 = S4.resample("4W", on="ds").sum()
S4 = S4.drop(columns=DROP)
S4['Participacion'] = S4[SELECCION]/S4['y']
S4 = S4.rename(columns={'y': 'Total', SELECCION: 'y'})
S4.reset_index(inplace=True)

# Entre semana
ES = df_entresem_zona[filter1]
ES = df_entresem_zona.drop(columns=DROP)
ES['Participacion'] = ES[SELECCION]/ES['y']
ES = ES.rename(columns={'y': 'Total', SELECCION: 'y'})
ES['y'] = ES['y'].replace(0, 0.1)

# Fin de semana
FS = df_findesem_zona[filter2]
FS = df_findesem_zona.drop(columns=DROP)
FS['Participacion'] = FS[SELECCION]/FS['y']
FS = FS.rename(columns={'y': 'Total', SELECCION: 'y'})
FS['y'] = FS['y'].replace(0, 0.1)

# Diario
D = df_zona[filter3]
D = D.drop(columns=DROP)
D['Participacion'] = D[SELECCION]/D['y']
D = D.rename(columns={'y': 'Total', SELECCION: 'y'})
D = D.dropna()
D['y'] = D['y'].replace(0, 0.1)

"""
```

La anterior función es almacenada como codigo en texto y llamada mediante el comando exec(funcion) de esta manera el siguiente codigo se encarga de hacer las iteraciones correspondiente para cada zona.

```
# ZONA 1 Adaptar tabla semanal por zona e implementar regresores teniendo en cuenta el valor medio de participación de la zona en el total de accidentes

filter = df_sem_zona_regresor['ds']>='2012-12-31'

DROP = ['Zona 9', 'Zona 5', 'Zona 2', 'Zona 3', 'Zona 4', 'Zona 6', 'Zona 7', 'Zona 8']
SELECCION = 'Zona 1'

exec(generar_tablas_zonas)

# Renombrar

df_sem_zona_regresor_filtered_Z1_D = D
```

```
df_sem_zona_regresor_filtered_Z1 = Z1
df_sem_zona_regresor_filtered_Z1_2S = S2
df_sem_zona_regresor_filtered_Z1_4S = S4
df_sem_zona_regresor_filtered_Z1_ES = ES
df_sem_zona_regresor_filtered_Z1_FS = FS
```

Y se tiene el codigo anterior para cada zona, de esta forma se generan todos los dataset que van a ser posteriomente usados.

Hasta este punto se realizo pero prepocesamiento de datos

7.1.3 Implementación de los algoritmos de predicción

Primero se realiza la importación de una tabla que contiene los nombres de los 72 datasets generados en la primera parte.

```
# Dataset que contiene todos la información de los datasets a evaluar con los horizontes de evaluación
dataset = pd.read_csv('/content/drive/Othercomputers/Mi Ordenador/Google Drive (flitscol@gmail.com)/Tesis de Maestria/3. Modelos IA/Facebook Prophet/Datos/DATASET.csv', encoding='latin-1')
dataset
```

Figura 27
Imagen de la tabla que contiene la información para la realización de las predicciones

	DATASET	PERIODO	ZONA	PREDICCIONES	VENTANA
0	df	D	Total	26	7
1	df_semanal_regresor	S	Total	26	1
2	df_semanal_regresor_2S	2S	Total	13	1
3	df_semanal_regresor_4S	4S	Total	6	1
4	df_entre_semana_regresor	S	Total	26	1
...	...	...	...	...	...
67	df_sem_zona_regresor_filtered_Z347_2S	2S	Zona 347	13	1
68	df_sem_zona_regresor_filtered_Z347_4S	4S	Zona 347	6	1
69	df_sem_zona_regresor_filtered_Z347_ES	S	Zona 347	26	1
70	df_sem_zona_regresor_filtered_Z347_FS	S	Zona 347	26	1
71	df_sem_zona_regresor_filtered_Z347_D	D	Zona 347	26	7

72 rows × 5 columns

Elaboración propia.

En la imagen anterior podemos ver en la columna DATASET el nombre cada dataset por zona y periodo, PERIODO contiene la indormacion del periodo de cada dataset, ZONA es el nombre se la zona, PREDICCIONES es la cantidad de predicciones que debe realizar cada algoritmo y VENTANA es la cantidad de periodos que el modelo debe predecir en cada predicion

Ahora comienza el entrenamiento y prediccion de los algoritmos con cada uno de los datasets anteriores realizando un total de 226 predicciones. De manera general los 3 algoritmos de prediccion implementados funcionan de manera similar, primero se crearon las funciones que realizan la optimizacion de hiperparametros y luego el entrenamiento y prediccion con los parametros encontrados, lo anterior dentro de un codigo que realiza esta misma iteración para todos los datasets y al mismo tiempo almacena los resultados de las predicciones y de las métricas de evaluación de cada modelo.

7.1.3.1 Facebook Prophet:

```
# Codigo de iteración para correr todas las zonas en todos los tiempos FACEBOOK PROPHET

predicciones_totales_T = []
resultados_tabla_T = []
```

```
for index, row in dataset.iterrows():
    DATASETN = row['DATASET']
    PERIODO = row['PERIODO']
    ZONA = row['ZONA']
    PREDICCIONES = row['PREDICCIONES']
    VENTANA = row['VENTANA']

    DATASET = globals()[DATASETN]
    ALGORITMO = 'Facebook Prophet'

    # OPTIMIZAR HIPERPARÁMETROS

    exec(optimizar_hiperparametros)

    # EJECUTAR ALGORITMO

     exec(facebook_prophet)

    # ALMACENAR RESULTADOS

      exec(resultados)

predicciones_finales_prophet = pd.concat(predicciones_totales_T)
resultados_finales_prophet = pd.concat(resultados_tabla_T)
```

El codigo anterior hace que el modelo de facebook prophet se ejecute para cada dataset, busque los hiperparametros optimizados y realice las predicciones en el horizonte de tiempo indicado y almacene los resultados

A continuación se muestran las 3 funciones que ejecuta en cada iteración:

Esta funcion optimiza los hiperparametros:

```
optimizar_hiperparametros = """

df = DATASET
particiones_sem = PREDICCIONES
semanas = VENTANA

# Dividir tiempos de prediccion
holidays['holiday'] = holidays['holiday'].apply(lambda x: 'f' if x!=0 else x) #
Convierte los festivos en 1
te = particiones_sem * semanas
train = df.iloc[:len(df)-te, :]
test = df.iloc[len(df)-te:, :]

# Define the hyperparameter space to search over
```

```
space = [
    Real(0.0001, 1.0, name='changepoint_prior_scale'),
    Categorical(['additive', 'multiplicative'], name='seasonality_mode'),
    Real(0.01, 50.0, name='seasonality_prior_scale')
]

@use_named_args(space)
def objective(changepoint_prior_scale, seasonality_mode, seasonality_prior_scale):
    # Create a Prophet model with the current hyperparameters
    m = Prophet(changepoint_prior_scale=changepoint_prior_scale,
holidays=holidays, weekly_seasonality=True, daily_seasonality=True,
yearly_seasonality=True, seasonality_mode='multiplicative',
seasonality_prior_scale=seasonality_prior_scale)

    # Fit the model on the training data
    m.fit(train)

    # Make predictions on the test data
    predictions = m.predict(test)[['ds', 'yhat']]

    # Calculate the MAPE of the predictions
    actual = test['y']
    predicted = predictions['yhat'][-len(test):]
    mape = mean_absolute_percentage_error(actual, predicted)

    return mape

# Establecer el tiempo límite en segundos
time_limit = 8000  # 2 horas, por ejemplo

# Crear una función de pérdida con límite de tiempo
def loss_with_timeout(start_time, time_limit, *args, **kwargs):
    if time.time() - start_time > time_limit:
        raise TimeoutError("Se ha alcanzado el límite de tiempo.")
    return objective(*args, **kwargs)

# Crear una función parcial que incluye el tiempo de inicio y el límite de tiempo
start_time = time.time()
loss_func = partial(loss_with_timeout, start_time, time_limit)

# Ejecutar la optimización bayesiana con la función de pérdida ajustada
try:
    result = gp_minimize(loss_func, space, n_calls=100, random_state=42)
except TimeoutError as e:
    print(e)
    result = result if result else None  # Si no se han completado trials, `result` será
None
```

```
if result:
    x0 = result.x[0]
    x1 = result.x[1]
    x2 = result.x[2]
    # Utiliza los mejores hiperparámetros encontrados hasta ahora
else:
    print("No se han completado trials antes del tiempo límite.")

"""
```

Esta funcion ejecuta el modelo con los hiperparametros encontrados:

```
facebook_prophet = """

# Hiperparametros Optimizados
changepoint_prior_scale = x0
seasonality_mode = x1
seasonality_prior_scale = x2

# Establecer parámetros de validación cruzada
df = DATASET
df = df.reset_index(drop=True)
particiones = PREDICCIONES
dias_particion = VENTANA
n_dias = len(df)
t_inicial = n_dias - (particiones*dias_particion)

# Inicializar variables para almacenar resultados
mape_resultados = []
mse_resultados = []
rmse_resultados = []
promedio_resultados = []
y_min_resultados = []
y_max_resultados = []
rango_resultados = []
predicciones = []
changepoint_prior_scale_resultado = []
seasonality_mode_resultado = []
seasonality_prior_scale_resultado = []

# Loop de validación cruzada y ejecución del modelo
for i in range(particiones):
    # Establecer los límites de los datos de entrenamiento y prueba para esta iteración
    train_inicio = 0
    train_fin = i * dias_particion + t_inicial
    test_inicio = train_fin
```

```
    test_fin = test_inicio + dias_particion
    holidays['holiday'] = holidays['holiday'].apply(lambda x: 'f' if x!=0 else x) # Convierte los festivos en 1

    # Dividir los datos en conjuntos de entrenamiento y prueba
    train = df.iloc[train_inicio:train_fin]
    test = df.iloc[test_inicio:test_fin]

    # Crear modelo
    model = Prophet(changepoint_prior_scale=changepoint_prior_scale, holidays=holidays, weekly_seasonality=True, daily_seasonality=True, yearly_seasonality=True, seasonality_mode='multiplicative', seasonality_prior_scale=seasonality_prior_scale)
    model.add_regressor('holiday')
    #model.add_regressor('Total')
    if PERIODO == 'D':
        pass
    else:
        model.add_regressor('SUBE')
        model.add_regressor('BAJA')

    # Entrenar modelo
    model.fit(train)

    # Obtener predicciones
    predictions = model.predict(test)[['ds', 'yhat']]

    # Unir predicciones con datos reales
    results = pd.concat([test.set_index('ds')['y'], predictions.set_index('ds')['yhat']], axis=1, join='inner')
    results.columns = ['y', 'yhat']

    # Cambia lo valores en cero por 0.1 para evitar errores en el calculo del MAPE
    results['y'] = results['y'].replace(0,0.1)

    # Almacenar resultados
    predicciones.append(results)

# Recopilar los resultados de cada iteración
predicciones_totales = pd.concat(predicciones)
predicciones_totales = predicciones_totales.reset_index()
predicciones_totales['DATASET'] = DATASETN
predicciones_totales['PERIODO'] = PERIODO
predicciones_totales['ZONA'] = ZONA
predicciones_totales['PREDICCIONES'] = PREDICCIONES
predicciones_totales['VENTANA'] = VENTANA
predicciones_totales['ALGORITMO'] = ALGORITMO
```

```
# Calcular las métricas de evaluación
mape = mean_absolute_percentage_error(predicciones_totales['y'],
predicciones_totales['yhat'])
mse = mean_squared_error(predicciones_totales['y'], predicciones_totales['yhat'])
rmse = mse ** 0.5
promedio = predicciones_totales['y'].mean()
y_min = predicciones_totales['y'].min()
y_max = predicciones_totales['y'].max()
rango = y_max - y_min

# Almacenar resultados
mape_resultados.append(mape)
mse_resultados.append(mse)
rmse_resultados.append(rmse)
promedio_resultados.append(promedio)
y_min_resultados.append(y_min)
y_max_resultados.append(y_max)
rango_resultados.append(rango)
changepoint_prior_scale_resultado.append(changepoint_prior_scale)
seasonality_mode_resultado.append(seasonality_mode)
seasonality_prior_scale_resultado.append(seasonality_prior_scale)

# Crear tabla de resultados
resultados_tabla = pd.DataFrame({
   'MAPE': mape_resultados,
   'MSE': mse_resultados,
   'RMSE': rmse_resultados,
   'Promedio': promedio_resultados,
   'Minimo': y_min_resultados,
   'Maximo': y_max_resultados,
   'Rango': rango_resultados,
   'changepoint_prior_scale': changepoint_prior_scale_resultado,
   'seasonality_mode': seasonality_mode_resultado,
   'seasonality_prior_scale': seasonality_prior_scale_resultado
})
#resultados_tabla.index = np.arange(1, particiones+1)
resultados_tabla['DATASET'] = DATASETN
resultados_tabla['PERIODO'] = PERIODO
resultados_tabla['ZONA'] = ZONA
resultados_tabla['PREDICCIONES'] = PREDICCIONES
resultados_tabla['VENTANA'] = VENTANA
resultados_tabla['ALGORITMO'] = ALGORITMO

"""
```

Esta función alamacena los resultados obtenidos:

```
resultados = """

predicciones_totales_T.append(predicciones_totales)
resultados_tabla_T.append(resultados_tabla)

"""
```

7.1.3.2 Light Gradient Boosting (LGB):

```
# Codigo de iteración para correr todas las zonas en todos los tiempos con LGB

predicciones_finales_LGB = []
resultados_finales_LGB = []

for index, row in dataset.iterrows():
   DATASETN = row['DATASET']
   PERIODO = row['PERIODO']
   ZONA = row['ZONA']
   PREDICCIONES = row['PREDICCIONES']
   VENTANA = row['VENTANA']

   DATASET = globals()[DATASETN]
   ALGORITMO = 'Light Gradient Boosting'

  # PREPROCESAMIENTO

   exec(eliminar_columnas_inutiles)

  # OPTIMIZAR HIPERPARÁMETROS

   exec(optimizar_hiperparametros_LGB)

  # EJECUTAR ALGORITMO

   exec(modelo_LGB)

  # ALMACENAR RESULTADOS

   exec(resultados_LGB)

predicciones_finales_LGB = pd.concat(predicciones_finales_LGB)
resultados_finales_LGB = pd.concat(resultados_finales_LGB)
```

El codigo anterior hace que el modelo de LGB se ejecute para cada dataset, realice un preprocesamiento adicional a los datos para que puedan ser leidos por el modelo, busque

los hiperparametros optimizados y realice las predicciones en el horizonte de tiempo indicado y almacene los resultados.

A continuación se muestran las 4 funciones que ejecuta en cada iteración:

Esta función elimina columnas que no usa el modelo:

```
eliminar_columnas_inutiles = """

# Selecciona solo las columnas del tipo de datos que cumplan la condición
columnas_utiles = DATASET.select_dtypes(include=["int", "float",
"bool"]).columns.tolist()

# Elimina todas las columnas que no cumplen la condición
columnas_no_utiles = set(DATASET.columns) - set(columnas_utiles) - set(["ds"])
DATASET.drop(columnas_no_utiles, axis=1, inplace=True)

# Borrar columnas que no deberian estar
columnas_a_eliminar = ['Total', 'Zona 12', 'total', 'Zona 347', 'Desconocido',
'Participacion']

# Elimina solo las columnas que existen en el DataFrame
for columna in columnas_a_eliminar:
    if columna in df.columns:
        df.drop(columna, axis=1, inplace=True)

"""
```

Esta funcion optimiza los hiperparametros:

```
optimizar_hiperparametros_LGB = """

# Hallar los mejores hiperparametros

# Divide los datos en conjuntos de entrenamiento y prueba (últimas 20 semanas)
df = DATASET
periodo = VENTANA * PREDICCIONES
#periodo = 50
train_df = df[:-periodo]
test_df = df[-periodo:]

# Prepara los datos para el modelo LightGBM
X_train = train_df.drop(['ds', 'y'], axis=1)
y_train = train_df['y']
X_test = test_df.drop(['ds', 'y'], axis=1)
y_test = test_df['y']
```

```
        # Define la función objetivo que queremos minimizar (en este caso, el MAPE)
        def objective(trial):
            params = {
                'boosting_type': 'gbdt',
                'objective': 'regression',
                'metric': 'mape',
                'num_leaves': trial.suggest_int('num_leaves', 10, 100),
                'learning_rate': trial.suggest_float('learning_rate', 0.001, 0.1),
                'feature_fraction': trial.suggest_float('feature_fraction', 0.1, 1.0),
                'bagging_fraction': trial.suggest_float('bagging_fraction', 0.1, 1.0),
                'bagging_freq': trial.suggest_int('bagging_freq', 1, 10),
                'verbose': 0
            }

            lgb_train = lgb.Dataset(X_train, y_train)
            lgb_eval = lgb.Dataset(X_test, y_test, reference=lgb_train)
            model = lgb.train(params, lgb_train, num_boost_round=1000,
        valid_sets=lgb_eval, early_stopping_rounds=100, verbose_eval=False)

            y_pred = model.predict(X_test, num_iteration=model.best_iteration)
            mape = mean_absolute_percentage_error(y_test, y_pred)

            return mape

        # Ejecuta la optimización de hiperparámetros con Optuna
        study = optuna.create_study(direction='minimize')
        study.optimize(objective, n_trials=200, timeout=8000)  # n_trials: número de
        iteraciones, timeout: tiempo máximo en segundos
        best_params = study.best_params

        """
Esta funcion ejecuta el modelo con los hiperparametros encontrados:
        modelo_LGB = """

        # Entrena el modelo con los mejores hiperparámetros
        best_params = best_params
        best_params['boosting_type'] = 'gbdt'
        best_params['objective'] = 'regression'
        best_params['metric'] = 'mape'
        best_params['verbose'] = 0

        # Establecer parámetros de validación cruzada
        df = DATASET
        particiones = PREDICCIONES
        dias_particion = VENTANA
        n_dias = len(df)
        t_inicial = n_dias - (particiones*dias_particion)
```

```
totalsemanas = particiones*dias_particion

exec(eliminar_columnas_inutiles) # Elimina columnas inutiles

# Inicializar variables para almacenar resultados
mape_resultados = []
mse_resultados = []
rmse_resultados = []
promedio_resultados = []
y_min_resultados = []
y_max_resultados = []
rango_resultados = []
predicciones = []
num_leaves_resultados = []
learning_rate_resultados = []
feature_fraction_resultados = []
bagging_fraction_resultados = []
bagging_freq_resultados = []

# Loop de validación cruzada
for i in range(particiones):
    # Establecer los límites de los datos de entrenamiento y prueba para esta iteración
    train_inicio = 0
    train_fin = i * dias_particion + t_inicial
    test_inicio = train_fin
    test_fin = test_inicio + dias_particion

    # Dividir los datos en conjuntos de entrenamiento y prueba
    train_df = df.iloc[train_inicio:train_fin]
    test_df = df.iloc[test_inicio:test_fin]

    # Prepara los datos para el modelo LightGBM
    X_train = train_df.drop(['ds', 'y'], axis=1)
    y_train = train_df['y']
    X_test = test_df.drop(['ds', 'y'], axis=1)
    y_test = test_df['y']

    # Entrenar el modelo con los mejores hiperparámetros encontrados
    lgb_train = lgb.Dataset(X_train, y_train)
    lgb_eval = lgb.Dataset(X_test, y_test, reference=lgb_train)
    model = lgb.train(best_params, lgb_train, num_boost_round=1000,
valid_sets=lgb_eval, early_stopping_rounds=100, verbose_eval=False)

    # Realizar las predicciones y almacenarlas
    predictions = model.predict(X_test, num_iteration=model.best_iteration)

    # Convertir las predicciones en un DataFrame
```

```
    predictions_df = pd.DataFrame(predictions, columns=['yhat'])
    predictions_df['ds'] = test_df['ds'].values

    # Unir predicciones con datos reales
    results = pd.concat([test_df.set_index('ds')['y'],
predictions_df.set_index('ds')['yhat']], axis=1, join='inner')
    results.columns = ['y', 'yhat']

    results['y'] = results['y'].replace(0,0.1) # Cambia lo valores en cero por 0.1 para
evitar errores en el calculo del MAPE

    # Almacenar resultados
    predicciones.append(results)

predicciones_totales = pd.concat(predicciones)
predicciones_totales = predicciones_totales.reset_index()
predicciones_totales['DATASET'] = DATASETN
predicciones_totales['PERIODO'] = PERIODO
predicciones_totales['ZONA'] = ZONA
predicciones_totales['PREDICCIONES'] = PREDICCIONES
predicciones_totales['VENTANA'] = VENTANA
predicciones_totales['ALGORITMO'] = ALGORITMO

# Calcular las métricas de evaluación
mape = mean_absolute_percentage_error(predicciones_totales['y'],
predicciones_totales['yhat'])
mse = mean_squared_error(predicciones_totales['y'], predicciones_totales['yhat'])
rmse = mse ** 0.5
promedio = predicciones_totales['y'].mean()
y_min = predicciones_totales['y'].min()
y_max = predicciones_totales['y'].max()
rango = y_max - y_min

# Almacenar resultados
mape_resultados.append(mape)
mse_resultados.append(mse)
rmse_resultados.append(rmse)
promedio_resultados.append(promedio)
y_min_resultados.append(y_min)
y_max_resultados.append(y_max)
rango_resultados.append(rango)
num_leaves_resultados.append(best_params['num_leaves'])
learning_rate_resultados.append(best_params['learning_rate'])
feature_fraction_resultados.append(best_params['feature_fraction'])
bagging_fraction_resultados.append(best_params['bagging_fraction'])
bagging_freq_resultados.append(best_params['bagging_freq'])
```

```
        # Crear tabla de resultados
        resultados_tabla = pd.DataFrame({
            'MAPE': mape_resultados,
            'MSE': mse_resultados,
            'RMSE': rmse_resultados,
            'Promedio': promedio_resultados,
            'Minimo': y_min_resultados,
            'Maximo': y_max_resultados,
            'Rango': rango_resultados,
            'num_leaves': num_leaves_resultados,
            'learning_rate': learning_rate_resultados,
            'feature_fraction': feature_fraction_resultados,
            'bagging_fraction': bagging_fraction_resultados,
            'bagging_freq': bagging_freq_resultados
        })
        #resultados_tabla.index = np.arange(1, particiones+1)
        resultados_tabla['DATASET'] = DATASETN
        resultados_tabla['PERIODO'] = PERIODO
        resultados_tabla['ZONA'] = ZONA
        resultados_tabla['PREDICCIONES'] = PREDICCIONES
        resultados_tabla['VENTANA'] = VENTANA
        resultados_tabla['ALGORITMO'] = ALGORITMO

        """
```

Esta función alamacena los resultados obtenidos:

```
        resultados_LGB = """

        predicciones_finales_LGB.append(predicciones_totales)
        resultados_finales_LGB.append(resultados_tabla)
        """
```

7.1.3.3 Long Short-Term Memory (LSTM):

Codigo de iteración para correr todas las zonas en todos los tiempos para LSTM

```
        predicciones_totales_T = []
        resultados_tabla_T = []

        for index, row in dataset.iterrows():
            DATASETN = row['DATASET']
            PERIODO = row['PERIODO']
            ZONA = row['ZONA']
            PREDICCIONES = row['PREDICCIONES']
            VENTANA = row['VENTANA']
```

```
    DATASET = globals()[DATASETN]
    ALGORITMO = 'LSTM'
    seq_length = 4

  # PREPROCESAMIENTO DE DATOS

    exec(preprocesamiento_datos)
    exec(dividir_dataset)

  # OPTIMIZAR HIPERPARÁMETROS Y EJECUTAR CODIGO

    exec(optimizar_hiperparametros_LSTM)  # Optimizar parametros


  # EJECUTAR CODIGO

    exec(ejecutar_modelo_LSTM)

  # ALMACENAR RESULTADOS

    exec(resultados_LSTM)

# Unir todos los resultados
predicciones_finales_LSTM = pd.concat(predicciones_totales_T)
resultados_finales_LSTM = pd.concat(resultados_tabla_T)
```

El codigo anterior hace que el modelo de LSTM se ejecute para cada dataset, realice un preprocesamiento adicional a los datos para que puedan ser leidos por el modelo, busque los hiperparametros optimizados y realice las predicciones en el horizonte de tiempo indicado y almacene los resultados.

A continuación se muestran las 5 funciones que ejecuta en cada iteración:

Esta función realiza preprocesamiento a los datos:

```
preprocesamiento_datos = """
#PERIODO = 'D'
df = DATASET

# La columna ds contiene las fechas y debe convertirse en el indice del dataframe
try:
   df['ds'] = pd.to_datetime(df['ds'])
   df.set_index('ds', inplace=True)
   df = df.reset_index()
except:
   df = df.reset_index()
```

```
filter = df['ds']>='2002-01-01'
df = df[filter]

# Obtener nombres de todas las columnas
todas_las_columnas = df.columns.tolist()

# Eliminar las columnas 'ds' y 'y' de la lista de nombres
todas_las_columnas.remove('ds')
todas_las_columnas.remove('y')

# Definir el orden de las columnas
nuevas_columnas = ['ds', 'y'] + todas_las_columnas

# Reordenar el DataFrame
df = df[nuevas_columnas]

# Selecciona solo las columnas del tipo de datos que cumplan la condición del tipo
de datos adecuado para LSTM
columnas_utiles = df.select_dtypes(include=["int", "float", "bool"]).columns.tolist()

# Elimina todas las columnas que no cumplen la condición
columnas_no_utiles = set(df.columns) - set(columnas_utiles) - set(["ds"])
df.drop(columnas_no_utiles, axis=1, inplace=True)

# Borrar columnas que no deberian estar
columnas_a_eliminar = ['Total', 'Zona 12', 'total', 'Zona 347', 'Desconocido',
'Participacion']

# Elimina solo las columnas que existen en el DataFrame
for columna in columnas_a_eliminar:
    if columna in df.columns:
        df.drop(columna, axis=1, inplace=True)

# Adicionar regresores para mejorar la prediccion del modelo
if PERIODO == 'D':
    df['ds'] = pd.to_datetime(df['ds'])
    df['dia'] = df['ds'].dt.day
    df['semana'] = df['ds'].dt.isocalendar().week
    df['mes'] = df['ds'].dt.month
    #df['año'] = df['ds'].dt.year
    df['diasem'] = df['ds'].dt.dayofweek

elif PERIODO == '4S' or PERIODO == '2S':
    df['ds'] = pd.to_datetime(df['ds'])
    df['mes'] = df['ds'].dt.month
    #df['año'] = df['ds'].dt.year
```

```
    else:
        df['ds'] = pd.to_datetime(df['ds'])
        df['mes'] = df['ds'].dt.month
        #df['año'] = df['ds'].dt.year
        df['semana'] = df['ds'].dt.isocalendar().week

    # Calcular el promedio de los últimos 10 valores 2 periodos atrás para ver el valor
    anterior cuanta desviación tiene de ese promedio
    promedio_10 = df['y'].shift(2).rolling(window=10).mean()

    # Crear la columna 'promedio_10'
    df['promedio_10'] = promedio_10

    # Calcular la diferencia entre el valor anterior y el promedio
    dif_anterior_promedio = df['y'].shift(1) - df['promedio_10']

    # Crear la columna 'dif_anterior_promedio'
    df['dif_anterior_promedio'] = dif_anterior_promedio

    # Reemplazar los valores nulos por ceros
    df.fillna(0, inplace=True)

    """
```

Esta función divide el dataset:

```
    dividir_dataset = """

    # Definir variables de entrada
    periodos = PREDICCIONES * VENTANA
    particiones_sem = periodos + seq_length  # El tamaño de particiones es afectado
    por el seq_lenght

    # Preprocesamiento
    try:
        df['ds'] = pd.to_datetime(df['ds'])
        df.set_index('ds', inplace=True)
    except:
        pass

    # Obtener el número de columnas en el dataset original
    num_columns = df.shape[1]

    # Escalar los datos
    scaler = MinMaxScaler()
    df_scaled = scaler.fit_transform(df)
```

```
# Crear conjuntos de entrenamiento y prueba
train_size = len(df) - periodos
train = df_scaled[:train_size]
test = df_scaled[train_size:]

# Crear secuencias de datos de entrada y salida directamente
x_train = np.array([train[i:i+seq_length] for i in range(len(train)-seq_length)])
y_train = train[seq_length:, 0]

x_test = np.array([test[i:i+seq_length] for i in range(len(test)-seq_length)])
y_test = test[seq_length:, 0]

"""
```

Esta funcion optimiza los hiperparametros:

```
optimizar_hiperparametros_LSTM = """

# Función de optimización
def objective(trial: Trial):
    # Hiperparámetros a optimizar
    units_1 = trial.suggest_categorical("units_1", [32, 64, 128, 256, 512])
    units_2 = trial.suggest_categorical("units_2", [32, 64, 128, 256, 512])
    activation_ = trial.suggest_categorical("activation_", ['tanh', 'relu'])
    dropout_ = trial.suggest_categorical("dropout_", [0, 0.01, 0.1, 0.2, 0.3, 0.5])
    optimizer_ = trial.suggest_categorical("optimizer_", ['Adam', 'Adamax'])
    learning_rate_ = trial.suggest_categorical("learning_rate_", [0.01, 0.001, 0.0001])

    # Crear el modelo LSTM
    model = Sequential()
    model.add(LSTM(units=units_1, activation=activation_, return_sequences=True,
input_shape=(x_train.shape[1], x_train.shape[2])))
    model.add(Dropout(dropout_))
    model.add(LSTM(units=units_2, activation=activation_,
return_sequences=False))
    model.add(Dropout(dropout_))
    model.add(Dense(1))

    # Compilar y ajustar el modelo
    if optimizer_ == 'Adam':
        optimizer = Adam(learning_rate=learning_rate_)
    elif optimizer_ == 'Adamax':
        optimizer = Adamax(learning_rate=learning_rate_)
    model.compile(optimizer=optimizer, loss='mean_squared_error')
    early_stop = EarlyStopping(monitor='val_loss', patience=25)
    history = model.fit(x_train, y_train, epochs=100, batch_size=32,
validation_split=0.1, callbacks=[early_stop], verbose=0)
```

```
    # Entrena el modelo y realiza predicciones
    y_pred = model.predict(x_test)

    # Desescalar los datos
    num_columns = x_test.shape[2]
    y_test_descaled = scaler.inverse_transform(np.concatenate((y_test.reshape(-1, 1),
np.zeros((len(y_test), num_columns - 1))), axis=1))[:, 0]
    y_pred_descaled = scaler.inverse_transform(np.concatenate((y_pred,
np.zeros((len(y_pred), num_columns - 1))), axis=1))[:, 0]

    # Calcular las métricas de evaluación
    mape = mean_absolute_percentage_error(y_test_descaled, y_pred_descaled)

    return mape

# Configurar Optuna
study = optuna.create_study(direction="minimize", sampler=TPESampler())
study.optimize(objective, n_trials=50, timeout=8000)  # n_trials: número de
iteraciones, timeout: tiempo máximo en segundos

# Extraer los mejores hiperparámetros
best_params = study.best_trial.params
units_1 = best_params["units_1"]
units_2 = best_params["units_2"]
activation_ = best_params["activation_"]
dropout_ = best_params["dropout_"]
optimizer_ = best_params["optimizer_"]
learning_rate_ = best_params["learning_rate_"]

"""
```

Esta funcion ejecuta el modelo con los hiperparametros encontrados:

```
    ejecutar_modelo_LSTM = """

    # Crear el modelo LSTM
    model = Sequential()
    model.add(LSTM(units=units_1, activation=activation_, return_sequences=True,
    input_shape=(x_train.shape[1], x_train.shape[2])))
    model.add(Dropout(dropout_))
    model.add(LSTM(units=units_2, activation=activation_, return_sequences=False))
    model.add(Dropout(dropout_))
    model.add(Dense(1))

    # Compilar y ajustar el modelo
    if optimizer_ == 'Adam':
        optimizer = Adam(learning_rate=learning_rate_)
```

```
elif optimizer_ == 'Adamax':
    optimizer = Adamax(learning_rate=learning_rate_)
model.compile(optimizer=optimizer, loss='mean_squared_error')
early_stop = EarlyStopping(monitor='val_loss', patience=25)
history = model.fit(x_train, y_train, epochs=100, batch_size=32,
validation_split=0.1, callbacks=[early_stop])

# Entrena el modelo y realiza predicciones
y_pred = model.predict(x_test)

# Desescalar los datos
num_columns = x_test.shape[2]
y_test_descaled = scaler.inverse_transform(np.concatenate((y_test.reshape(-1, 1),
np.zeros((len(y_test), num_columns - 1))), axis=1))[:, 0]
y_pred_descaled = scaler.inverse_transform(np.concatenate((y_pred,
np.zeros((len(y_pred), num_columns - 1))), axis=1))[:, 0]

# Almacenar resultados
results = pd.DataFrame({'y': y_test_descaled, 'yhat': y_pred_descaled})

 # Cambia los valores en cero por 0.1 para evitar errores en el cálculo del MAPE
results['y'] = results['y'].replace(0, 0.1)

# Crear lista para almacenar las predicciones
predicciones = []
predicciones.append(results)

predicciones_totales = pd.concat(predicciones)
predicciones_totales = predicciones_totales.reset_index()
predicciones_totales['DATASET'] = DATASETN
predicciones_totales['PERIODO'] = PERIODO
predicciones_totales['ZONA'] = ZONA
predicciones_totales['PREDICCIONES'] = PREDICCIONES
predicciones_totales['VENTANA'] = VENTANA
predicciones_totales['ALGORITMO'] = ALGORITMO

# Calcular las métricas de evaluación
mape = mean_absolute_percentage_error(predicciones_totales['y'],
predicciones_totales['yhat'])
mse = mean_squared_error(predicciones_totales['y'], predicciones_totales['yhat'])
rmse = mse ** 0.5
promedio = predicciones_totales['y'].mean()
y_min = predicciones_totales['y'].min()
y_max = predicciones_totales['y'].max()
rango = y_max - y_min

# Almacenar resultados
```

```
mape_resultados = [mape]
mse_resultados = [mse]
rmse_resultados = [rmse]
promedio_resultados = [promedio]
y_min_resultados = [y_min]
y_max_resultados = [y_max]
rango_resultados = [rango]

# Crear tabla de resultados
resultados_tabla = pd.DataFrame({
   'MAPE': mape_resultados,
   'MSE': mse_resultados,
   'RMSE': rmse_resultados,
   'Promedio': promedio_resultados,
   'Minimo': y_min_resultados,
   'Maximo': y_max_resultados,
   'Rango': rango_resultados,
   'Units 1': units_1,
   'Units 2': units_2,
   'Activation': activation_,
   'Dropout': dropout_,
   'Optimizer': optimizer_,
   'Learning Rate': learning_rate_
})
resultados_tabla['DATASET'] = DATASETN
resultados_tabla['PERIODO'] = PERIODO
resultados_tabla['ZONA'] = ZONA
resultados_tabla['PREDICCIONES'] = PREDICCIONES
resultados_tabla['VENTANA'] = VENTANA
resultados_tabla['ALGORITMO'] = ALGORITMO

"""
```

Esta función alamacena los resultados obtenidos:

```
resultados_LSTM = """

predicciones_totales_T.append(predicciones_totales)
resultados_tabla_T.append(resultados_tabla)

"""
```

De esta manera, se realiza la predicción para todos los datasets en cada uno de los 3 algoritmos y se almacenan los resultados y las métricas de evaluación de cada modelo.