Identificación de zonas con alta probabilidad de accidentes de tránsito urbano, mediante la detección inteligente de riesgos de colisión

Monografía de Trabajo de Grado

Juan Jose Paredes Rosero

Santiago Felipe Yepes Chamorro

Director: MSc. Ricardo Salazar C.

Co-Director: Esp. Javier Alexander Hurtado.

Asesor: PhD. Álvaro Pachón de la Cruz.

Universidad del Cauca

Facultad de Ingeniería Electrónica y Telecomunicaciones

Departamento de Telemática

Línea de Investigación:

Aplicaciones y Servicios sobre internet

Popayán, mayo 2022

Identificación de zonas con alta probabilidad de accidentes de tránsito urbano, mediante la detección inteligente de riesgos de colisión

Juan Jose Paredes Rosero

Santiago Felipe Yepes Chamorro

Monografía de Trabajo de Grado Presentado a la Facultad de Ingeniería Electrónica y de Telecomunicaciones de la Universidad del Cauca para obtener el título de:

Ingeniero en Electrónica y Telecomunicaciones

Director: MSc. Ricardo Salazar C.

Co-Director: Esp. Javier Alexander Hurtado.

Asesor: PhD. Álvaro Pachón de la Cruz.

Universidad del Cauca

Facultad de Ingeniería Electrónica y Telecomunicaciones

Departamento de Telemática

Línea de Investigación:

Aplicaciones y Servicios sobre internet

Popayán, mayo 2022

A mi padre, Luis Antonio, por ser la motivación y apoyo en todos los momentos de mi vida, por sus enseñanzas y entrega incondicional. A mi hermano, Luis Alberto, por su sabiduría, cuidado e inspiración. A mi amor, Daniela, por su confianza, por impulsar mis sueños y ser mi compañía a lo largo de este trayecto. A mis tíos y primos por sus deseos, apoyo, cariño y afecto. A mis amigos por ser parte de mi formación, especialmente a mi compañero, Santiago, por ser parte de cada uno de mis proyectos. Este trabajo es por y para Ustedes. ¡Gracias!

Juan Jose Paredes Rosero

A Dios y la Santísima Virgen por ser mi principal fuente de fortaleza espiritual, por su constante ayuda y protección en todos los ámbitos de mi vida. A mi padre (en el cielo) y a mi madre, por su amor incondicional, por sus constantes sacrificios, entrega y arduo trabajo, por motivarme a seguir siempre adelante y porque a través de sus enseñanzas y educación me han forjado como la persona que soy. A mis hermanos, por su amor, apoyo, consejos y compañía. A mis padrinos, madrinas y tías por su cariño y apoyo brindado desde mi infancia. A mi compañero de trabajo de grado, Juan José, por su dedicación, por compartir sus conocimientos, por su ayuda incondicional, sobre todo en los momentos más difíciles y por hacer de este proceso una experiencia mucho más amena. A todos Ustedes, desde el fondo de mi corazón, ¡Muchas Gracias!

Santiago Felipe Yepes Chamorro

Agradecimientos

Expresamos nuestros más sinceros agradecimientos a nuestro director el MSc. Ricardo Salazar Cabrera, por guiar el desarrollo del presente trabajo durante todo este tiempo a través de sus correcciones, recomendaciones y consejos. Agradecemos su comprensión ante los diferentes inconvenientes que tuvimos que atravesar, su dedicación, apoyo, paciencia, enseñanzas y conocimientos compartidos a partir de su experiencia.

Igualmente agradecemos al PhD. Álvaro Pachón de la Cruz y PhD. Juan Manuel Madrid Molina por su disposición, aportes y contribuciones en la elaboración del articulo generado a partir de este trabajo.

A todas las personas que se vieron interesados e involucrados en el desarrollo de este trabajo. Especialmente, agradecemos a Helbert Julian Hernández Velazco y German Darío Velazco Rojas por brindar su tiempo y apoyo en los procesos de recolección de datos.

A toda la comunidad *Open-Source*, especialmente a los equipos de trabajo que han contribuido de manera desinteresada en el desarrollo de las tecnologías usadas en el presente proyecto, ya que sin ellas no pudo haberse logrado.

También agradecemos a nuestra Alma Mater, la Universidad del Cauca, especialmente a todos los profesores que durante estos años contribuyeron de una u otra forma en nuestra formación como profesionales y nuestro crecimiento personal.

Por último, pero no menos importante, agradecemos a nuestros familiares, quienes de forma incondicional nos han brindado su apoyo emocional, espiritual y material a lo largo de todas nuestras vidas, siendo también participes de este logro.

Abstract

Traffic accidents are one of the biggest problems worldwide, being one of the leading causes of death and economic loss in the world. Low and middle-income countries, mainly their intermediate cities, are the most affected by this problem. Despite the different attempts to counteract this problem, no significant results have been achieved. However, advanced data analysis techniques including Machine Learning (ML) have gained strength in this area in recent years. Unfortunately, the transit authorities of intermediate cities in developing countries have insufficient, incomplete, or simply non-existent data. This situation does not allow ML techniques to be used for accident analysis. Therefore, the objective of this work is to review the different alternatives that help in data collection and the creation of intelligent solutions related to the detection of possible traffic accidents.

This work proposes a system to identify areas with high probability of traffic accidents by means of the near-crash theory. The construction of the system considered the following five development phases: a) Identification of devices, kinematic variables and algorithms that help characterize near-crash events; b) Construction of an architecture for the development of the system prototype, considering reference Intelligent Transportation Systems (ITS) architectures and other similar related architectures; c) Design and development of an Intelligent collision risk detection system prototype (ICRDS), including a data collection module (Hardware), an intelligent information analysis module (Software) and an ML model (which considered algorithms: Support Vector Machine, Decision Tree and Random Forest); d) Design and development of field tests in the context of an intermediate city of a developing country, with which a data set of kinematic variables measured in vehicles was built; e) Validation of the performance of the system and the classifier algorithm through the field tests. The results of the implementation of the system prototype were satisfactory, allowing to detect near-crashes from an ML model generated with the RF algorithm. The average performance obtained in each of the metrics of the ML model was greater than 0.95. In addition, with the validation tests performed, it was possible to identify areas with a high accident rate in the city chosen as a case study, demonstrating that there is a relationship between the areas where the greatest number of near-crashes occurred and the areas with the most frequent accident rates.

Resumen

Los accidentes de tránsito son uno de los mayores problemas a nivel mundial, siendo una de las principales causas de muerte y pérdidas económicas en el mundo. Los países en desarrollo, principalmente sus ciudades intermedias, son algunas de las más afectadas por esta problemática. A pesar de los diferentes intentos por contrarrestarla, no se han logrado resultados significativos; sin embargo, técnicas avanzadas de análisis de datos o *Machine Learning* (ML) han tomado fuerza en esta área en los últimos años. Desafortunadamente, las autoridades de tránsito de ciudades intermedias de países en desarrollo cuentan con datos insuficientes, incompletos o simplemente son inexistentes. Esta situación no permite que técnicas de ML fueran usadas para el análisis de accidentalidad. Por lo tanto, el objetivo de este trabajo es revisar las diferentes alternativas que ayuden en la recolección de datos y la creación de soluciones inteligentes relacionadas con la detección de posibles accidentes de tránsito.

Este trabajo propone un sistema que permite identificar zonas con alta probabilidad de accidentes de tránsito mediante la teoría de *near-crash* (o casi-choques). La construcción del sistema consideró las siguientes cinco fases de desarrollo: a) Identificación de los dispositivos, variables cinemáticas y algoritmos que ayuden a caracterizar eventos de *near-crash*; b) Construcción de una arquitectura para el desarrollo del prototipo del sistema, considerando arquitecturas de sistemas inteligentes de transporte (ITS) de referencia y otras arquitecturas similares relacionadas; c) Diseño y desarrollo de un prototipo del sistema de detección inteligente de riesgo de colisión (SDIRC), incluyendo un módulo de recolección de datos (*Hardware)*, un módulo de análisis inteligente de la información (*Software*) y un modelo de ML (que consideró los algoritmos: *Support Vector Machine, Decision Tree* y *Random Forest);* d) Diseño y desarrollo de pruebas de campo en el contexto de una ciudad intermedia de un país en desarrollo, con las cuales se construyó un conjunto de datos (*data set*) de variables cinemáticas medidas en vehículos; e) Validación del funcionamiento del sistema y del algoritmo clasificador mediante las pruebas de campo. Los resultados de la implementación del prototipo del sistema fueron satisfactorios, permitiendo detectar *near-crashes* a partir de un modelo de ML generado con el algoritmo de RF. El rendimiento medio obtenido en cada una de las métricas del modelo de ML fue superior a 0.95. Además, con las pruebas de validación realizadas se logró identificar zonas de alta accidentalidad en la ciudad escogida como caso de estudio, demostrando que existe una relación entre las zonas donde ocurrieron mayor número de *near-crashes* y las zonas de accidentalidad más frecuente.

Tabla de Contenido

Lista de figuras

Lista de tablas

Lista de acrónimos

AAD	Análisis y aprendizaje de datos.
ABS	*Anti-Look Braking (*Sistema Antibloqueo de Ruedas).
Acc	*Accuracy* (Precisión).
Acel	Aceleración.
ADA	Administrador de datos archivados.
ADR	Almacenamiento de datos recolectados.
AII	Análisis inteligente de la información.
ANN	*Artificial Neural Network* (Red neuronal artificial).
ANSV	Agencia Nacional de Seguridad Vial.
ARC-IT	*Architecture Reference for Cooperative and Intelligent Transportation.*
AT	Accidentes de tránsito.
AUC	*Area Under Curve* (Área Bajo la Curva).
BN	*Bayesian Network* (Red Bayesiana).
CAN	*Controller Area Network* (Controlador de Red de Zona).
CART	*Classification and Regression Trees* (Árboles de clasificación y regresión).
CNN	*Convolutional Neural Network* (Red Neuronal Convolucional).
CRISP-DM	*Cross Industry Standard Process for Data Mining* (Proceso Estándar de la Industria para la Minería de Datos).
CSV	*Comma separated values* (Valores separados por coma).
DANE	Departamento Administrativo Nacional de Estadística.
DC	**"***Data set"* central.
DT	*Decision Tree* (Árbol de Decisión).
DUV	Determinación de la ubicación del vehículo.
EBV	Equipo a bordo del vehículo.
ECU	*Engine Control Unit* (Unidad de control de motor).
ENU	Este, Norte y arriba**.**
Fcm	Fuerza de campo magnético.
Fn	Falsos negativos.
Fp	Falsos positivos.
FPD	Filtro y procesamiento de datos.
FRAME	*European Intelligent Transport Systems Framework Architecture*.
GPS	*Global Positioning System* (Sistema de Posicionamiento Global).
HW	*Hardware.*
IA	Inteligencia artificial.
IDE	Entorno de desarrollo integrado.
IMU	*Inertial Measurement Unit* (Unidad de Medición Inercial).
ICRDS	*Intelligent Collision Rick Detection System.*
IoT	*Internet of Things* (Internet de las cosas).
ITS	*Intelligent Transportation System* (Sistema de transporte inteligente).
KNN	*K nearest neighborhoods* (K vecinos más cercanos).

Lat	Latitud.
LIDAR	*Laser Imaging Detection and Ranging* (Medición y Detección de Objetos Mediante Láser).
Lon	Longitud.
LR	*Logistic Regression* (regresión logística).
LSTM	*Long short-term Memory* (Memoria larga a corto plazo).
ML	*Machine Learning* (Aprendizaje Automatizado).
MLP	*Multilayer perceptron* (Perceptrón multicapa).
NB	*Naive Bayes.*
ND	*Naturalistic driving* (Conducción naturalista).
NED	Norte, Este y Abajo.
OBD	*On Board Diagnostic* (Diagnostico a Bordo).
OMS	Organización Mundial de la Salud.
OSI	*Open Systems Interconnection* (Sistemas abiertos interconectados)
PIB	Producto Interno Bruto.
PMBOK	*Guide to the Project Management Body of Knowledge* (Guía para los fundamentos de dirección de proyectos).
PMI	*Product Management Institute* (Instituto de Gestión de Proyectos).
RBFN	*Radial basis Function Network* (Red de función de base radial).
RCV	Registro cinemático del Vehículo.
RD	Recolección de datos.
RF	*Random Forest* (Bosques Aleatorios).
RNN	Red Neuronal Recurrente.
ROC	*Receiver Operating Characteristics* (Características Operativas del Receptor).
SDA	Sistema de datos archivados.
SDIRC	Sistema de Detección Inteligente de Riesgos de Colisión.
SDK	Software Development Kit (Kit de desarrollo de software).
Seg	Segundos.
SHRP2	*Strategic Highway Research Program 2* (Segundo Programa de Investigación en Carreteras).
SPR	Sistema de presentación de resultados.
STCL-Net	*Spatiotemporal convolutional Long Short-term Memory Network.*
SVM	*Support Vector Machines* (Maquina de Vectores de Soporte).
SW	*Software.*
Ts	*Timestamp* (Marca de tiempo).
UI	*User Interface* (Interfaz de Usuario).
Vang	Velocidad Angular.
Vel	Velocidad.
Vn	Verdaderos negativos.
Vp	Verdaderos positivos.

Capítulo 1

1. Introducción

Este capítulo presenta el problema planteado, los objetivos abordados, las contribuciones relevantes del trabajo y las metodologías utilizadas para su desarrollo. Además, la información sobre el sometimiento de un artículo (de revisión relacionado con la investigación) a una revista relacionada con el área del transporte. Al finalizar, se describe la estructura del documento de monografía.

1.1. Planteamiento del problema

Los accidentes de tránsito (AT) son una de las problemáticas que actualmente enfrenta la población mundial. En el año 2016, hubo un aproximado de 1.35 millones de AT en el mundo, según la Organización Mundial de la Salud (OMS) [1], ingresando en las 10 principales causas de muerte en el mundo. Los AT también han costado el 3% de su Producto Interno Bruto (PIB) a muchos países [1]. Las causas principales de AT son el crecimiento del índice de motorización (cantidad de autos movilizándose en un país por cada mil habitantes) y la expansión de la población mundial; aunque, hay otros factores que influyen en el incremento del índice de accidentes (cantidad de accidentes por cada mil habitantes), como la impericia del conductor, omisión de normas, falta de señalización, exceso de velocidad, vehículos que no cumplen las normas o el mal estado de las vías [1].

En Colombia, los AT también son un problema considerable. En el año 2018, la tasa de mortalidad por AT fue de 18,5 muertes por cada 100.000 habitantes, siendo Colombia la séptima a nivel latinoamericano (esta cifra es una tasa de mortalidad, la que indica cuantos casos de muerte se presentan por cierta causa, en cierta cantidad de población) [2].

De acuerdo con [3], entre los tipos de AT más conocidos (choque, volcamiento, caída de ocupante y atropello) los dos más frecuentes en Colombia son: el choque entre vehículos y el atropello, tanto en los casos en los que se presentan muertes, como en los que solo hay lesiones [3]. En los casos donde hay muertes, los dos tipos de AT tienen un porcentaje de 57,22% (choque) y 28,16% (atropello). En los casos donde sólo hay lesiones los porcentajes son aún mayores: 70,48% (choque) y 19,72% (atropello) [3]. El análisis espacial de AT en Colombia acentúa que hay mayor riesgo de accidentalidad en el área urbana que en el área rural (ya que en el área urbana ocurren el 54,5% de las muertes por AT y el 94,09% de las lesiones por AT) [3]. Lo anterior evidencia que los choques en la zona urbana de una ciudad colombiana son un aspecto de alta relevancia en cuanto a AT.

En Popayán (ciudad capital del Departamento del Cauca), el asunto de los AT también es un tema preocupante. En la ciudad ocurren el 69,91% de los casos de AT reportados en el Cauca. En esta ciudad, al igual que los datos del país, los choques son la principal causa de accidentes (89,32%) [4]. Factores como cruzar sin observar y adelantamientos son

algunos de los más significativos, pero muchas causas de estos choques no son aclaradas. A esta situación se suman los problemas de infraestructura vial que presenta la ciudad, como son: el mal estado de las vías, estrechez de los carriles, falta de señalización, entre otros [4]. Estos datos evidencian la importancia de conocer las causas y factores de ocurrencia de esta clase de choques y su ubicación, con el fin de tomar medidas que busquen reducir la cantidad de accidentes en la región.

Los gobiernos a nivel mundial han intentado contrarrestar la problemática con la implementación de políticas públicas orientadas a prevenir los AT y mitigar sus impactos negativos, lo cual se evidencia en el Plan Mundial de Seguridad Vial (2011-2020) [5]. En el caso del gobierno colombiano, fue llevada a cabo la definición del Plan Nacional de Seguridad Vial en el año 2011 y la creación de la Agencia Nacional de Seguridad Vial (ANSV) en el año 2013. El objetivo de dicha agencia es propiciar medidas de tipo administrativas, educativas y operativas que reduzcan el número de AT y por ende el número de lesionados y muertes causadas por los AT [5], [6].

Algunas de esas medidas han aprovechado los beneficios de las tecnologías emergentes de los últimos años, logrando reducciones en la siniestralidad vial, particularmente desde el año 2017, pasando de 194.014 siniestros en dicho año a 180.373 siniestros en el año 2019 [4]. Como ejemplo de estas medidas tecnológicas implementadas se encuentran: el control de velocidad y el cumplimiento de las normas de tránsito de formas automatizadas, utilizando cámaras de alta resolución. Sin embargo, el número de víctimas fatales no ha disminuido desde el año 2012, en el cual se presentaron 6.136 [4]. Para intentar seguir reduciendo el número de AT y víctimas fatales, es necesario diseñar nuevos métodos que enfrenten esta problemática. Por lo cual, el aprovechamiento de los avances tecnológicos que faciliten la toma de decisiones, a partir del entendimiento y explotación de los datos generados por diferentes entidades, es fundamental [7].

El análisis de datos ha tomado fuerza en los últimos años para identificar los factores de riesgo o proponer soluciones a esta problemática relacionada con los AT [8], [9]. Muchas aplicaciones han sido desarrolladas empleando diferentes tecnologías pertenecientes a campos como: el Internet de las cosas (*Internet of Things* o IoT), la minería de datos, y el aprendizaje de máquina (*Machine Learning* o ML). La tecnología IoT permite recolectar datos de conducción o de tráfico en determinadas zonas. Mientras que la minería de datos y la tecnología ML facilitan el análisis y procesamiento de los conjuntos de datos previamente recolectados e incluso predecir este tipo de siniestros [9].

En los últimos años, el ML ha sido una herramienta ampliamente usada, no solo en los modelos predictivos/explicativos de los riesgos de choque; sino en procesos de análisis descriptivos de los mismos [10]; sin embargo, cuando un modelo de ML es empleado, la calidad y cantidad de datos juegan un papel importante en la obtención de resultados óptimos. En el caso de Popayán, al igual que en varias ciudades intermedias del país, el proceso de obtención de datos de AT puede orientarse, en principio, a la consulta de información ofrecida por entidades como el Departamento Administrativo Nacional de Estadística (DANE), la ANSV o las plataformas de datos abiertos del gobierno. En muchos casos estos datos no son suficientes para extraer la información esperada (como, por

ejemplo: la causa del accidente y su ubicación exacta) o simplemente no existen. En este último caso, el proceso de recolección de datos puede implicar, para estas entidades, largos periodos de tiempo manteniendo la expectativa de ocurrencia de AT. Por lo tanto, es necesario recurrir a métodos que permitan la detección de posibles accidentes de forma ágil y precisa.

Uno de estos métodos para obtener la información necesaria es la conducción naturalista (ND, por la sigla en inglés de *Naturalist Driving*). Este es un método para la recolección de datos que proporciona información sobre el AT [11]. ND hace uso, principalmente, de los datos cinemáticos de un vehículo (por ejemplo: velocidad, aceleración, frenado, etc.), aunque algunos estudios también incluyen otros tipos de datos como video o imágenes según sus necesidades. En la ND, surge el concepto de *near-crashes* (casi-choques o casi-colisiones), estos son eventos sustitutos de los choques, en el cual los factores de riesgo para los AT son más sensibles que en los choques comunes [12]. Por lo tanto, hacer uso de la ND podría ayudar en la construcción de un "***data set"*** conformado por variables vehiculares, recolectadas durante diferentes de viajes donde puedan presentarse posibles AT. Mediante este "data set" podría desarrollarse un modelo de ML capaz de encontrar zonas con alta probabilidad de AT y sus posibles causas.

Los trabajos relacionados y revisados en la literatura presentan una gran variedad de opciones para intentar detectar AT y luego analizar sus posibles causas. Las principales brechas están relacionadas con el contexto de dichos trabajos y la no identificación de zonas con mayor riesgo de AT. Además, la gran mayoría de estos trabajos no han sido realizados en ciudades intermedias de países en desarrollo (como es propuesto en este proyecto) y tan solo uno de ellos se centró en usar los datos para identificar zonas con mayor accidentalidad, adicionalmente, algunos trabajos construyen un "***data set"*** empleando dispositivos y/o sensores que requieren de alto presupuesto, mientras que este proyecto busca realizar dicha recolección con dispositivos que sean viables de adquirir con un presupuesto limitado, adaptándose al contexto de las ciudades intermedias de países en vía de desarrollo.

De acuerdo con lo anterior, para ayudar en la prevención de AT es planteada la siguiente pregunta de investigación: ¿Cómo determinar las zonas de alta probabilidad de AT en un entorno urbano de una ciudad colombiana?

La hipótesis propuesta para dar solución a la pregunta de investigación es la siguiente: Es posible determinar las zonas con alta probabilidad de AT con ayuda de los *near-crashes*, utilizando una aplicación que use modelos de ML para el análisis de datos recolectados mediante pruebas de campo de conducción naturalista.

1.2. Objetivos

1.2.1. Objetivo general

Proponer un sistema de detección de zonas de alta probabilidad de AT basado en conducción naturalista y análisis de datos recolectados en trayectos de vehículos.

1.2.2. Objetivos específicos

- Identificar las variables y dispositivos relevantes asociados a la detección de *near-crashes* y a la recolección de datos en ND
- Desarrollar un prototipo del sistema de detección de zonas de alta probabilidad de AT, incluyendo un módulo de recolección de datos y un módulo de análisis inteligente de la información.
- Construir el "*data set*" de las variables vehiculares seleccionadas, mediante la ejecución de pruebas de campo en la ciudad seleccionada.
- Evaluar el prototipo del sistema propuesto a partir del "*data set*" y el módulo de análisis inteligente de información.

1.3. Contribuciones

El trabajo propuesto fortalece la línea de investigación del Grupo de Ingeniería Telemática relacionada con las aplicaciones y servicios sobre Internet. Concretamente los aportes más relevantes que tendrá el trabajo son:

- Una revisión de literatura para la elección de las variables cinemáticas, dispositivos y algoritmos que mejor caracterizan un *near-crash*.
- El diseño e implementación de un módulo hardware y software que permita, por medio de ciertos sensores, la recolección de datos en trayectos vehiculares en ciertas rutas de una ciudad. El hardware, software, y sensores a utilizar, al igual que los datos a capturar son especificados en la sección 5 del presente documento.
- El diseño e implementación del módulo de análisis de la información recolectada, para la detección de las zonas de alta probabilidad de AT. Este módulo comprende también, la evaluación de algoritmos de ML, a través de la implementación de la metodología CRISP-DM (ver sección 1.4.3). El desempeño de estos algoritmos fue evaluado con las siguientes métricas: *"precision", "recall", "F1-Score", "AUC"*.
- La implementación de todo el prototipo del sistema inteligente para la detección de zonas de alta probabilidad de choque, integrando el módulo encargado de la recolección de datos, y el módulo *software* encargado de analizar la información recolectada en los trayectos de los vehículos.

- La clasificación de zonas con mayor riesgo de ocurrencia de un AT (en la ciudad de Popayán, seleccionada como caso de estudio). Esta información aporta conocimiento sobre la caracterización de los AT en la ciudad de Popayán.
- El diseño de las pruebas de campo para realizar los recorridos de los vehículos, utilizando el módulo de recolección de datos. Incluye entre otros, el número de trayectos, el número de vehículos a utilizar, el número de conductores, las rutas a seguir en los trayectos, el número de repeticiones de cada recorrido.
- Una aplicación móvil y una aplicación hibrida (desarrollada a través de una Raspberry Pi) para la captura de variables cinemáticas de un vehículo.
- Una plataforma web para el análisis de los datos cinemáticos capturados por los dispositivos, incluyendo un modelo de ML para la detección de eventos de *near-crash*.
- Dos "*data set*". El primero presenta datos de maniobras de conducción agresivas, y el segundo presenta datos de conducción naturalista en la ciudad de Popayán.

1.4. Metodología

El desarrollo de los objetivos propuestos fue realizado a través de una serie de actividades, según las directrices detalladas en el Cuerpo de Conocimientos de Gestión de Proyectos (PMBOK), instrumento desarrollado por el Instituto de Gestión de Proyectos (*Project Management Institute* o PMI) [13]. El proyecto fue segmentado en los siguientes 5 paquetes de trabajo (WP, por su nombre en inglés *Work Package*): Identificación de variables y dispositivos relevantes; desarrollo del prototipo del sistema; pruebas de campo y construcción del "*data set*"; análisis y evaluación de resultados; preparación de documentación y otros entregables.

El desarrollo de la aplicación o módulo de análisis inteligente de la información (que hace parte del prototipo del sistema) fue realizado a través de la metodología SCRUM [14], ya que es una metodología de desarrollo ágil, que presenta ciertos beneficios como: la adaptación del proyecto, control de riesgo, incremento de la productividad y mejora en la eficiencia.

La metodología de ND fue utilizada para la recolección de los datos y el diseño de las pruebas de campo.

Por último, la metodología CRISP-DM fue utilizada para el desarrollo del proceso de análisis de datos, sobre el "*data set*" recolectado.

A continuación, son presentadas brevemente cada una de estas metodologías utilizadas en el proyecto (a excepción de lo ya mencionado del PMBOK del PMI).

1.4.1. Metodología SCRUM

SCRUM es definido como un marco de trabajo donde son aplicadas un conjunto de buenas prácticas y roles con el fin de hacer el trabajo más colaborativo y que tenga el mejor resultado posible [14]. Esta metodología es apoyada por tres pilares: transparencia, inspección y adaptación y es dividida en flujos de trabajo cortos con duración fija denominados *Sprints*. Cada *Sprint* debe llevar a cabo tres actividades generales, las cuales deben ser terminadas para considerar a un *Sprint* como completado. Estas son: planificación, ejecución y revisión y retrospectiva.

1.4.2. Metodología para las pruebas de campo, ND

ND es un método usado para la investigación de comportamientos de conducción y AT [12]. Uno de sus focos principales es la recolección de datos a partir de pruebas de campos en condiciones de conducción normales (es decir no usar entornos controlados), estos datos son recolectados a partir de sensores a bordo del vehículo siendo usualmente datos cinemáticos del vehículo [12].

1.4.3. Metodología para el análisis de datos, CRISP-DM

Cross Industry Standard Process for Data Mining (CRISP-DM) es una metodología para el desarrollo de procesos de minería de datos [15], que proporciona un modelo de ciclo de vida para proyectos de análisis de datos. Este modelo contiene seis fases que están relacionadas entre sí. La **Figura 1** presenta las fases de esta metodología [16].

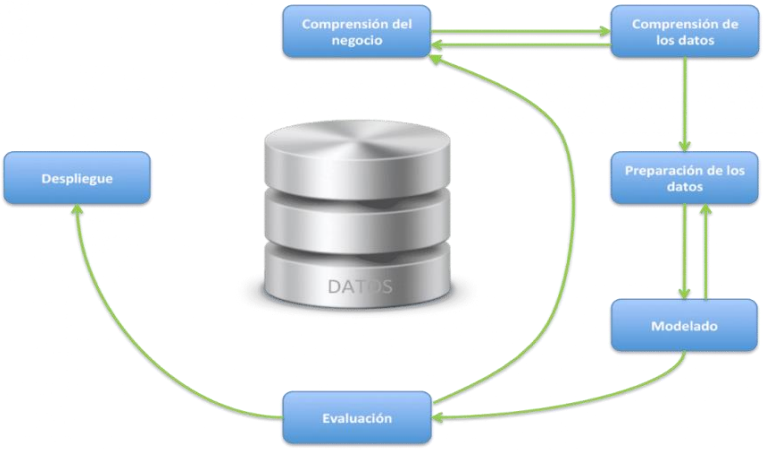

Figura 1. Fases metodología CRISP-DM [16]

1.5. Artículo generado y sometido a revisión

El trabajo presentado en esta monografía fue presentado a la comunidad científica mediante el envío de un artículo de revisión a una revista relacionada con la investigación teórica y experimental en el campo del transporte. El artículo fue enviado a la revista *Journal of Traffic and Transportation Engineering (English Edition)*, la cual está ubicada en el cuartil

dos (Q2) en el *Scimago Journal Rank* (SJR del año 2020), y en la categoría A2 de acuerdo con el listado de homologación de *Publindex* (Min Ciencias, para el 2022).

Autores del articulo: Juan Jose Paredes Rosero, Santiago Felipe Yepes, Ricardo Salazar-Cabrera, Álvaro Pachón de la Cruz, Juan Manuel Madrid Molina.

Titulo: *System for the identification of areas with a high probability of traffic accidents: A review of the best alternatives.*

Revista: *Journal of Traffic and Transportation Engineering (English Edition).*

Estado: Envío inicial a la revista: 2021-11-22. Revisión inicial realizada por los evaluadores, recibida el 2022-04-25. Actualmente se están realizando los ajustes solicitados.

Texto del artículo: Disponible en el **Anexo A. Artículo de revisión.**

1.6. Estructura del documento

A continuación, es descrita la estructura de las demás secciones del presente trabajo:

- Capítulo 1 Introducción, contiene el planteamiento del problema, los objetivos, las contribuciones, la metodología y la publicación.
- Capítulo 2 Marco teórico, describe los conceptos, métodos y tecnologías utilizadas para la formulación del presente trabajo.
- Capítulo 3 Estado del arte, presenta la revisión de la literatura por medio de la metodología PRISMA y los trabajos relacionados con el presente trabajo.
- Capítulo 4 Parámetros y arquitectura del sistema, describe las alternativas y el proceso de selección de las características del módulo de recolección de datos (*Hardware*), y el módulo de análisis inteligente de la información (*Software*) para lograr identificar *near-crashes*.
- Capítulo 5 Diseño y desarrollo del prototipo, este capítulo describe los procedimientos realizados para el desarrollo del prototipo del sistema, "*data set*" y modelo de ML.
- Capítulo 6 Diseño y desarrollo de las pruebas, especifica la forma en que fueron realizadas las pruebas de campo para el entrenamiento y validación del sistema.
- Capítulo 7 Resultados, presenta los resultados obtenidos en el desarrollo del prototipo del sistema, el modelo de ML y las pruebas de campo realizadas en el capítulo anterior.
- Capítulo 8 Conclusiones y trabajos futuros, presenta las conclusiones y aportes del presente documento, además de los trabajos futuros.

Capítulo 2

2. Marco teórico

En este capítulo son presentados los fundamentos teóricos para el desarrollo y funcionamiento del prototipo del sistema propuesto. En primera instancia es descrita la técnica de conducción naturalista, empleada para las pruebas de campo del presente trabajo; en segundo lugar, se explica el concepto de *near-crash* y su relación con los accidentes de tránsito; en tercera instancia la cinemática vehicular junto con la teoría del cuerpo rígido también es descrita validando el uso de dispositivos de recolección de datos cinemáticos para detectar ciertos eventos de conducción; posteriormente la tecnología de Machine Learning, sus tipos y el funcionamiento de los algoritmos empleados son abordados demostrando su importancia y potencial para lograr los objetivos del presente trabajo; finalmente, diferentes técnicas de preprocesamiento de los datos son descritas brevemente debido a su necesidad de uso, especialmente cuando grandes cantidades de datos son manipuladas y su importancia es alta.

2.1. Conducción naturalista

La ND es una técnica de investigación que observa electrónicamente cómo conducen las personas, mediante el uso de una serie de dispositivos de captura de datos [17]. Dicha visión sobre el comportamiento del conductor es obtenida gracias a que son registrados ciertos detalles sobre el conductor, el vehículo y el entorno usando equipos de recolección de datos no intrusivos, durante diferentes viajes cotidianos realizados en circunstancias reales y sin control experimental [18], [19].

Por lo anterior, los resultados arrojados en los estudios de conducción naturalista son realistas. Esto permite observar y analizar las interrelaciones entre el conductor, vehículo, carretera y el tráfico en situaciones de conducción normales, situaciones de conflictos menores y accidentes reales [19]. Por esta razón, la técnica de ND permite tanto la investigación del comportamiento de los conductores, como la investigación de diferentes aspectos de la seguridad vial [12]. Así, según el enfoque del estudio, la información recolectada puede ser útil para reducir los accidentes de tránsito, la carga medioambiental del transporte por carretera, la congestión vehicular, entre otros [19].

De acuerdo con [18], según la ubicación de la recolección de datos, los estudios de ND pueden ser clasificados en dos categorías: estudios basados en vehículos (estudios individuales del conductor) y estudios "*in situ*". En los estudios "*in situ*", los datos de ND son recolectados en un sitio en particular usando cámaras de video estáticas instaladas cerca de las vías de estudio [18]. Sin embargo, según [12], la técnica con la intensión de estudiar la seguridad vial usando cámaras y otros sensores localizados a largo de una vía, es llamada técnica del conflicto de tráfico (*The traffic conflict technique*).

Con los estudios "*in situ*" no es posible examinar el comportamiento del conductor, mientras que, en los estudios basados en vehículos dicho comportamiento puede ser examinado de forma individual usando diferentes dispositivos [18]. Algunos de los dispositivos comúnmente usados son: sistema de posicionamiento global (GPS), unidad de medición inercial (IMU, combinación de acelerómetros y giroscopios), sensores de radar, diagnósticos a bordo (OBD), cámaras de vídeo, dispositivos de seguimiento ocular, etc. [17], [18]. En ocasiones, en este tipo de estudios, los dispositivos o equipos de recolección de datos son instalados en vehículos propios de los conductores voluntarios participantes [19].

Los dispositivos instalados en los vehículos registran de forma continua el comportamiento de conducción. Algunos de estos dispositivos, como la IMU, están orientados a capturar datos cinemáticos precisos, teniendo en cuenta variables como la velocidad, aceleración/desaceleración, posición en el carril, localización, distancias lateral y longitudinal entre vehículos, etc. [18]. Por medio de los datos de estas variables es posible inferir diferentes maniobras de conducción, como excesos de velocidad, frenadas repentinas, cambios de carril, entre otras.

Por otro lado, los dispositivos de seguimiento ocular y las cámaras ubicadas a bordo del vehículo, están principalmente dirigidas a capturar las acciones del conductor como maniobras manuales, movimientos de los ojos, de la cabeza, o distracciones [12]. Esto permite analizar el efecto que tiene la presencia de pasajeros o la realización de tareas secundarias (como uso de celulares para llamadas, chat, el uso de otros dispositivos de entretenimiento, etc.) sobre el comportamiento al volante [18].

Adicionalmente, dependiendo de la complejidad de un estudio de ND también puede ser necesaria la captura de condiciones externas al vehículo, es decir su entorno, por ejemplo, las condiciones climáticas, el tráfico, tipo y estado de las vías, obstáculos, entre otros. Esto lleva a los investigadores a emplear otro tipo de dispositivos como radares, *Light Detection And Ranging* (LIDAR) o cámaras para capturar el exterior del vehículo [19], [20].

Una clasificación adicional de los estudios de ND mencionada en [18] fue realizada según el enfoque de la investigación: estudios de comportamiento basados en conducción normal y en eventos críticos. En los estudios de conducción normal, son registrados continuamente los datos generados por un vehículo. La mayoría de los estudios de conducción naturalista a gran escala realizaron este tipo de recolección de datos. Entre ellos están: *UDRIVE y Australia 300 car naturalistic driving study* [18], SHRP 2 [21], *100 car naturalistic driving study* [22].

Por otra parte, los estudios basados en eventos críticos de seguridad presentan un obstáculo relacionado con las colisiones, ya que son eventos poco frecuentes y aleatorios. Durante la conducción, el conductor rara vez se encuentra con un evento crítico para su seguridad [18]. Aunque en los estudios de ND a gran escala los costos aumenten considerablemente, es muy probable que se detecten algunos accidentes y otros eventos críticos [19]. En este sentido, el número de eventos de interés detectados puede disminuir en estudios con un alcance reducido, como es el caso de esta investigación, siendo los

costos la principal limitante. Para ayudar a alivianar estas limitantes, los investigadores han propuesto el uso de *near-crashes* en las investigaciones de ND, siendo estos últimos eventos mucho más comunes [12], [19].

Las ventajas ofrecidas por la técnica de ND han despertado el interés de investigadores, haciendo que el número de investigaciones y estudios de ND aumente en los últimos años [18]. Una de sus principales ventajas es ofrecer perspectivas más amplias para entender el comportamiento del tráfico en situación normales del día a día. Debido a que los participantes no están involucrados en un experimento, no hay un experimentador presente, ni intervenciones experimentales, y el participante no se preocupa por objetivos que deba adivinar o lograr. Además, brindan la posibilidad de observar conflictos, *near-crashes* o incluso accidentes reales registrados en tiempo real [19].

2.2. Casi choque (*near-crash)*

Los casi choques o casi colisiones (del inglés *near-crash)* surgen como una medida sustituta de los choques habituales en estudios de análisis y seguridad vial. Comúnmente el *near-crash* es definido como un evento donde un vehículo efectúa una acción rápida y evasiva que impide una colisión, esta puede ser un choque entre un vehículo, peatón o cualquier obstáculo que pueda causar un AT [21].

Para considerar una medida sustituta apta para el remplazo de los AT es necesario satisfacer los siguientes dos principios: el primero, que los factores causales sean equiparados a los de un AT y el segundo, que la relación de frecuencia se asocie con la de los AT. Los *near-crashes* cumplen con ambos principios, por lo tanto, los resultados de los estudios que usan *near-crash* no difieren de los resultados que únicamente usan AT como eventos de interés [11].

Una de las razones por la que muchos estudios usan los *near-crashes* como medida sustituta, se encuentra en el aumento en la frecuencia de aparición de eventos de interés en los estudios de seguridad vial. De esta manera, usar eventos de *near-crash* permite generar aproximaciones reales en la evaluación del riesgo de AT en casos donde: los datos de AT liberados por algunas entidades gubernamentales no son suficientes o no existen, las muestras de accidentes en experimentos de ND son escasas y cuando los datos no existen debido a que los análisis son realizados en instalaciones viales nuevas. [12], [17].

Como se menciona en la sección 2.1, los eventos de *near-crash* y choques cuentan con un comportamiento aleatorio que dificulta definir los factores causales que contribuyen a su aparición. Debido a esto, en estudios viales es necesario incluir una combinación de varios factores causales; aunque esto trae ciertas ventajas, como aumentar la probabilidad de encontrar verdaderos *near-crashes*; también genera ciertas desventajas, como aumentar la complejidad en la obtención de datos [12]. Cabe aclarar que los factores causales de un evento de choque son los mismos que los de un *near-crash*, la diferencia se encuentra en el resultado final, siendo este último la evasión del accidente [21].

Entre los factores causales del *near-crash* son encontrados factores viales y ambientales, como pueden ser: el estado de las vías, intersecciones, densidad del tráfico, cruces o el clima. Estos elementos son considerados como criterios de ámbito cualitativo. Por otro lado, los factores cinemáticos presentados en el vehículo definen el apartado cuantitativo de los eventos de *near-crash* [21].

Las variaciones cinemáticas que presenta un vehículo en eventos de *near-crash* dependen de la ejecución de una acción evasiva. Realizar este tipo de acción en el vehículo rompe con comportamientos habituales en la aceleración, frenada, o giros que normalmente tiene el vehículo [12]. Cada uno de estos eventos involucra diferentes comportamientos de las variables cinemáticas, que en conjunto pueden caracterizar de forma adecuada un evento de *near-crash* [22], [23].

En estudios de pequeña escala donde resulta difícil obtener datos del entorno y la probabilidad de que se presente un choque es demasiado baja, para obtener eventos de interés, usar el *near-crash* como medida sustituta, por medio de la cinemática vehicular, ayuda a identificar posibles AT que podrían presentarse en las zonas de estudio donde la técnica de ND es aplicada. [12].

2.3. Cinemática vehicular

En la actualidad, hay disponibilidad de una gran multitud de instrumentos que permiten la detección de *near-crashes*, estos pueden ser: sensores de movimiento, sensores de posición, dispositivos GPS, cámaras, sensores de proximidad, fotómetros, radares, etc. [24]. Cada instrumento aporta conocimiento en la detección de eventos de riesgo, pero la literatura ha logrado identificar que la dinámica vehicular y el posicionamiento pueden ser usados para abordar investigaciones de ND [20], [25].

De acuerdo con [26], el modelo dinámico que define el movimiento de un vehículo genera cierto conocimiento que no es relevante para casos de estudios de seguridad vial. Por lo tanto, el movimiento de un vehículo puede definirse como una unidad, ignorando la estructura interna del mismo. De esta manera, [26] expone la teoría del cuerpo rígido para definir la dinámica del vehículo, esta teoría explica que el movimiento de un cuerpo rígido en un espacio tridimensional puede abstraerse a través de dos tipos de movimiento básicos: traslación y rotación. Estos movimientos pueden expresarse a través de medidas cinemáticas de velocidad, aceleración, velocidad angular y fuerza magnética; estas medidas (a excepción de la velocidad) pueden descomponerse en 3 ejes “X”, “Y” y “Z”.

El avance tecnológico ha logrado que muchos dispositivos asequibles y de bajo costo cuenten con herramientas que permitan capturar datos relacionados con la cinemática vehicular. Uno de ellos es el *Smartphone* este cuenta con una IMU interna, con la cual es posible obtener datos de sensores como el acelerómetro, giroscopio y magnetómetro; además de un módulo GPS para obtener datos de velocidad, latitud y longitud [24]. Otros tipos de dispositivos aprovechan la conexión en el puerto OBD-II ubicado en el vehículo (la

conexión puede lograrse en autos con protocolos no privativos) para obtener información relevante del vehículo como: velocidad, posición del pedal acelerador, etc. [26].

Los datos obtenidos en estos sensores contienen ruido causado por la condición de la carretera, motor, fricción del vehículo y otros factores externos como pueden ser el ruido gaussiano blanco [27]. Por lo tanto, es importante usar técnicas de filtrado que mitiguen este problema. En la literatura estudios viales como [26] y [27] hacen uso del filtro Kalman para eliminar el ruido y la interferencia de los datos.

Los estudios de ND usan la cinemática del vehículo para identificar diferentes eventos de conducción. Por ejemplo, la velocidad angular en "Z" puede detectar maniobras de conducción lateral (girar, cambiar de carril) y el acelerómetro en "Y" puede detectar maniobras longitudinales (frenar, acelerar) [27]. Identificar y usar estos eventos genera conocimiento en estudios de análisis y seguridad vial, gracias a este conocimiento es posible generar modelos predictivos de comportamientos de conducción o eventos de riesgo como los *near-crashes* [12].

2.4. *Machine Learning* en estudios de ND

El *Machine Learning* es considerado como una de las ramas más importantes de la Inteligencia Artificial (IA) [28], [29]; sin embargo, no solo reúne conceptos relacionados con la IA, es un campo de investigación que además involucra conceptos de estadística e informática [30]. Su definición puede variar un poco de una referencia a otra, por ejemplo, en [28] es resaltada la definición de ML dada por Tom Mitchell, quien afirma que *"… el ML es el estudio de los algoritmos informáticos que mejoran automáticamente a través de la experiencia"*. Complementando esta idea, es posible afirmar que el ML consiste en codificación de programas que ajustan automáticamente su rendimiento en función de su exposición a la información proporcionada por ciertos datos. En otras palabras, el ML permite extraer conocimiento a partir de los datos [30].

En los últimos años, las aplicaciones de ML han aumentado considerablemente logrando estar presentes en muchas actividades de la vida cotidiana y el éxito de algunas de ellas radica en la automatización de la toma de decisiones [30]. Recomendaciones automáticas sobre qué películas ver, qué comida pedir, qué productos comprar, consumo de cualquier tipo de contenido personalizado, reconocimiento facial, reconocimiento de voz, predicción de la cantidad de productos a elaborar, o la detección de spam, son algunos ejemplos de las aplicaciones del ML [30].

En estudios de seguridad vial el ML es usado como una técnica de análisis de *near-crashes* y choques [11]. Este permite abarcar distintas áreas de interés, como: detección de áreas problemáticas en la vía, detección de AT en tiempo real, prevención de AT y clasificación de la severidad de un AT [9].

2.4.1. Tipos de ML

Fundamentalmente, es posible encontrar tres tipos de ML: el aprendizaje supervisado, el aprendizaje no supervisado y el aprendizaje por refuerzo [29]. Estos son descritos de forma general a continuación:

2.4.1.1. Aprendizaje supervisado

Este tipo de ML abarca los algoritmos que aprenden de un conjunto de datos de entrenamiento con ejemplos etiquetados. Estos algoritmos buscan modelar la relación existente entre ciertas características medidas a partir de datos suministrados por un usuario (*inputs* o *features*) y una etiqueta deseada asociada a los mismos (*output* o *target*). De esta forma los algoritmos pertenecientes a este tipo de aprendizaje generalizan el conjunto de todas las entradas posibles, siendo capaces de etiquetar datos nuevos y desconocidos [29], [31]. Principalmente, dentro del aprendizaje supervisado son encontrados dos tipos de tareas: clasificación y predicción o regresión. Estas dos tareas pueden ser diferenciadas teniendo en cuenta la etiqueta de los datos: en la clasificación las etiquetas de los datos de entrada son categorías o clases discretas, mientras que en la predicción son valores continuos [31].

2.4.1.2. Aprendizaje no supervisado

Este tipo de ML busca modelar las características de un conjunto de datos sin tener como referencia una etiqueta. Al no tener una etiqueta o salida ejemplificada, los algoritmos con este tipo de aprendizaje exploran los datos según criterios estadísticos, geométricos o de similitud, dejando que los datos “hablen por sí mismos” [29], [31]. Para resumir y encontrar características o estructuras claves en los datos, en el aprendizaje no supervisado los modelos emplean técnicas como agrupamiento, reducción de dimensionalidad, detección de *outliers* y detección de novedad [29]. Por ejemplo, en el agrupamiento, distintos grupos son identificados a partir de los datos de entrada, mientras que la reducción de dimensionalidad busca obtener representaciones más breves y concisas de datos, generalmente, cuando tienen un amplio número de características (*features)* [31].

2.4.1.3. Aprendizaje por refuerzo

A este tipo de ML pertenecen aquellos algoritmos que aprenden explorando iterativamente un espacio de soluciones. El aprendizaje se da gracias a recompensas que le proporcionan al algoritmo información sobre la calidad de las soluciones exploradas. Con base en lo anterior son encontradas las mejores soluciones [29].

2.4.2. Algoritmos de ML de clasificación

Como se mencionó en la sección 2.4.1.1, los algoritmos de aprendizaje supervisado aprenden de un conjunto de datos suministrados por los usuarios. Estos datos contienen ejemplares con ciertas características de entrada y sus respectivas salidas deseadas, además representan la forma como el algoritmo recibe “supervisión” [30]. A partir de lo

anterior, los algoritmos son capaces de generalizar dichas muestras y encontrar una manera de producir la salida deseada para nuevos datos de entrada [31]. Si las salidas deseadas corresponden a una lista discreta de clases o categorías predefinidas, se estaría hablando de una tarea de clasificación [30].

En ocasiones, etiquetar los datos de entrenamiento para un algoritmo suele ser un proceso manual y laborioso porque dependiendo de la complejidad del problema pueden ser requeridas enormes cantidades de ejemplos. Una vez los datos a suministrar son etiquetados, cada uno con una respectiva clase, es posible entrenar el algoritmo y este podrá encargarse de asignar una clase a datos nuevos no etiquetados [30].

Regularmente, la clasificación es separada en dos tipos: la primera es la clasificación binaria, en la cual se busca distinguir entre dos clases; la segunda es la clasificación multiclase, donde el número posible de clases es superior a dos [30].

Dentro de la literatura relacionada con análisis de seguridad vial, es posible encontrar que algunos de los algoritmos de clasificación más utilizados son: *Support Vector Machine* (SVM), *Decision Trees* (DT) y *Random Forest* (RF) (las secciones 4.2.3 y 4.3 sustentan esta elección). Estos algoritmos son descritos brevemente a continuación:

2.4.2.1. *Máquina de Vectores de Soporte (Support Vector Machine)*

Las máquinas de vectores de soporte (*Support Vector Machine*, SVM) son una clase de algoritmos de aprendizaje supervisado potentes y flexibles para la clasificación. Las SVM realizan una clasificación discriminativa, en la cual se trata de encontrar una forma de dividir o separar las clases entre sí, a diferencia de la clasificación generativa, donde se busca modelar cada una de las clases o categorías de los datos [31].

Las SVM son un ejemplo común de clasificadores lineales (binarios). Estos clasificadores están orientados a separar dos clases usando un límite de decisión que es una función lineal de la entrada: una línea, un plano o un hiperplano para datos con dimensiones superiores [30]. Para encontrar el modelo óptimo de clasificación, dicha frontera o límite de decisión es ajustado buscando obtener la mayor distancia posible entre él y los puntos más cercanos de ambas clases de datos [29]. Típicamente dichos puntos representan los bordes de cada clase (al ver los datos proyectados en un espacio) y son llamados vectores de soporte, de ahí el nombre de esos algoritmos [30].

Debido a la baja flexibilidad de las líneas e hiperplanos muchos problemas de clasificación no pueden ser solucionados con un límite de decisión lineal [29]. El truco del *kernel* es un método que permite resolver el anterior inconveniente [30], [31]. Existen tres implementaciones de este método en SVM: *kernel* líneal, polinómico, y de función base radial o RBF [29]. Los dos últimos *kernels* extienden las capacidades de las SVM, permitiéndole establecer límites de decisión no lineales adecuados a un conjunto de datos [31].

El SVM puede ajustarse a través de un parámetro de regularización denotado con la letra C. Entre más alto sea el valor de C la regularización es menor, mientras que valores más

bajos de C corresponden a una mayor regularización [30]. Además, cuando el *kernel* empleado es RBF, es necesario ajustar otros parámetros adicionales como el grado de curvatura (*gamma*) [29].

Las SVM son robustas y tienen buen rendimiento en variedad de conjuntos de datos; pero su tiempo de ejecución y uso de memoria son críticos cuando el número de muestras de los datos es muy grande. Además, las SVM pueden ser difíciles de inspeccionar cuando se busca entender el porqué de una clasificación concreta [30].

2.4.2.2. *Árbol de decisión (Decision Tree)*

Los árboles de decisión (*Decision Tree,* DT) son algoritmos de ML sencillos, intuitivos y eficientes al momento de extraer conocimiento de los datos [28]. Los DT pueden ser generados por diferentes métodos, entre los que están: ID3 (*Iterative Dichotomiser*), C4.5, C5.0 y *Classification And Regression Trees* (CART) [28]; no obstante, independientemente del método, todos tendrán una estructura similar.

La estructura de un DT está conformada por nodos internos, cada uno de los cuales representa una prueba de valor de una determinada *feature* del "*data set"* [32]. De cada uno de estos nodos se desprenden dos ramificaciones o reglas de decisión correspondientes a un valor o rango de valores de la característica en cuestión. Los nodos finales o nodos hoja, están asociados a un valor de la etiqueta (*target*), es decir a una clase. El punto inicial de un DT es el nodo raíz. Este corresponde a la *feature* que mayor cantidad de información aporta [28].

Durante la creación de un DT, el proceso de identificar la característica que transmite más información sobre las etiquetas de los datos es realizado de forma recursiva por medio de la medida de la entropía (falta de regularidad en los datos) [28]. Así, la cantidad de información aportada por un atributo es calculada mediante la diferencia entre la entropía del "*data set*" antes de considerar dicho atributo y la entropía después de considerarlo [32].

Para aplicar la entropía en características cuyos valores son continuos, estos son convertidos a atributos booleanos mediante el uso de umbrales, de esta forma, un valor continuo de la característica en cuestión toma un valor booleano dependiendo de si dicho valor está por encima o por debajo del umbral seleccionado [32].

La poda o *pruning* es una técnica que permite reducir el peligro de sobreajuste. Esta consiste en reemplazar uno o más subárboles con hojas, donde cada hoja tendrá la clase más común entre los ejemplos de entrenamiento que alcancen el subárbol eliminado (usando el clasificador original) [32].

Generalmente, se prefieren los árboles de decisión pequeños que los árboles grandes, lo anterior debido a que tienen una mayor interpretabilidad, es decir, son más fáciles de analizar, explicar e incluso corregir. Además, ayudan a eliminar información irrelevante y redundante. Por otra parte, los árboles de decisión grandes tienden a sobre ajustarse [32].

2.4.2.3. *Bosque aleatorio (Random Forest)*

Los bosques aleatorios (*Random Forest*, RF) son un método de ensamble. Estos son métodos que combinan múltiples algoritmos de ML para crear modelos más robustos empleando una técnica de agregación [29]. Uno de los inconvenientes de los árboles de decisión es que tienden a sobre ajustarse a los datos de entrenamiento. Los métodos de ensamble, como los RF, tienen propiedades que ayudan a combatir el sobre ajuste [30].

Como su nombre lo indica, un RF es una colección de árboles de decisión, donde cada uno realiza la respectiva tarea (clasificación o predicción) de manera aceptable y debe ser diferente de los demás, incluyendo sus errores cometidos [30]. Asegurar una correlación lo más baja posible es necesario para que la combinación funcione correctamente [29]. Para lograr esto, los RF inyectan cierta aleatoriedad al proceso de creación de los árboles de decisión que lo conforman [30].

Los RF logran esta aleatorización de dos formas. La primera es seleccionando diferentes muestras de datos para ser utilizadas en la construcción de cada árbol de decisión. Este proceso es llamado muestreo *Bootstrap* y consiste en seleccionar repetitiva y aleatoriamente una muestra del conjunto de datos de entrenamiento hasta construir un nuevo conjunto de datos de igual tamaño que el original [30].

La segunda forma consiste en seleccionar aleatoriamente un subconjunto de *features* del "*data set"*, a partir del cual son elegidas aquellas características que representen la mejor prueba para cada nodo interno de un árbol de decisión [30].

Para realizar una clasificación, los RF utilizan la estrategia de "voto suave" (*soft-voting*). Con esta estrategia los árboles de decisión brindan una probabilidad para cada posible etiqueta de salida. Posteriormente estas etiquetas se promedian para seleccionar la clase con mayor probabilidad [30].

Los RF son uno de los métodos de ML más usados debido a su potencia y baja necesidad de ajuste de parámetros. Sin embargo, su interpretabilidad disminuye debido al gran número de árboles de decisión que podría tener y a la profundidad de estos. Además, los RF requieren más memoria y procesamiento para su entrenamiento, principalmente cuando el conjunto de datos es muy grande y el número de árboles de decisión que lo conforman es muy alto. Cabe resaltar que este último también es un parámetro ajustable de los RF [30].

2.4.3. Métricas de evaluación

En el ciclo de desarrollo de modelos de ML es importante tener en cuenta el apartado de evaluación, con el propósito de identificar si un modelo ha logrado entrenar los datos de forma correcta, lo anterior se conoce comúnmente como rendimiento del modelo. Definir el rendimiento de un modelo puede resultar engañoso, por lo que es importante tener claro cuál es el objetivo para evaluar los aspectos de interés en la aplicación [32].

Existen cuatro medidas fundamentales para definir el rendimiento de un modelo, las cuales pueden ser definidas de la siguiente manera tomando como ejemplo el caso de un modelo que busque clasificar eventos de *near-crash* y eventos de conducción normal:

- Verdadero positivo (Vp), cuando la etiqueta es positiva y el clasificador predice correctamente. Es decir, cuando hay un evento de *near-crash y* el clasificador es capaz de detectar dicho evento.

- Verdadero negativo (Vn), cuando la etiqueta es negativa y el clasificador predice correctamente. Es decir, no hay un evento de *near-crash* y el clasificador tampoco detecta el evento etiquetándolo correctamente como conducción normal.

- Falso positivo (Fp), cuando la etiqueta es negativa y el clasificador la predice incorrectamente. Es decir, cuando en un evento de conducción normal, donde no hay un *near-crash*, el clasificador lo etiqueta incorrectamente como si hubiese sido este último.

- Falso negativo (Fn), cuando la etiqueta es positiva y el clasificador predice incorrectamente. Es decir, cuando hay un evento de *near-crash* pero el clasificador no es capaz de detectarlo, etiquetándolo incorrectamente como un evento de conducción normal.

Haciendo uso de estas medidas de rendimiento es posible definir distintas métricas de evaluación, como la precisión (o Acc, por la abreviación de su equivalente en inglés: *accuracy*), este es el cálculo de las clasificaciones correctas hechas por un modelo [32]. El mejor valor que puede tener Acc es 1 en un rango de 0 a 1. Y su ecuación es la siguiente:

$$Acc = \ (Vp + Vn\ )/(Fp + Fn + Vp + Vn) \qquad (1)$$

Esta medida es adecuada para clasificar un *"data set"* con clases balanceadas (lo cual significa que el número de clases positivas es aproximadamente igual al número de clases negativas), pero para *"data sets"* con clases no balanceadas es necesario tomar otro tipo de medidas, ya que el rendimiento del Acc puede aparentar ser perfecto y aun así errar al tomar nuevos datos [32]. Como ejemplo, es posible tomar una muestra con 100 datos, donde 99 datos son clasificados como verdaderos negativos y 1 de ellos es clasificado como un falso negativo; al aplicar la **ecuación (1)** se obtiene un resultado igual a 0.99. Este resultado sugiere un clasificador perfecto, pero en realidad este clasificador solo está retornando clases negativas, por lo tanto, el rendimiento no es extraordinario. En estos casos es necesario usar métricas de evaluación que logren solventar el problema de clases no balanceadas [32].

Existen métricas que centran su criterio en evaluar con mayor prioridad la clase con menos muestras, algunas de estas son: "*precision"* y "*recall".*

- ***Precision****,* es la probabilidad de que el clasificador este en lo correcto cuando una clase positiva es predicha [32]. Su ecuación es la siguiente:

$$precision = \ Vp/(Vp + Fp) \quad (2)$$

- ***Recall**,* es la probabilidad de que el clasificador logre reconocer todos los eventos positivos [32]. Su ecuación es la siguiente:

$$recall = \ Vp/(Vp + Fn) \quad (3)$$

Cada una de estas métricas tiene un uso específico para el tipo de aplicación a la que esté dirigida, como ejemplo: "*precision"* es mayormente usada en sistemas de recomendaciones, donde es importante entregarle al cliente el producto correcto; y "*recall"* en casos de diagnósticos médicos, donde es importante evitar diagnósticos erróneos [32]. En muchos casos usar las dos métricas conlleva grandes ventajas, de esta manera surge la métrica de evaluación "***F1-score***", combinando las dos anteriores en una sola medida cuantificativa. Su ecuación es la siguiente:

$$F1 = 2\,.\,(precision\,.\ recall)/(precision + recall) \quad (4)$$

El área bajo la curva ROC (AUC, del inglés *Area Under Curve ROC*) es otra métrica usada para definir el rendimiento de un clasificador, esta toma en cuenta métricas como el "*recall"* y la tasa de falsos positivos ("*specificity",* probabilidad de que una clase negativa sea predicha como positiva) [33]. En general AUC busca que un clasificador distinga correctamente todas las clases positivas y negativas, entre mejor sea esta distinción más se aproxima al valor máximo (1.0). En estudios de ND el AUC permite comparar con mayor objetividad el rendimiento de varios clasificadores de *near-crashes* [34].

2.5. Pre-procesamiento de los datos

Independientemente del algoritmo de ML utilizado, es importante usar técnicas de pre-procesamiento en los datos recolectados en vehículos, con el fin de preparar los modelos predictivos. De esta manera es posible reducir costos computacionales o el sobre ajuste de los modelos. Dos aproximaciones usadas para lograr lo anterior son: la selección de características, donde las variables con más importancia para el modelo son seleccionadas; y la extracción de características, donde las variables son transformadas a un nuevo conjunto de datos que no pierde la información relevante [10].

Generalmente, las pruebas de ND permiten obtener grandes cantidades de datos, pero los eventos críticos dentro de estos datos son escasos, aun si los eventos de estudio (choques) fueran substituidos por los *near-crashes*, por esta razón los *"data sets"* de ND son considerados como no balanceados. En este tipo de "*data sets*", las técnicas de pre-procesamiento como la selección y extracción de características ayudan a crear modelos con buen rendimiento [26].

2.5.1. Selección de características

La selección de características consiste en identificar y descartar las variables que no tengan gran importancia en la construcción del modelo. Usar esta técnica no garantiza una

óptima selección de variables; aun así, el uso de ella ha demostrado tener buenos resultados, como evitar el sobre ajuste y lograr mejoras en el rendimiento del modelo [10].

2.5.2. Extracción de características

La extracción de características consiste en generar una nueva dimensionalidad de los datos por medio de la combinación matemática de sus variables. Las ventajas de este método incluyen: lograr una abstracción de los datos, generar un contexto en datos temporales y reducir la complejidad del problema [10].

2.5.3. Ventana de tiempo deslizante

La ventana de tiempo deslizante (*sliding window* o *time window*) surge como una técnica usada para la extracción de características o patrones en los datos. Esta se define como la división de datos secuenciales en conjuntos caracterizados por una medida significativa [34].

La ventana de tiempo deslizante se compone de dos características: 1) Tamaño de la ventana, que es el número de muestras que contiene la ventana. 2) Intervalo deslizante, que es el número que define el punto inicial en el que comenzara la próxima ventana [26]. En la **Figura 2** podemos apreciar un ejemplo de una ventana deslizante, con un tamaño igual a 4 e intervalo igual a 1. La “transformación” hace referencia a la función de transformación aplicada en las muestras contenidas por una ventana, normalmente es un cálculo aritmético o estadístico (Ej. El promedio de todas las muestras dentro de la ventana).

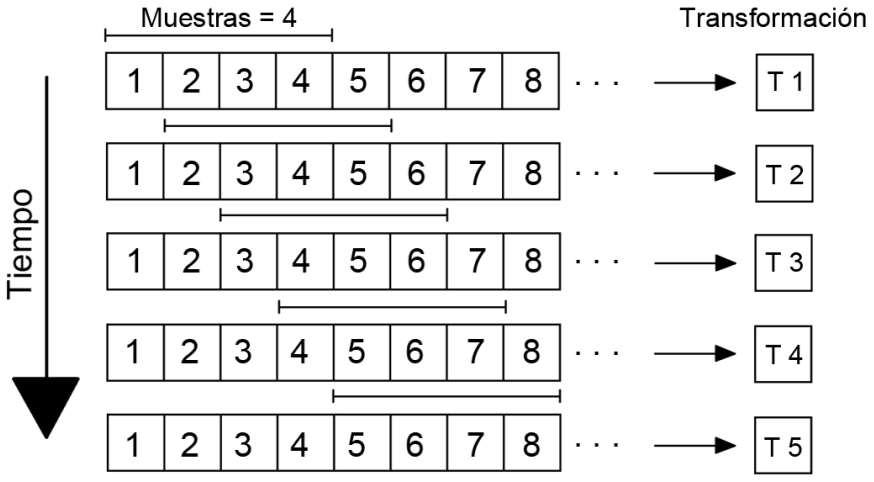

Figura 2. Ventana de tiempo deslizante

Capítulo 3

3. Estado del arte

3.1. Revisión de la literatura

En este trabajo fue realizada una revisión sistemática de la literatura. En dicha revisión fue usada la metodología *PRISMA* [35], la cual propone cuatro fases: identificación, detección, elección e inclusión. El resumen del proceso desarrollado en cada una de las fases es presentado a continuación. Los detalles de la revisión realizada se describen en el **Anexo B. Revisión sistemática de la literatura mediante la metodología PRISMA.**

3.1.1. Fase de identificación

En la fase de identificación, fueron realizadas búsquedas en la literatura con respecto a los siguientes temas: ciencias de la computación o datos, accidentalidad vehicular y medidas de prevención. El proceso de búsqueda en la literatura fue realizado por medio de una cadena de consulta conformada por un conjunto de palabras asociadas a los temas propuestos anteriormente, quedando de la siguiente manera: ***(“artificial Intelligence" OR "machine learning") AND (reduction OR prevention OR prediction) AND ("driving safety" OR "driving risk" OR "near crashes" OR "vehicle crashes" OR "vehicle accidents")***. A partir de ella se procedió a buscar en tres bases de datos: *Science Direct*, *EBSCO*, *IEEE*.

Al final de este proceso y haciendo uso de algunos filtros de búsqueda fue obtenido un total de 687 documentos. Además, fueron añadidos 17 documentos de otras fuentes, obteniendo un total de 704 documentos que pasaron a ser evaluados en la siguiente fase.

3.1.2. Fase de detección

En la fase de detección fueron filtrados los documentos encontrados en la fase anterior. De esta forma fueron incluidos los artículos que tuvieran una relación con el objetivo de nuestro trabajo, esta relación fue comprobada revisando el título y resumen (*abstract*) de los documentos. Al finalizar el filtrado, los artículos duplicados también fueron eliminados. De esta forma fueron obtenidos, al final de toda la fase, 141 documentos que pasaron a ser evaluados en la fase de elección.

3.1.3. Fase de elección

En la fase de elección fueron revisados cada uno de los 141 documentos obtenidos en la fase de detección. Los criterios de elegibilidad manejados en esta etapa (de tal forma que los documentos cumplieran al menos uno de ellos) fueron:

- Documentos relacionados con ND o *near-crashes*.
- Documentos que tengan relación o hagan uso de ML o presenten documentación estadística para la predicción, identificación de accidentes viales o *near-crashes*.
- Documentos que presenten pruebas de campo, diseños de prototipos o detalles acerca de recolección de datos de accidentes, choques o *near-crashes*.
- Documentos que manipulen "*data sets"* de ND.

El total de documentos obtenidos al finalizar esta fase fueron 27, los cuales pasaron a evaluarse en la siguiente fase.

3.1.4. Fase de inclusión

En la última fase, los 27 documentos (obtenidos en la anterior fase) fueron clasificados en diferentes grupos basados en sus similitudes, determinando los siguientes cuatro grupos: 1) identificación y estudio de características que influyen en los choques haciendo uso de "*data sets"* de conducción naturalista; 2) diseño e implementación de sistemas o técnicas de recolección de datos de conducción normal o naturalista; 3) marcos de trabajo, metodologías, técnicas, modelos, análisis y estudios relacionados con la detección de accidentalidad; 4) tecnologías de análisis de datos o algoritmos de ML para realizar predicciones o detección de accidentes.

Para los cuatro grupos identificados fue realizada una síntesis cualitativa y para los grupos 2 y 4 fue realizada una síntesis cuantitativa, a partir de ciertos parámetros de comparación. Estas síntesis pueden ser consultadas en detalle en el **Anexo B. Revisión sistemática de la literatura mediante la metodología PRISMA**. Al finalizar la revisión de literatura fue construido el estado del arte dividiendo el estado actual del problema en los grupos mencionados anteriormente.

3.2. Trabajos relacionados

La anterior revisión sistemática permitió conocer el estado actual del conocimiento, identificando diferentes puntos de vista, aproximaciones, sistemas, experimentos, métodos y tecnologías que fueron de ayuda y complemento para el desarrollo del presente trabajo. A continuación, los trabajos más relevantes son presentados para cada uno de los grupos identificados en la fase de inclusión de la revisión sistemática.

3.2.1. Identificación y estudio de características que influyen en los choques haciendo uso de "*data set*" de ND

- El artículo ***Performance of basic kinematic thresholds in the identification of crash and near-crash events within naturalistic driving data*** [23] identifica por medio del "*data set*" estadounidense SHRP2 y el canadiense CNDS situaciones

donde se presenta un choque o un *near-crash*. Los autores definieron las variables y valores claves que permiten la detección de AT y usan la curva ROC para ajustar los umbrales de sensibilidad de dichas variables. Obtuvieron como resultado una tabla con las variables más significativas, sus respectivos umbrales y la tasa de falsos positivos.

Este artículo brindó un buen punto de partida considerando el análisis exhaustivo de los parámetros que permiten la identificación del choque y casi-choque. El uso de la base de datos SHRP2, una de las fuentes más importantes para este tipo de estudios, resalta la importancia del trabajo. La diferencia de este trabajo con la investigación realizada fueron que: no realizan el proceso de recolección de datos, ni detectaron las zonas con mayor probabilidad de AT, tampoco hicieron uso de técnicas de ML y el contexto de recolección de los datos fue diferente.

3.2.2. Diseño e implementación de sistemas o técnicas de recolección de datos de conducción normal o naturalista

- El artículo ***Driving analytics using Smartphones: Algorithms, comparisons and challenges*** [24], centra su investigación en el uso de los *Smartphones* como una alternativa para reunir datos y analizar el comportamiento de conducción aprovechando la información del giroscopio, el acelerómetro y el GPS. El enfoque de los autores incluyo un algoritmo de orientación del dispositivo, para corregir los datos básicos del acelerómetro. El articulo menciona la aplicación de un modelo de ML basado en la teoría del conjunto aproximado (*Rough Set*) para identificar las reglas, encontrar valores de umbral óptimos y detectar patrones críticos basándose en los datos corregidos del acelerómetro.

  Este estudio permitió identificar los *Smartphones* como una opción factible para recopilar los datos de conducción necesarios para el proyecto de investigación. A partir de los resultados de este trabajo, los sensores de estos *Smartphones* fueron calificados como fiables y precisos para detectar eventos de aceleración, frenado y giros bruscos. El sensado de los datos por parte de los *Smartphones* (para identificar su fiabilidad) fue comparado con el sensado por un sistema OBD-II el cual proporciona información similar y es comúnmente utilizado en el diagnóstico de la operación de un vehículo. Las brechas principales de este trabajo, respecto al trabajo de investigación fueron: la falta de detección de las zonas de alta probabilidad de AT y el contexto con el que los datos son medidos (el cual es diferente).

- El objetivo principal del artículo ***Driver behavior profiling: an investigation with different Smartphone sensors and Machine Learning*** [34] fue evaluar el rendimiento de múltiples combinaciones de algoritmos de ML y sensores de un *Smartphone* con sistema operativo Android para detectar eventos de conducción agresiva. Los eventos detectados fueron: la aceleración, frenado, giros y cambio de carril (tanto agresivos como no agresivos). Para esto, en la recolección de datos emplearon sensores (acelerómetros, magnetómetros y giroscopios) y para la

detección de cada evento aplicaron algoritmos de ML (*Artificial Neural Network*, *Bayesian Network,* SVM y RF).

Los resultados y conclusiones de este trabajo representaron un aporte considerable en la detección de *near-crashes*, ya que los autores identificaron algunos eventos de conducción relevantes para dicha detección. Adicionalmente, identificaron algunos algoritmos útiles para la detección de eventos relacionados con la conducción. La diferencia principal del artículo con el trabajo de investigación estuvo relacionada con la falta de detección de zonas con alta probabilidad de AT, ya que solo fueron detectados eventos de conducción para establecer ciertos comportamientos del conductor.

3.2.3. Marcos de trabajo, metodologías, técnicas, modelos, análisis y estudios relacionados con la detección de accidentalidad

- El artículo ***Evaluating the Driving Risk of Near-Crash Events Using a Mixed-Ordered Logit Model*** [11] presenta una metodología para la captura, predicción y clasificación de *near-crashes*. La captura de los datos, correspondiente a la parte experimental del trabajo, fue realizada con ND usando sensores a bordo del automóvil. Al finalizar la experimentación, los datos fueron almacenados en una base de datos para posteriormente descubrir y clasificar (por medio del método *K-means*) los eventos de *near-crashes* que tenían mayor probabilidad de riesgo.

  Una diferencia observada en este artículo (respecto a la propuesta de este trabajo) fue el uso de algunos de los sensores empleados para la recolección de datos como cámara y LIDAR. Ya que el presente trabajo no contempla el uso de este tipo de sensores debido a su alto costo. El contexto donde se llevó a cabo este estudio (Wuhan, China) difiere de la ciudad propuesta en nuestro trabajo (Popayán). Y, por último, no identificaron las zonas con mayor probabilidad de AT dentro de la ciudad.

- El artículo ***Near Crashes as Crash Surrogate for Naturalistic Driving Studies*** [12], presenta el beneficio de usar *near-crashes* en conjunto con los choques para aumentar la posibilidad de detectar con mayor frecuencia los AT. El artículo comienza haciendo una revisión de los métodos que solo hacen uso de choques para la detección de AT, dejando claro que es necesaria una mejor metodología para evaluar los riesgos de AT. De esta forma los autores determinan que los *near-crashes* en conjunto con los choques obtienen mejores resultados y menores gastos logísticos al momento de predecir un choque.

  Este estudio refuerza la idea de elegir *near-crashes* debido a la dificultad logística que representa capturar datos de choques reales dada su naturaleza esporádica. En el estudio del artículo no se realizó el proceso de recolección de datos con ND, ni se identificaron las zonas con mayor probabilidad de AT.

- El artículo ***A critical overview of driver recording tools*** [20] compara los estudios relacionados con métodos para la detección de accidentes. Los métodos analizados

en este artículo van desde métodos tradicionales de encuestas (informes policiales, cuestionarios, etc.) hasta el uso de sistemas o dispositivos en recorridos con ND (sistemas OBD-II o dispositivos como *Smartphone*). Al finalizar los autores entregaron su visión de las ventajas y desventajas que tiene cada método.

La aproximación realizada en este artículo entregó posibilidades sobre métodos viables para llevar a cabo el estudio del presente trabajo. El uso de *Smartphone*, sistemas como el OBD-II o sensores electrónicos que midieran la inercia en 9 ejes (conocidos como IMUs) fueron algunas de las opciones presentadas. Este artículo presenta una gran diferencia respecto al trabajo de investigación por ser un artículo de revisión. Debido a esto, además de que no identificaron zonas de alta probabilidad de AT, tampoco realizaron un proceso de recolección de datos con ND.

- El artículo ***A Novel Model-Based Driving Behavior Recognition System Using Motion Sensors*** [26] propone un sistema de reconocimiento del comportamiento de conducción basándose en la clasificación de eventos agresivos y no agresivos. Este sistema ejecutó un proceso de investigación compuesto por seis segmentos, dando mayor importancia al análisis teórico. De esta forma, los autores evitaron que su sistema propuesto fuera similar a una caja negra (como ocurrió en otros estudios). Los datos fueron evaluados mediante varios clasificadores (SVM*, Bayesian Network*, DT*,* etc.), siendo SVM el mejor clasificador, con una precisión de 93.25%.

  El sistema propuesto por este trabajo fue factible y muy completo en comparación con otros trabajos. El artículo realiza aportes importantes con relación a algunos objetivos del presente trabajo de investigación, el contexto donde se realizó la recolección de datos (rutas de una ciudad intermedia), la construcción de un "*data set"* limpio, un análisis profundo del mismo y una preparación cuidadosa de los modelos de clasificación de ML. La brecha principal de este trabajo (con respecto al trabajo de investigación) se da respecto a los eventos en los que se enfoca, dado que en este se detectan eventos diferentes a los *near-crashes*.

3.2.4. Tecnologías de análisis de datos o algoritmos de ML para realizar predicciones o detección de accidentes

- El artículo ***Prediction of Near-Crashes from Observed Vehicle Kinematics using Machine Learning*** [36] propone como hipótesis que las pequeñas turbulencias captadas por los datos cinemáticos de un vehículo pueden ser usadas para predecir *near-crashes*. Los datos usados provenían de la base de datos SHRP2 y fueron analizados por distintos modelos de ML (usando algoritmos como SVM, DT*, Gaussian Naive Bayes* y *Adaptive Boost*). Al final, considerando las medidas de precisión y *F1-score* los autotrs determinaron que el modelo *Adaptive Boost* se comporta de la manera más precisa (100%) evitando así los falsos positivos y validando la hipótesis inicial.

Este artículo presenta un modelo de ML para identificar *near-crashes*. Las brechas encontradas con respecto a la investigación realizada fueron: la falta de un proceso de recolección de datos, el contexto de procedencia de los datos usados (no es una ciudad intermedia de país en desarrollo) y la falta de identificación de las zonas con alta probabilidad de AT.

- El artículo ***Driving risk assessment using near-crash database through data mining of tree-based model*** [37] considera un experimento con ND en el cual construyeron una base de datos conformada únicamente por *near-crashes*. Los autores utilizaron el método *K-means* para clasificar los *near-crashes* en 3 niveles de riesgo de acuerdo con las características de frenado. Finalmente, usando CART exploraron la relación entre el riesgo de conducción, las características del conductor-vehículo y los entornos de la carretera para identificar los factores más influyentes en los niveles de riesgo.

  El sistema de registro de conducción de este estudio obtuvo ciertos datos de variables usadas para la detección de *near-crashes*. Estas variables fueron consideradas para el desarrollo de la investigación. A diferencia de este artículo, donde se valida el nivel de riesgo en un *near-crash* a través de video, el presente trabajo pretende realizar la validación únicamente de la ocurrencia de *near-crash* y no su nivel de riesgo. Además, la validación propuesta tomaba en cuenta la fuente de datos obtenida mediante un dispositivo ubicado en el vehículo que utilizara sensores de bajo presupuesto.

- El artículo ***A spatiotemporal deep learning approach for citywide short-term crash risk prediction with multi-source data*** [38] investiga cómo una aproximación de *Deep Learning* puede predecir los riesgos de accidente en una ciudad, haciendo uso de diferentes bases de datos de múltiples fuentes. Los tipos de datos recolectados fueron: datos de choque, datos de viajes de taxi, atributos de red en la carretera, características de uso de la tierra y población. Los datos recolectados provinieron de diversas fuentes como: agencias de transporte, población (por medio de censo), centros climáticos, entre otros. Como resultado principal, un mapa de calor muestra las zonas de mayor riesgo de accidentalidad.

  La generación del mapa de calor que identifica las zonas de mayor riesgo es un aspecto importante de este artículo y se consideró dentro del desarrollo de la investigación. Este artículo presenta algunas diferencias respecto al proyecto de investigación; una de ellas radica en la fuente de datos, este estudio partió de bases de datos ya construidas por diferentes agencias. Además, en este artículo no fueron considerados los eventos de *near-crashes*, debido a la gran cantidad y calidad de información encontrada en las múltiples bases de datos.

- El artículo ***Brake Maneuver Prediction – An Inference Leveraging RNN Focus on Sensor Confidence*** [39] propone una red neuronal recurrente (RNN) con el propósito de aumentar la seguridad a través de la predicción de la acción del freno. Los autores basaron su estudio en la ND, capturando los datos del freno por medio

del sistema OBD-II y finalizaron conformando un "*data set*" que fue analizado por la técnica de RNN. Al finalizar, concluyeron que al usar la técnica de RNN, el área bajo la curva ROC fue incrementada en un 14% comparado con algoritmos convencionales.

El artículo solo considera incrementar la precisión del factor de acción del freno, a diferencia del trabajo de investigación desarrollado, en donde son considerados más factores para determinar los AT. Además, el enfoque principal de este artículo es la disminución de los falsos positivos, a diferencia del trabajo a desarrollar, en el cual la identificación de zonas con mayor probabilidad de AT es el principal objetivo.

3.3. Resumen de evaluación de trabajos relacionados

A continuación, la **Tabla 1** presenta un resumen de los criterios considerados en las propuestas de trabajos relacionados. El presente trabajo consideró todos los criterios identificados, mientras que los trabajos relacionados no tuvieron en cuenta algunos o varios de dichos criterios.

Tabla 1. Resumen de evaluación de los trabajos relacionados

Criterio/Propuestas	[11]	[12]	[20]	[23]	[24]	[26]	[34]	[36]	[37]	[38]	[39]
Identificación de zonas con mayor probabilidad de AT.										✓	
Identificación de AT.		✓	✓	✓						✓	✓
Identificación de *near-crashes.*	✓	✓	✓	✓				✓	✓		
Ciudad intermedia como contexto.						✓	✓				
Recolección de datos con ND.	✓				✓	✓			✓		✓
Uso de modelos de ML para analizar la información recolectada.	✓				✓	✓	✓	✓	✓	✓	✓
Uso de sensores asequibles		✓	✓	✓	✓	✓	✓			✓	✓

Capítulo 4

4. Parámetros y arquitectura del sistema

Este capítulo describe el proceso mediante el cual fueron seleccionados los parámetros de un Sistema de Detección Inteligente de Riesgos de Colisión (SDIRC), conformado por un módulo de recolección de datos (*Hardware o* HW*)* y un módulo de análisis inteligente de la información (*Software* o SW) (**Figura 3**); el primero encargado de la recolección y almacenamiento de ciertas variables relacionadas con *near-crashes,* y el segundo encargado del análisis de los datos para identificar *near-crashes*. Además, este capítulo presenta la arquitectura propuesta para el desarrollo del prototipo del SDIRC.

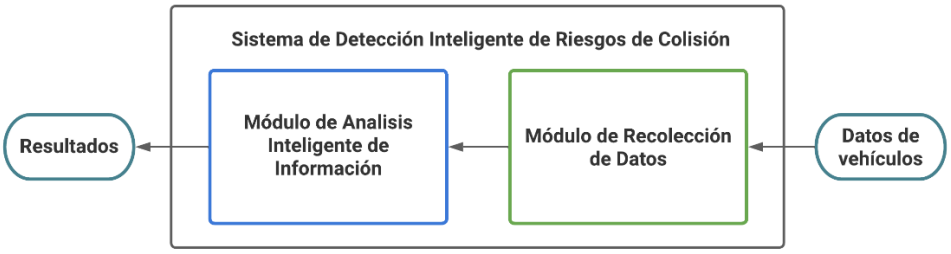

Figura 3. Esquema general de la arquitectura para el SDIRC

4.1. Identificación de los parámetros relevantes del sistema

Con el fin de determinar los factores necesarios para los módulos de recolección de datos y análisis inteligente de la información según las necesidades y contexto del presente trabajo, fue realizada una revisión detallada del estado del arte. La revisión de cada artículo fue enfocada en resolver las siguientes preguntas: ¿Qué variables o datos consideró y para qué los utilizó?, ¿Qué *hardware* utilizó para la medición o recolección de los datos?, ¿Cómo fue el experimento o prueba de campo empleado en la recolección? y ¿Qué algoritmos o *software* implementó para el análisis o procesamiento de esos datos?

Las respuestas a las anteriores preguntas permitieron obtener una visión general de diferentes variables, dispositivos, modelos y/o técnicas de procesamiento de datos empleados en soluciones a problemas de seguridad vial (similares a la hipótesis de este trabajo), y como estos están correlacionados.

Los resultados de las caracterizaciones fueron sintetizados en la **Tabla 2** y **Tabla 3**, las cuales presentan la información clave extraída de los artículos mencionados.

La **Tabla 2** muestra los dispositivos o sistemas identificados en la literatura, que son usados en la recolección de los datos vehiculares o de comportamiento de conducción. Cabe resaltar que los valores listados en las columnas sensores y variables son una combinación de los presentes en los artículos relacionados.

Tabla 2. Hardware utilizado en la recolección de datos del vehículo

Dispositivo	Sensores	Variables	Artículo(s)
Smartphone	• Acelerómetro. • Giroscopio. • GPS. • Magnetómetro.	• Aceleración lineal. • Tasa de rotación. • Latitud, longitud, elevación. • Tiempo. • Campo Magnético.	[20], [24], [34]
Vehículo Instrumentado	• Cámaras. • Acelerómetro de 3 ejes. • Radar delantero. • GPS. • Sensor de iluminación. • Giroscopio. • Seguidor ocular. • Sensor de alcohol pasivo. • Sensor infrarrojo. • Botón de incidente y micrófono. • Sensor de giro.	• Múltiples videos y audio. • Desaceleración. • Aceleración longitudinal. • Aceleración lateral. • Latitud, longitud, elevación. • Tiempo. • Velocidad. • Velocidad angular. • Presión del pedal de freno. • Posición de cambio. • Activación de *airbag* • Angulo del volante, etc. • Tiempo de colisión trasero y delantero (TTC: *Time to Colission*).	[11], [12], [20], [23], [36], [37]
OBD-II	• Sensores incorporados en el vehículo.	• Posición del pedal de freno. • Posición del pedal acelerador. • Sensor de combustible. • Presión del combustible. • Velocidad. • Revoluciones del motor. • Aceleración. • Tiempo de activación • Control de estabilidad electrónico, ESC (*Electronic Stability Control*) • Activación de *airbag*.	[11], [20], [23], [36] [39]
IMU *(Inertial Measurement Unit)*	• Acelerómetro de 3 ejes. • Giroscopio de 3 ejes. • Magnetómetro de 3 ejes.	• Aceleración lateral y longitudinal. • Tasa de rotación. • Velocidad.	[26]

Por otro lado, la **Tabla 3** detalla los algoritmos utilizados en los modelos de procesamiento y análisis de los datos recolectados.

Tabla 3. Algoritmos de ML empleados en estudios de seguridad vial

Artícu-los	Algoritmos	Métricas de evaluación	Características (*Features)*	Etiqueta (*Target*)	Técnicas adicionales / Complementos

[23]	No especificado.	*Specificity*, *sensibility*, ROC *curve*.	Aceleración longitudinal, Aceleración en autopista, monitor de activación del *Anti-Lock Braking System* (ABS), aceleración lateral, tiempo de activación ESC, control de tracción, sacudida lateral, tasa de rotación, maniobras de viraje, activación de airbag.	Rendimiento en la detección de choques y *near-crashes.*	Tablas de proporción de detecciones válidas para cada característica y combinaciones de estas.
[11], [12]	Uso de análisis estadístico.	*P-value, Z-statistics*.	Aceleración lateral, aceleración longitudinal, botón de incidente crítico, TTC delantero, TTC trasero, velocidad de guiñada, desaceleración, tiempo de avance, presión del freno, desaceleración media, energía cinemática, congestión, razón de *near-crash*, hora, día, tipo de Carretera, experiencia de conducción (años), edad, fin de semana (si o no), millas de conducción.	*Near-crash*	*Stata* (Software para análisis estadístico y *Data Science*) y reloj de sincronización.
[36]	*K-Nearest Neighbors* (KNN), RF, DT, *Gaussian Naive Bayes* (NB), AdaBoost, SVM.	*F1 Score*, *Recall, precision*.	Velocidad, aceleración longitudinal, aceleración lateral, velocidad angular, posición del pedal (%).	Positiva: *Near-crash*. Negativa: Conducción normal.	*Python SciKit learn (*)*, ventana de tiempo deslizante (10 *folds*), técnica de *cross validation* con K=5*.
[37]	*K-means*, CART.	Acc.	*Clustering*: Desaceleración máxima, desaceleración promedio, porcentaje de reducción de la energía cinética en los vehículos. Clasificación: Velocidad al momento de frenar, factores desencadenantes,	*Clustering* y clasificación en: Bajo riesgo, riesgo moderado, alto riesgo.	-

			objeto potencial de choque, tipo potencial de choque, edad del conductor, maniobra realizada por el vehículo, posible segunda tarea del conductor, entre otras.		
[24]	MODLEM (**), Zero R, C4.5 *Trees*, *Multilayer Perceptron* (MLP).	Tasa de Vp, Tasa de Fp, Acc, *Precision*, *F1-Score*.	Frenada brusca, aceleración brusca, giro derecho brusco, giro izquierdo brusco.	Positiva: Detección correcta de un evento brusco. Negativa: Otros casos.	*Cross Validation* con K=10, *MatLab*.
[34]	*Artificial Neural Network* (ANN), SVM, RF, *Bayesian network* (BN).	ROC *curve*, AUC.	Aceleración con gravedad incluida de 3 ejes, aceleración sin gravedad excluida de 3 ejes, campo magnético de 3 ejes, tasa de rotación de 3 ejes, marca de tiempo.	Frenada Agresiva, aceleración Agresivo, giro a la derecha agresivo, giro a la izquierda agresivo, cambios de línea agresivo a la izquierda, cambio de línea agresivo a la derecha.	Uso del software WEKA (***), *Cross Validation* (****) con K=10, LIBSVM(*****)
[39]	RNN.	Tasa de Vp, Tasa de Fp, AUC, Acc.	Velocidad de las 4 ruedas, posición del pedal de aceleración, aceleración lateral y longitudinal, ángulo del volante.	Porcentaje presionado del pedal del freno en 0,1 segundos.	Pre-procesamiento de las señales de los sensores para obtener muestras a 10 Hz.
[26]	SVM, *Radial basis function network* (RBF-N), *Logistic Regression* (LR), BN, C4.5 *Trees*, *Gaussian* NB, KNN.	Acc, *Confusion Matrix*.	Aceleración 3 ejes, velocidad angular 3 ejes, intensidad de inducción magnética 3 ejes.	Aceleración, frenado, giro izquierdo, giro derecho, giro en U, cambio de carril izquierdo, cambio de carril derecho.	Software iNEMO (******), Filtro reductor de ruido: *Kalman Filter*.
[38]	*Spatiotemporal convolutional Long Short-Term Memory Network* (STCL-Net). Compuesta de tres redes neuronales:	Error cuadrático medio, Error absoluto medio, Error porcentual	Define 3 tipos de variables según sus datos espaciales y temporales. **Tipo I:** Kilómetros viajados diariamente por cada vehículo, área comercial, área residencial, población	Mapa de predicción de riesgo de choque.	Realizan una distribución espacial de áreas en Manhattan dividiéndola en celdas de diferentes tamaños.

	Convolutional neural networks (CNN), *Long Short-Term Memory* (LSTM), *Convolutional Long Shor-Term Memory*.	absoluto medio.	x10^4, densidad del carril (long. de carretera / área celda), porcentaje de autopistas, porcentaje de vías arteriales, porcentaje de vías locales, intersecciones. **Tipo II:** Temperatura, precipitación, nevada, presión, velocidad del viento. **Tipo III:** Riesgo de choque, viajes de taxis.		Las variables de los diferentes "*data set"* son clasificadas en 3 tipos. Para cada tipo es asignado una de las 3 redes que componen el STCL-Net.

Notas: *SciKit Learn* (*): Librería de ML de código abierto que soporta algoritmos de aprendizaje supervisado y no supervisado, varias herramientas para ajustes de modelo, pre-procesamiento de datos, selección y evaluación de modelos, y muchas otras utilidades [40].
MODLEM (**): Algoritmo de aprendizaje automático que induce un conjunto mínimo de reglas, estas reglas se pueden adoptar como clasificador, se inventó para hacer frente a datos numéricos sin discretización [24].
WEKA (***): Colección de algoritmos de ML para tareas de minería de datos [41].
Cross validation (****): procedimiento de remuestreo utilizado para la evaluación de modelos de ML, el proceso tiene un parámetro K que indica el número de divisiones de la muestra de datos [42].
LIBSVM(*****): librería con soporte para múltiples lenguajes de programación que facilita el uso de SVM [43].
iNEMO (*Inertial modules (******)*): son IMUs que integran tipos de sensores complementarios [26].

4.2. Análisis de alternativas

El análisis de alternativas para el desarrollo del prototipo de un SDIRC fue realizado a través de la caracterización propuesta y los parámetros relevantes encontrados (sección 4.1). El análisis fue realizado para cada uno de los siguientes componentes: dispositivos, variables a medir, y algoritmos o modelos de ML. Dicho análisis estuvo enfocado en identificar y señalar los beneficios y obstáculos involucrados por el uso de cada alternativa. La **Figura 4** resume el análisis realizado.

Debido a que existe una fuerte dependencia entre las variables a medir, los dispositivos de recolección de datos y los algoritmos de ML, elegir cualquiera de los componentes (por ejemplo: variables) condiciona en la elección de los otros componentes (por ejemplo: dispositivos de recolección y algoritmos de ML). La información clave encontrada en el análisis, junto con la correlación existente entre los tres componentes, permitieron realizar la elección de la alternativa que fue considerada la más conveniente.

4.2.1. Dispositivo de recolección de datos

De acuerdo con lo mencionado por los autores en [12], parte del potencial de los datos naturalistas figura en los datos cinemáticos precisos del vehículo y en los datos recolectados por instrumentación ubicada dentro de los mismos; sin embargo, la recolección de datos de conducción puede lograrse mediante el uso de diversos tipos de

sensores y/o dispositivos, desde *Smartphones* hasta equipos dedicados como cámaras de monitoreo, cajas telemáticas, sistemas OBD-II, etc. [24].

Debido a las múltiples opciones que podrían tomarse para el desarrollo del módulo de recolección de datos (HW*)*, la **Tabla 2** fue determinada como punto de partida para analizar la información. Esta tabla muestra la relación entre: los sensores, el dispositivo que los incorpora y las variables medidas. El análisis comparativo de las alternativas identificadas busca seleccionar el dispositivo más adecuado para el contexto del presente trabajo, partiendo de las 4 alternativas presentadas en la **Tabla 2** (vehículo instrumentado, smartphones, IMU y OBD-II). Las funciones principales de este dispositivo son: medición de datos o variables y el almacenamiento de la información vehicular.

El primero de ellos fue denominado “Vehículo instrumentado”. Esta opción permite la recolección precisa de datos de conducción basados en la metodología de ND. Lo anterior es posible gracias al gran equipamiento de sensores e instrumentos que deben integrarse en el vehículo para capturar la elevada cantidad de variables requeridas [20].

Las ventajas principales del “vehículo instrumentado”, con respecto a las otras opciones, son: permitir el análisis y validación robusta de los eventos de *near-crash*, la creación de modelos de ML con buen rendimiento y evitar modelos sub-ajustados [20]. El sub-ajuste hace referencia a la incapacidad del modelo de ML para modelar un *“data set”* de entrenamiento, obteniendo un rendimiento pobre [42]. En el trabajo presentado en [23], los sensores son combinados en grupos para lograr un mejor rendimiento en la detección de *near-crashes* o choques. Esta técnica de agrupación (utilizada en los “vehículos instrumentados”) les concede una ventaja con respecto a las demás opciones de dispositivos, las cuales no la utilizan.

Para el SDIRC y el contexto en particular, un grupo grande de sensores a bordo de un vehículo haría que el dispositivo exceda el presupuesto disponible, debido a la inversión necesaria para su adquisición, al tipo de pruebas a realizar y a la cantidad de pruebas requeridas. Otro aspecto para considerar es que algunos de estos sensores son usados para captar variables que no son de utilidad en el contexto establecido en el proyecto, relacionado con *near-crashes* (por ejemplo, el seguidor ocular).

El dispositivo “*Smartphone*” es el más utilizado en estudios recientes de ND [20], debido a las ventajas que genera su uso, estas son: recolectar grandes cantidades de datos en corto tiempo, el costo del desarrollo de una solución con estos dispositivos es relativamente bajo, facilita la recolección de datos de varios vehículos en simultaneo (mediante la distribución de la misma aplicación móvil en varios dispositivos), y por último, su uso es sustentable y escalable [24].

No obstante, el *“Smartphone”* posee ciertas desventajas que pueden afectar al desarrollo del SDIRC. La principal es la cantidad de ruido captada por los sensores [20]. Algunos trabajos ([24] y [34]) hacen uso de filtros o métodos de limpieza para solucionar el problema del ruido, lo anterior implica una mayor demanda de tiempo en el proceso de desarrollo del módulo de análisis inteligente de la información. Otra desventaja es la posible ocurrencia

de interrupciones en la captura de datos, causada por el uso del dispositivo por parte del usuario para cualquier otra tarea durante la conducción [26].

Los sistemas OBD-II son otra opción identificada. Este dispositivo fue desarrollado con el fin de proveer a los vehículos capacidades de autodiagnóstico y generación de reportes [20]. A partir de su segunda generación (OBD-II), fueron estandarizados elementos como: la interfaz para el diagnóstico, los protocolos de señalización eléctrica y el formato de los mensajes, facilitando la interacción con algunos sistemas [20]. Además, la interfaz de diagnóstico de estos sistemas permite la lectura de datos de un vehículo, directamente desde sus sensores incorporados, sin necesidad de procesos de calibración o de técnicas de pre-procesamiento de datos. Gracias a esto, los sistemas OBD-II han sido señalados como la solución más precisa en la recolección de datos vehiculares [24].

En algunos sistemas de recolección de datos que utilizan el sistema OBD-II de los vehículos como fuente de datos, el ancho de banda del protocolo de comunicaciones puede limitar el número de datos o variables que se desea monitorear simultáneamente, como es mencionado en [20]. Lo anterior podría representar una desventaja para trabajos que demanden la lectura de un gran número de variables del vehículo, ya que el ancho de banda permitido por el protocolo de comunicación podría ser excedido. Un ejemplo de lo anterior es el caso citado por [20], en el cual fue empleado el protocolo *Controller Area Network* (CAN). Este es un protocolo de comunicaciones en serie que soporta multiplexación y control distribuido en tiempo real, y es usado en diferentes aplicaciones de control, entre ellas las de vehículos de carretera [44]. CAN no es el único protocolo de comunicación, los sistemas OBD-II fueron diseñados para ser compatibles con diferentes protocolos en las capas física y de enlace de datos del modelo *Open Systems Interconnection* (OSI). Esto condiciona la ocurrencia o no de la limitación mencionada, ya que el ancho de banda puede aumentar o disminuir dependiendo del protocolo de comunicación implementado en sistema OBD-II del vehículo [45].

A pesar de la diversidad de variables que es posible recolectar empleando el sistema OBD-II, surgen varias desventajas adicionales al usar este tipo de dispositivos en el presente trabajo. Primero, muchos de estos dispositivos no son universales o algunos vehículos no tienen liberados sus protocolos, por lo tanto, capturar estas variables depende del tipo de vehículo a usar [26]. Segundo, la imposibilidad de medir datos relacionados con el giro del vehículo, su velocidad angular o tasa de rotación, variables ampliamente usadas en estudios relacionados con los *near-crashes* [20].

El otro tipo de dispositivos son los denominados IMU. Estos dispositivos agrupan sensores inerciales que permiten la captura de datos cinemáticos en un vehículo [26], utilizados recientemente, debido principalmente a su bajo costo de implementación. Otro factor importante que ha incrementado su uso es la forma no intrusiva de capturar y almacenar los datos, ya que pueden situarse a bordo de un vehículo sin causar incomodidad o consumo energético extra. A diferencia del *Smartphone*, los IMU no experimentan interrupciones en la captura y pueden alcanzar una mayor precisión en sus medidas [26].

Una de las desventajas de usar IMU es la necesidad de un filtro posterior a la captura de los datos para eliminar el ruido generado [26]. Además, los dispositivos IMU suelen ser componentes de un sistema más complejo, ya que únicamente suministran los datos de los sensores. Esto hace necesario un proceso de integración con otros componentes que almacenen y procesen los datos medidos.

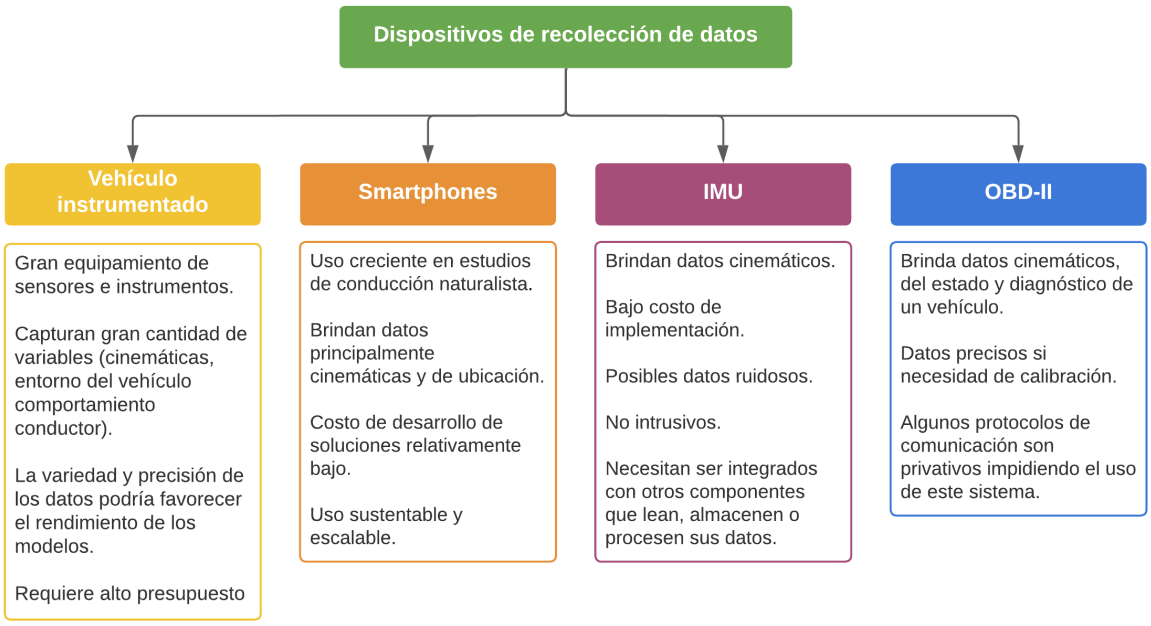

Figura 4. Resumen de los dispositivos de recolección de datos

4.2.2. Variables

En la literatura examinada fueron encontradas 21 variables de interés (estas pueden observarse en la **Tabla 2** y la **Figura 5**), siendo las más comunes: aceleración/desaceleración; velocidad; aceleración longitudinal y lateral; latitud y longitud; tiempo; posición del pedal de freno y acelerador; tasa de rotación y airbag. Para la selección de las variables utilizadas se priorizaron aquellas que estuvieran estrechamente ligadas a los eventos de *near-crash* y choques. Es decir, las variables más relevantes para el desarrollo del SDIRC fueron las que caracterizaron de mejor manera la ocurrencia de estos eventos de interés. Por lo anterior, fue necesario considerar la definición de *near-crash* realizada en la sección 2.2, como también las variables utilizadas en los trabajos más relacionados con *near-crashes,* incluidos en el estado del arte ([11], [12], [23], [36]).

En [11] un *near-crash* es catalogado a través de tres indicadores: *time headway* (intervalo de tiempo entre dos vehículos sucesivos), la desaceleración longitudinal y la presión del freno. Además, presenta los 10 factores que más influyen en el nivel de riesgo de choque, estos son: el promedio de desaceleración, la energía cinética del vehículo, la causa que provocó el *near-crash* (evidenciada en video), la congestión vehicular, la hora del día, el tipo de carretera, las millas conducidas, la edad del conductor, experiencia y tipo de día.

En [23] son presentados los umbrales de variables que obtienen un mejor rendimiento al detectar choques o *near-crashes.* Estas variables son: la desaceleración, la sacudida lateral (*lateral-jerk*) y la activación del sistema antibloqueo de ruedas (*Anti-Look Braking* o ABS).

Los autores en [12] y [36], utilizaron variables vehiculares como: aceleración, velocidad, frenado, tasa de rotación y dirección; variables de entorno como: condiciones de la superficie y clima; y variables de comportamiento del conductor como: distracción y obstrucción visual.

La revisión de los trabajos mencionados evidencia que en la ocurrencia de un *near-crash* predominan dos eventos de conducción: la desaceleración (o frenado) y el cambio de dirección o giro evasivo. Por esta razón, para la detección de un *near-crash*, a partir de variables cinemáticas y considerando los anteriores eventos, es importante usar sensores como el acelerómetro y el giroscopio, puesto que estos sensores son capaces de medir variables como aceleración y tasa de rotación, respectivamente.

En otros artículos ([20], [24], [26], [34]) además del uso de los dos anteriores sensores, también es utilizado un magnetómetro. Dicho sensor está enfocado en la medida de la intensidad del campo geomagnético de la tierra. El magnetómetro en conjunto con el giroscopio obtiene una mayor precisión al momento de detectar cambios de orientación o dirección del vehículo (giro brusco o evasivo), respecto a un punto de referencia (norte magnético terrestre).

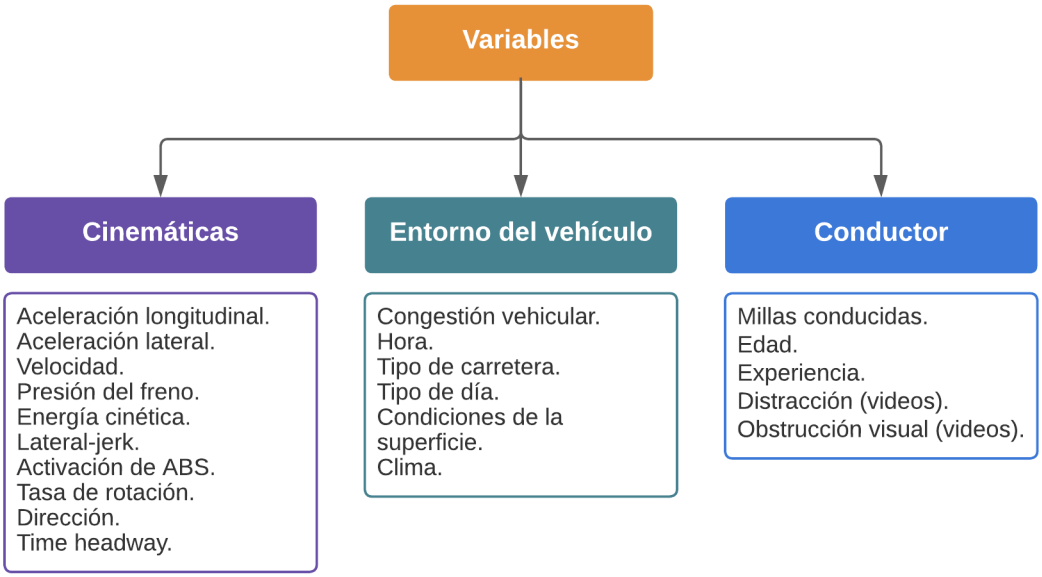

Figura 5. Resumen de las variables para la detección de *near-crashes*

4.2.3. Algoritmos y modelos de ML

En los documentos de la sección 3.2 fueron identificados un total de 17 algoritmos, algunos algoritmos pertenecieron a la parte principal del modelo de ML, realizado por los autores, mientras que otros fueron usados como algoritmos comparativos para demostrar la hipótesis del autor. De los 17 algoritmos fue posible identificar que la mayoría pertenecían

a técnicas de aprendizaje supervisado como la clasificación (~70%) y muy pocos de ellos se dedican a usar otros tipos de técnicas (~30%).

Para definir los algoritmos que fueron utilizados en el desarrollo de la propuesta de este documento, son tomados como punto de partida aquellos documentos en los que su modelo clasificara directamente los eventos de choque o *near-crash*. Un ejemplo de esto se encuentra en [36] donde los *near-crash* son clasificados a partir de los datos generados por eventos cinemáticos previos a la ocurrencia de un AT. Los algoritmos clasificadores con mejor rendimiento en este documento son: RF, SVM, *Gaussian* NB, y *AdaBoost*.

Otros documentos ([24], [26], [34], [39]) utilizan modelos de ML para realizar clasificación de eventos de conducción diferentes a los *near-crashes*. Sin embargo, debido a la naturaleza de los eventos clasificados, los modelos de estos estudios pueden ser tomados en cuenta para la detección de *near-crashes* ya que están estrechamente relacionados. Los autores de [24] detectaron eventos de conducción bruscos como: frenado, aceleración y giros bruscos; para esto emplearon diferentes algoritmos de clasificación. Los algoritmos con mejor desempeño fueron: MODLE, MLP y C4.5 *Trees*.

De igual forma, los autores de [34] detectaron eventos como: frenado, aceleración, giros agresivos y adicionalmente el cambio de carril; para lo anterior emplearon algoritmos como ANN, SVM, RF y BN. Además, identificaron la mejor combinación de sensor de movimiento (y sus ejes), algoritmo de aprendizaje (y sus parámetros), y número de muestras en la ventana deslizante para detectar los eventos mencionados.

Otro estudio similar a los anteriores es [26], donde los autores buscaron detectar los mismos eventos que en [34], pero su etiqueta (*target*) no solo incluía eventos agresivos, también incluía los eventos de conducción normales; para lo anterior, emplearon los siguientes algoritmos: SVM, RBF-N, LR, BN, C4.5 *trees*, NB, KNN. La variedad de algoritmos que utilizaron los autores está sustentada en la intención de identificar cual es el mejor clasificador para cada evento.

Los autores de [39] no abarcaron varios eventos o comportamientos de conducción como en los anteriores estudios. Ellos se centraron en la maniobra de frenado y emplearon las RNN para predecir dicha maniobra. Aunque el modelo no fue enfocado a la clasificación, puede ser una base para la detección de *near-crashes*, debido a que el frenado es una de las maniobras más usadas para evitar colisiones.

En [38] el modelo presenta una solución basada en un algoritmo de *Deep Learning* llamado STCL-*Net*, en el que se emplea una gran cantidad de datos e información de múltiples fuentes. Considerando el contexto planteado para el presente trabajo (ciudad intermedia) y el alcance del mismo, la cantidad de datos y variables a analizar es mucho menor al de [38]; por esta razón, es difícil estimar que un algoritmo como STCL-Net pueda ser de utilidad . Por otro lado, el rendimiento de este algoritmo, comparado con otros algoritmos como: CNN, LSTM, ANN, demuestra que una aproximación de Deep Learning puede remplazar a soluciones de ML convencionales, siempre y cuando la solución sea aplicada a grandes cantidades de datos.

Finalmente, la **Tabla 4** describe un análisis cuantitativo del rendimiento de los 17 algoritmos identificados. En esta se indica el valor obtenido por el algoritmo en cada una de sus métricas de evaluación, donde es posible apreciar que estas métricas difieren en cada artículo. La razón de lo anterior radica en el contexto donde fueron usados los algoritmos, el cual dependía del objetivo de desarrollo y las variables de entrada del modelo. Cabe aclarar que los valores indicados en la **Tabla 4** corresponden a los mejores resultados obtenidos por los algoritmos en cada artículo, ya que el rendimiento varió dependiendo de las diferentes técnicas de optimización (como el uso de diferentes hiperparámetros o divisiones del "*data set*" por medio de la validación cruzada).

Tabla 4. Rendimiento de los algoritmos de ML

Algoritmos	**Artículo**	**Rendimiento (%)**							**Comentarios**
		Acc	*Precision*	*Recall*	*F1-Score*	Tasa de Vp	Tasa de Fp	AUC	
AdaBoost	[36]	95	98	100	99	-	-	-	Mejor rendimiento con un horizonte de turbulencia de 2 a 3 segundos.
ANN / MLP	[34]	-	-	-	-	-	-	99.3 a 99.9	Valores obtenidos de 35 mejores resultados.
	[24]	-	93.2	93.2	-	93.2	12.7	-	-
BN	[34]	-	-	-	-	-	-	-	No especificado.
	[26]	91.10	-	-	-	-	-	-	-
C4.5 *Trees*	[24]	-	94.3	94	-	94	5.8	-	-
	[26]	84.75	-	-	-	-	-	-	-
CART	[37]	62	-	-	-	-	-	-	-
STCL – Net (CNN,LSTM)	[38]	99.21	-	-	-	-	0	-	El modelo tuvo mejor rendimiento con datos semanales.
DT	[36]	-	98	97	96	-	-	-	Horizonte de turbulencia usado para: Acc=2 seg. *F1-Score*=6 seg. Recall=7 seg.
GNB	[36]	-	100	93	95	-	-	-	Horizonte de turbulencia usado para: Acc=6 seg. *F1-Score=2* seg. Recall=2 seg.

KNN	[26]	82.95	-	-	-	-	-	-	-
LR	[26]	89.3	-	-	-	-	-	-	-
MODLEM	[24]	-	96.7	96.6	-	96.6	3.1	-	-
NB	[26]	83.20	-	-	-	-	-	-	-
RBF-N	[26]	85.55	-	-	-	-	-	-	-
RF	[36]	-	98	97	97	-	-	-	Horizonte de turbulencia usado para: Acc=4 seg. *F1-Score=4* seg. Recall=2,3,4 seg.
	[34]	-	-	-	-	-	-	98 a 99.6	Valores obtenidos de 35 mejores resultados.
RNN	[39]	75.47	-	-	-	65.83	14.9	-	El enfoque de RNN tiene mejor rendimiento.
SVM.	[36]	-	94	95	94	-	-	-	Horizonte de turbulencia usado para: Acc=6 seg. *F1-Score=2* seg. Recall=2 seg.
	[34]	-	-	-	-	-	-	-	No especificado
	[26]	93.25	-	-	-	-	-	-	-
Zero R.	[24]	-	49.1	70.1	-	70.1	70.1	-	-

4.3. Selección de alternativas

De acuerdo con el análisis realizado en la sección 4.2, fueron determinadas las siguientes variables como las más adecuadas para el SDIRC: aceleración (Acel), velocidad (Vel), velocidad angular (Vang), fuerza del campo magnético (Fcm), marca de tiempo (*timestamp* o Ts), latitud (Lat) y longitud (Lon).

Para la captura de dichas variables fue determinado el uso de los siguientes sensores: acelerómetro, giroscopio, magnetómetro, reloj interno del dispositivo y GPS. De esta manera fue posible medir adecuadamente la cinemática presentada en el vehículo y otros datos necesarios para el prototipo. La relación entre cada variable y los sensores capaces de brindarlas se muestra a través de la **Tabla 5**.

Debido a que los eventos de *near-crash* se manifiestan en intervalos de tiempo cortos, el uso de una medida de tiempo (Ts), con exactitud de milisegundos, resultó ser necesario. En consecuencia también fue conveniente considerar el análisis de datos temporales incluyendo ventanas de tiempo deslizantes o fotogramas con determinado número de registros, como lo propusieron en sus trabajos los autores de [34], [36].

Además, considerando que el enfoque principal del SDIRC desarrollado fue la identificación de zonas con alta probabilidad de AT, es muy importante utilizar las variables de posición: longitud y latitud. Estas variables pueden ser obtenidas a través de un GPS, el cual fue utilizado en la mayoría de los artículos mencionados. Además, el GPS por sí solo, o en conjunto con el acelerómetro, permite medir la velocidad del vehículo. Esta variable contribuye a una eficiente caracterización del *near-crash*, evitando posibles falsos positivos.

Tabla 5. Relación de variables y sensores

Variable	Sensor
Aceleración	Acelerómetro
Velocidad	GPS, OBD-II
Velocidad angular	Giroscopio
Fuerza de campo magnético	Magnetómetro
Marca de tiempo	Reloj del dispositivo
Ubicación (Latitud y longitud)	GPS

En la **Tabla 6,** fueron relacionados los dispositivos de recolección de datos identificados con cada una de las variables seleccionadas anteriormente, indicando si posee el sensor que es capaz de medirla o no. Además, presenta la complejidad y nivel de costo de implementación de cada dispositivo. La presentación de los dispositivos es realizada en orden descendente de acuerdo con su idoneidad para incorporarlo en el sistema, siendo la primera opción el dispositivo considerado más conveniente.

Tabla 6. Clasificación de dispositivos

Dispositivo	Variables							Complejidad	Costo
	Acel	Vel	Vang	Fcm	Ts	Lat	Lon		
Smartphone	✓	✓	✓	✓	✓	✓	✓	Media	Bajo
OBD-II	✓	✓			✓			Baja	Bajo
IMU	✓		✓	✓				Media	Bajo
Vehículo Instrumentado	✓	✓	✓	✓	✓	✓	✓	Alta	Alto

Considerando la información de la **Tabla 6** y el contexto planteado, el *Smartphone* fue seleccionado como la alternativa de mayor viabilidad para el desarrollo del SDIRC. Sin embargo, debido a las desventajas que conlleva el uso del *Smartphone*, fue considerado conveniente el empleo de una fuente de recolección de datos alternativa. Esto con el fin de

comparar los datos recolectados y evidenciar inconvenientes en el proceso de recolección del dispositivo principal (*Smartphone*).

Para seleccionar el dispositivo de recolección de datos alternativo, fue considerada igualmente la **Tabla 6**. Debido al alto costo y la alta complejidad de implementación de un vehículo instrumentado, esta opción de dispositivo como dispositivo alternativo fue descartada. Por otro lado, las opciones restantes, OBD-II e IMU, no permiten la captura de todas las características seleccionadas para este trabajo. Esto llevó a proponer el desarrollo de un dispositivo de recolección híbrido, el cual fue conformado por: una interfaz para la lectura de datos del sistema OBD-II del vehículo, un dispositivo IMU, un módulo GPS y una tarjeta microcontroladora encargada del procesamiento de los datos recolectados.

Por otra parte, para seleccionar el algoritmo de ML que se acoplara de mejor manera a las variables del sistema y tuviera un buen desempeño, fueron establecidas 3 condiciones cuantitativas y una condición cualitativa.

Las condiciones cuantitativas usadas para la selección de los algoritmos fueron: la cantidad de implementaciones, el rendimiento obtenido y el número de datos usados. Los algoritmos que destacaron por la cantidad de implementaciones son SVM y RF, siendo usados en aproximadamente 6 artículos de los presentados en la **Tabla 3**.

Los modelos que destacaron por su rendimiento usaron algoritmos como RF, *AdaBoost*, y DT, obteniendo una media por encima de 0.97. Esto fue evidenciado considerando la **Tabla 4**, donde son presentadas las diferentes métricas de rendimiento, permitiendo identificar los algoritmos con mejor desempeño según cada una de estas métricas. Comparando la "*Precision"* de cada algoritmo, destacaron *AdaBoost*, DT y RF, todos con 0.98; en el caso de "*Recall"*, nuevamente destacaron *AdaBoost* con 1, seguido de DT y RF ambos con 0.97; finalmente, comparando el *F1-score*, *AdaBoost* también logró el mejor rendimiento con 0.99, seguido de RF con 0,97 y DT con 0,96.

Los modelos que destacaron por el número de datos utilizados fueron aquellos que hicieron uso del *"data set"* SHRP2 o pruebas de campo con recorridos considerablemente extensos, trabajos como [34] y [36] tienen esta característica.

La condición cualitativa utilizada para la selección del modelo fue la relación entre las variables de entrada, la etiqueta (*target*) y los objetivos del presente trabajo. Por ejemplo para el trabajo de [36] se tiene el objetivo de detectar *near-crashes* a través de pequeñas turbulencias cinemáticas generadas en el vehículo, por lo tanto los algoritmos usados en este trabajo pueden ser usados para la construcción del modelo de ML del presente trabajo. En este sentido, pudo evidenciarse a través de los artículos obtenidos en la revisión sistemática y la **Tabla 3** que los algoritmos que mejor relación tenían con las variables y objetivos de este trabajo fueron: SVM, RF, DT, *AdaBoost*, y GNB.

Finalmente, la selección del algoritmo de ML no fue única, ya que fueron seleccionados 3 algoritmos para la creación de diferentes modelos. Esto permitió comparar el rendimiento de cada uno de los algoritmos y de esta manera determinar el modelo adecuado para el SDIRC. En este sentido los algoritmos seleccionados para el desarrollo del módulo de

análisis inteligente fueron: SVM, RF, y DT, considerando los resultados de las anteriores condiciones y realizando una ponderación a cada aspecto analizado. Los algoritmos *AdaBoos* y GNB fueron descartados considerando sus principales desventajas, entre ellas están no soportar datos ruidosos y que su aplicabilidad en datos del mundo real es limitada, debido a que normalmente se asume que las *features* del "*data set*" son independientes [40], [46].

4.4. Análisis de arquitecturas

Para proponer una arquitectura del SDIRC, fue necesario llevar a cabo una revisión de las arquitecturas de sistemas similares propuestas en otros trabajos. La revisión consideró tres fuentes: 1) Arquitecturas encontradas en los artículos resultantes al finalizar la revisión sistemática (los cuales se pueden revisar en detalle en el **Anexo B. Revisión sistemática de la literatura mediante la metodología PRISMA**), 2) Arquitecturas de otras fuentes relevantes, 3) Arquitecturas de Sistemas de Transporte Inteligente (ITS) de referencia internacional.

Del primer grupo de artículos considerado como fuente para la revisión de arquitecturas fue posible identificar que muchas de las arquitecturas presentadas en ellos, no tenían relación con el sistema propuesto.

En el segundo grupo, la mayoría de los artículos analizados no tenían relación directa con el objetivo de este trabajo, pero sus arquitecturas tenían características que podían ser usadas en el desarrollo de la arquitectura del sistema propuesto. Muchos de los artículos analizados correspondían a los encontrados en las fases iniciales de la revisión sistemática de literatura, por lo que había una relación con la idea central de esta revisión.

En el último grupo, las siguientes arquitecturas relacionadas con ITS fueron analizadas: *Architecture Reference for Cooperative and Intelligent Transportation*, (ARC-IT) [47] y *European Intelligent Transport Systems Framework Architecture* (FRAME) [48]. Cada arquitectura ITS provee un marco de trabajo y paquetes de servicio que ayudan en la construcción del SDIRC.

El objetivo de esta revisión fue identificar los componentes principales de la arquitectura necesaria para el sistema y las metodologías de diseño adecuadas para dicha arquitectura. La arquitectura finalmente propuesta fue la adaptación de varias arquitecturas existentes, incluyendo los componentes más relevantes, considerando el contexto planteado. A continuación, son presentados los resultados de la revisión de arquitecturas para cada grupo (la definición a detalle de cada arquitectura, así como su revisión son presentado en el **Anexo C. Revisión de arquitecturas**).

4.4.1. Arquitecturas encontradas en los artículos resultantes al finalizar la revisión sistemática

Las arquitecturas de este grupo son en su mayoría arquitecturas dedicadas a definir el proceso para la creación de modelos de ML. Este tipo de arquitectura por lo general define las fases que son empleadas en la creación de modelos de ML y la forma en la cual se logra la predicción de su salida.

No obstante, entre estos artículos fueron identificadas ciertas arquitecturas que proponen un sistema de reconocimiento del comportamiento de conducción. Ellas proporcionaron un concepto básico de lo requerido para el desarrollo de la arquitectura del sistema propuesto en este trabajo. Principalmente en [34] se plantea una división en segmentos: el primero define las funciones que tiene el dispositivo de recolección (*Smartphone*) y el segundo define las funciones relacionadas con el análisis de datos (ejecutadas por un computador).

4.4.2. Arquitecturas de otras fuentes relevantes

Una característica identificada en varios artículos de este grupo fue el planteamiento de un flujo de información directo entre el módulo de recolección de datos (Ej. sensores del vehículo) y el módulo encargado de su análisis y procesamiento. Dicha característica está asociada a ciertos escenarios donde puede ser empleada, Ej. vehículos autónomos o sistemas de asistencia al conductor.

El trabajo realizado en [49] busca construir un sistema de adquisición y análisis de datos vehiculares basado en *cloud* para monitorear en tiempo real el comportamiento del conductor, y presenta una característica similar a la anteriormente mencionada. En este sistema el flujo de información va directamente desde un sistema OBD-II hacía un *Smartphone*; sin embargo, la arquitectura propuesta en este trabajo presenta una variación interesante. El *Smartphone* está encargado de realizar un procesamiento inicial y simple de los datos, adicionalmente se propone un filtro que envía únicamente información relevante hacia otro objeto de análisis y procesamiento más complejo, ubicado en la nube. Este último objeto ejecuta un análisis de datos más robusto, por lo cual puede tardar más tiempo en completarse.

En el enfoque utilizado en [49] para la construcción de su sistema propuesto fue posible identificar dos ventajas. La primera, que el módulo de análisis y procesamiento inicial de los datos (correspondiente a una aplicación móvil de un *Smartphone*) ayuda a que el consumo de ancho de banda se reduzca significativamente, eliminando o descartando datos irrelevantes y enviando al componente *cloud* únicamente datos que vale la pena analizar. La segunda, que el módulo de análisis complejo reduce el consumo energético del módulo inicial y centraliza el acceso a la información más relevante, al encargarse de tareas de procesamiento más pesadas.

A diferencia de lo anterior, cuando el análisis de los datos resulta mucho más simple, el mismo objeto de recolección podría encargarse de esa tarea. Es así como los autores de [50] lo propusieron en su arquitectura, donde el análisis de los datos es simplificado con

pequeñas validaciones que pueden realizarse a medida que estos están siendo recolectados. Además, la arquitectura de este artículo presenta un objeto de almacenamiento temporal para los datos medidos. Esta última característica resulta útil en sistemas donde el procesamiento o análisis de los datos es realizado esporádicamente y no necesariamente en el mismo vehículo; es decir, primero se realiza la recolección y almacenamiento (temporal) de los datos para que después de un cierto periodo de tiempo, sean analizados por el mismo dispositivo o extraídos y analizados por otro modulo del sistema.

La arquitectura propuesta en [51] presenta dos segmentos, uno *offline* y otro *online* (**Figura 6**). El segmento *offline* muestra una característica que guarda cierta similitud a la mencionada en [34]. Además, dicho segmento describe una base de datos encargada del almacenamiento durante un periodo de tiempo muy extenso. Esta base de datos es conectada a dos objetos relacionados con el análisis de datos: uno de extracción de características y otro de entrenamiento. Por otra parte, el segmento *online* se orienta a la aplicación del modelo clasificador. Aquí, el flujo de información parte de un módulo de recolección de datos y pasa por módulos de pre-procesamiento, antes de enviar los datos al modelo clasificador. La estructuración de los módulos de una arquitectura en segmentos *online* y *offline* puede presentar ciertas ventajas o desventajas dependiendo del sistema a implementar. Para el presente trabajo usar un módulo *online* podría aumentar los costos de desarrollo del prototipo, también implicaría posibles pérdidas de información durante el envío de información a un componente de análisis de datos alojado en la nube, debido a la infraestructura de red encontrada en muchas de las ciudades intermedias de países en desarrollo. Por otro lado, considerar la división de procesos en dos módulos puede permitir la reducción de costos computacionales en los dispositivos de recolección, manteniendo una asignación de responsabilidades acorde a la capacidad de los módulos.

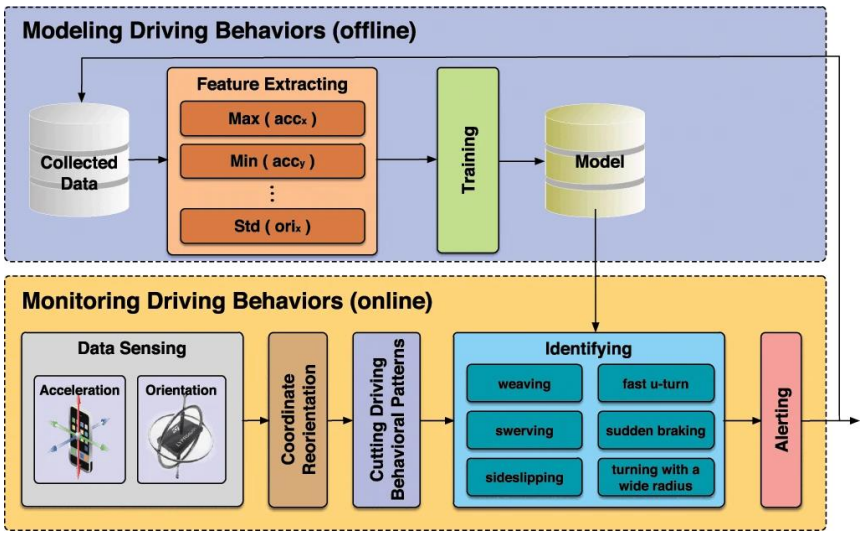

Figura 6. Arquitectura para el sistema de detección de comportamientos de conducción inadecuados, propuesta en uno de los trabajos revisados. Fuente: [51]

4.4.3. Arquitecturas de Sistemas de Transporte Inteligente

En esta sección son presentadas las dos arquitecturas de referencia internacional más utilizadas a nivel mundial para los ITS, estas son: ARC-IT y FRAME.

4.4.3.1. ARC-IT

El uso de esta arquitectura permite tener una base para la planificación, definición e integración del sistema propuesto [47]. ARC-IT se divide en diferentes vistas, empresarial, física y de comunicación. Sin embargo, para el modelado de la arquitectura del sistema propuesto fue requerido el uso de la vista física. Esta vista contiene clases, objetos físicos, objetos funcionales y flujos de información; todos estos facilitan la descripción de la arquitectura propuesta.

En ARC-IT también es posible encontrar arquitecturas establecidas para un servicio en específico, llamadas paquetes de servicios. Por lo tanto, los siguientes servicios fueron tomados como referencia para el desarrollo de la arquitectura del sistema propuesto: DM02 Supervisor de rendimiento, VS03 conocimiento situacional y VS05 Advertencia de velocidad de curva, los cuales pertenecen a las áreas de servicios manejo de datos (*data management*) y seguridad vehicular (*vehicle safety*).

La **Figura 7** presenta el paquete de servicio para supervisar el rendimiento de los vehículos conectados en la red ITS, además de sus datos históricos. La información se obtiene a través de sensores, detectores, fuentes de datos operativos, entre otras fuentes.

ARC-IT proporciona una arquitectura sólida y con una buena abstracción de sus objetos, la cual facilita la creación de sistemas como el propuesto en el presente trabajo.

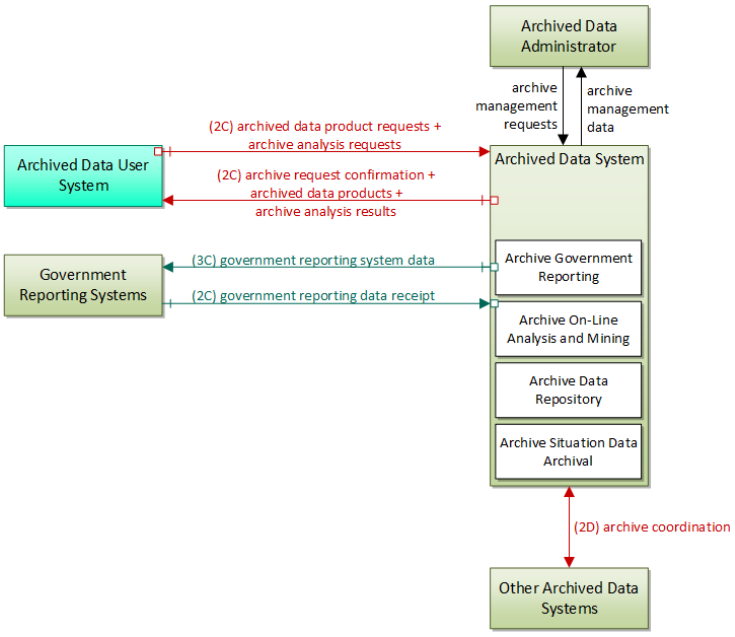

Figura 7. DM02 Monitoreo de rendimiento (*performance monitoring*) [47]

4.4.3.2. FRAME

La arquitectura FRAME al igual que ARC-IT divide su estructura en vistas, siendo la vista funcional la necesaria para implementar servicios ITS [48]. La vista funcional se divide en 8 áreas funcionales, de las cuales dos están relacionadas con el propósito del sistema propuesto, estas son: el área para proporcionar instalaciones de seguridad y emergencia; y el área para soporte de los servicios del vehículo anfitrión.

En estas dos áreas fueron encontradas algunas arquitecturas que fueron tomadas como referencia para el desarrollo del SDIRC, estas son: detección y análisis del vehículo; datos de vehículo; manejo de reportes para un vehículo robado; y detectar fraude o infracción.

FRAME es un buen complemento para ARC-IT, principalmente en el uso de flujos y objetos que permiten tomar como base comunicaciones que ARC-IT no contempla. Algunos flujos y objetos de FRAME fueron usados para lograr el desarrollo de la arquitectura del sistema propuesto en el presente trabajo.

4.5. Arquitectura propuesta

Mediante la revisión de otras arquitecturas de sistemas similares o relacionados; y la revisión de literatura realizada, fue posible proponer la arquitectura del SDIRC (la definición en detalle de cada componente de la arquitectura es presentada en el **Anexo C. Revisión de arquitecturas**). Como primer punto, fue decidido aplicar el enfoque de arquitectura dividida en segmentos. Lo anterior considerando que fueron identificados dos módulos principales para el sistema: uno de recolección de datos (HW) y otro de análisis inteligente de la información (SW), cada uno de los cuales formaría un segmento o subsección de la arquitectura.

Inicialmente, fue analizada la comunicación entre los dos segmentos mencionados, teniendo en cuenta que no era requerido que dicha comunicación operase en línea, ya que, realizar un intercambio de información en línea necesitaría de una comunicación estable y de baja latencia. Este tipo de comunicación en el contexto de interés podía ser difícil de lograr, debido a que la operación de redes móviles no tiene la cobertura y calidad necesarias, y no existen otro tipo de redes de comunicaciones que pudieran servir para este objetivo (como por ejemplo una red LoRa [52]). Adicionalmente, en caso de utilizar las redes móviles existentes, el costo de uso también era un factor para considerar.

La detección de *near-crashes* requiere de datos medidos enviados con una alta frecuencia, la pérdida de datos debido a una inadecuada comunicación implicaría un mal desempeño del modelo y por ende resultados pobres. Teniendo en cuenta esto, el uso de un módulo de almacenamiento temporal fue considerado adecuado, generando ciertas ventajas como la disminución de costos del sistema y reducción en la complejidad de desarrollo. Lo anterior implicó que lo más adecuado fuera que el módulo encargado del análisis de los datos se implementara como un segmento offline. De esta forma fue necesario que los datos fuesen extraídos por un agente externo después de un periodo de tiempo de recolección, para posteriormente ingresarlos al módulo de análisis.

ARC-IT fue seleccionada como arquitectura modelo, para ajustar y proponer una opción para el SDIRC, teniendo en cuenta su organización, claridad, estandarización y permanente actualización. Los segmentos o módulos, objetos físicos, objetos funcionales y flujos de información usados en la arquitectura propuesta fueron determinados, considerando lo propuesto por ARC-IT. Algunos de ellos fueron tomados directamente de los paquetes de servicios establecidos en ARC-IT. Otros, por el contrario, fueron definidos o ajustados según las características del contexto o las funcionalidades necesarias del sistema.

4.5.1. Segmentos o módulos

Los segmentos son descritos como las subsecciones, módulos o grupos de una arquitectura, en el diseño de la arquitectura propuesta los segmentos denotan los dos módulos que fueron definidos en los objetivos del sistema a desarrollar. Estos son: módulo de recolección de datos y módulo de análisis inteligente de la información.

- **Módulo de recolección de datos (*Data collection module*):** En este módulo son agrupados aquellos objetos físicos que tengan funcionalidades relacionadas con el sensado, almacenamiento y recolección de datos del vehículo.
- **Módulo de análisis inteligente de la información (*Intelligent information analysis module*):** En este módulo son agrupados aquellos objetos físicos que tengan funcionalidades relacionadas con el análisis o manejo de los datos del vehículo, estos datos deben ser previamente obtenidos por el módulo de recolección de datos.

4.5.2. Clases

- **Vehículo (*Vehicle*, color: azul):** Los objetos físicos encontrados en esta clase incluyen información u objetos funcionales que permitan la administración o manejo del sistema vehicular.
- **Soporte (*Support*, color: amarillo-verde):** Los objetos físicos encontrados en esta clase tratan funciones habilitadoras, facilitadoras o de gestión de comunicaciones.
- **Centro (*Center*, color: verde-cian):** Los objetos físicos en esta clase tienen funciones de aplicación, gestión, administración y manejo. Peculiarmente esta clase no es involucrada directamente con el sistema o no es anclada en la red del sistema.

4.5.3. Actores

- **Administrador de Datos Archivados (*Archived Data Administrator*)**: Dicho usuario realiza las tareas de extracción de los datos del módulo de recolección de datos y los ingresa al módulo de Sistema de Datos Archivados (*Archived Data System*). Además, es el encargado de inicializar las tareas de procesamiento requeridas y monitorear los resultados obtenidos.

4.5.4. Objetos físicos

- **Vehículo básico (*Basic vehicle*):** Representa al vehículo operando, incluye sus interfaces, plataformas y datos electrónicos a bordo del vehículo.
- **Vehículo OBE (*Vehicle OBE – Vehicle On Board Equipment*):** Provee al vehículo con funciones sensoriales, de procesamiento, de almacenamiento y de comunicaciones basadas en los viajes o trayectos.
- **Administrador de datos archivados (*Archived data administrator*):** Representa las operaciones realizadas por un operador humano, en la arquitectura propuesta este objeto es el encargado de manejar los datos entre los dos módulos o segmentos.
- **Sistema de datos archivados (*Archived data system*):** Recolecta, archiva, maneja y distribuye los datos entregados por medio del administrador de datos archivados.
- **Sistema de datos de usuario archivados (*Archive data user system*):** Es el sistema que utiliza el usuario para acceder, manipular, analizar y procesar los datos archivados. La interfaz que le deja a los usuarios visualizar los resultados
- **Sistema de presentación de resultados (*Results Presentation System*):** Es el sistema que utiliza el usuario para visualizar la información de los trayectos o viajes vehiculares.

4.5.5. Objetos funcionales

- **Almacenamiento de los datos recolectados (*Collected data storage*):** Permite almacenar los datos vehiculares que han sido recolectados en sus viajes o trayectos.
- **Registro cinemático del vehículo (*Vehicle kinematics recording*):** Permite la recolección de datos cinemáticos a través de sensores a bordo del vehículo.
- **Determinación de la ubicación del vehículo (*Vehicle location determination*)** Permite la recepción de la actual ubicación del vehículo.
- **Filtro y preprocesamiento de los datos (*Data filtering and preprocessing*):** Permite el filtrado de los datos y su preprocesamiento, esto con el fin de lograr tener datos confiables para su posterior análisis.
- **Análisis y minería de archivos (*Archived analysis and mining*):** Permite ejecutar funciones avanzadas de análisis, resumen y minería de grandes conjuntos de datos; con el fin de lograr la predicción de comportamientos o identificar patrones específicos de un problema. Además, permite el almacenamiento y gestión del "*data set"*.

- **Central *data set:*** Corresponde al conjunto de los datos recopilados de todos los diferentes vehículos que hagan parte del proceso de recolección de datos.

La arquitectura propuesta, sus módulos, componentes y flujos son presentados en la **Figura 8**. Los textos están en inglés debido a que esta arquitectura fue presentada en el artículo de investigación de una revista internacional.

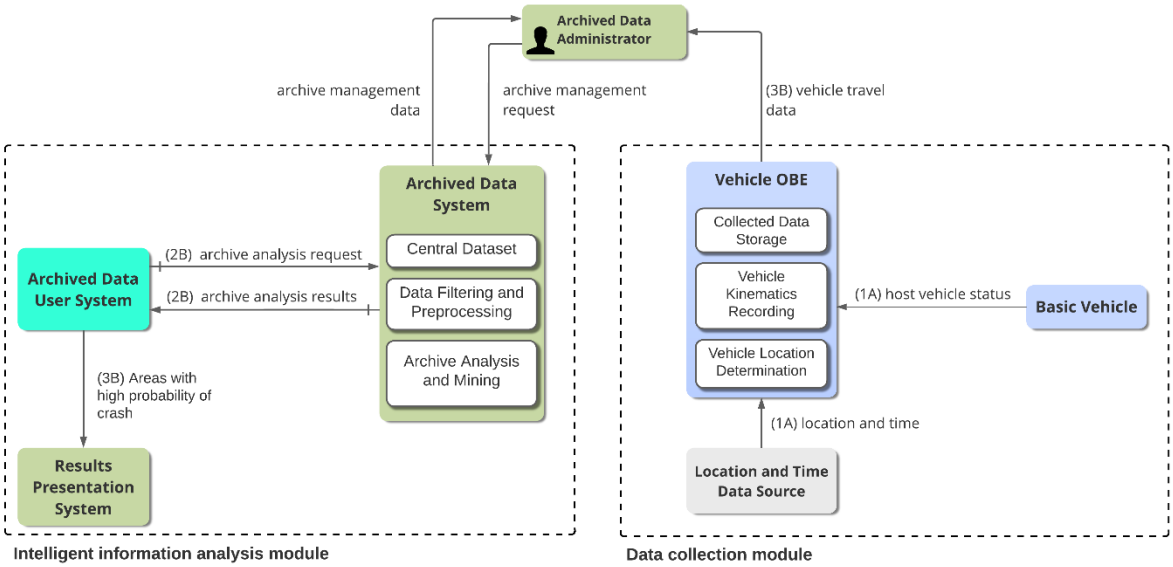

Figura 8. Arquitectura del SDIRC

Capítulo 5

5. Diseño y desarrollo del prototipo

A partir de la arquitectura previamente propuesta, fue posible diseñar un prototipo para el SDIRC, el cual es presentado en la **Figura 9**. Cada uno de los módulos, objetos físicos, objetos funcionales y flujos de información son descritos en detalle en este capítulo, incluyendo las tecnologías usadas y su proceso de implementación y desarrollo.

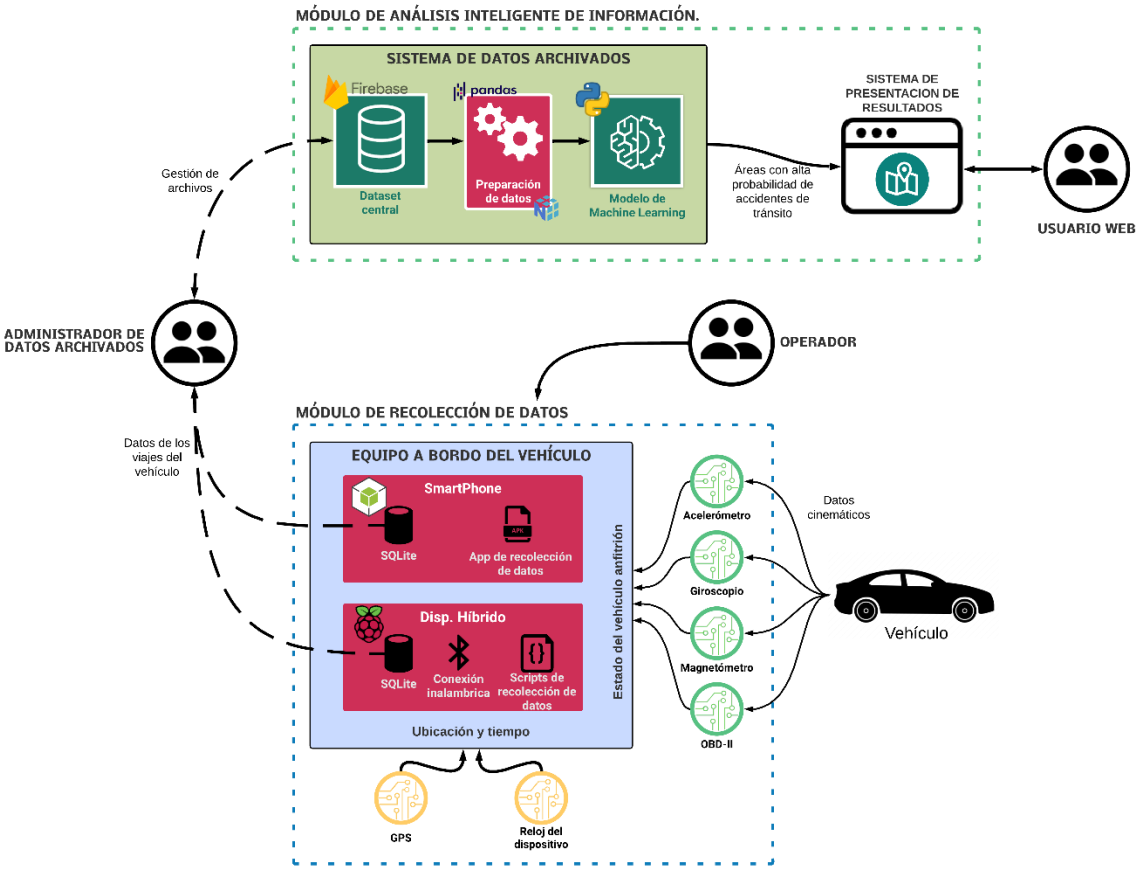

Figura 9. Prototipo propuesto para el SDIRC

Para el desarrollo del prototipo fue usada la metodología de desarrollo ágil SCRUM [14]. En la fase de inicio de esta metodología fue identificado el prototipo como la parte principal del caso de negoció con el fin de ser presentado a los interesados (o *stakeholders*) del proyecto, así como también para la construcción de la visión del proyecto. Cada funcionalidad fue descrita inicialmente como épicas y posteriormente como historias de usuario.

El desarrollo de todo el proyecto fue abordado en 3 *sprints*. Los documentos generados durante el desarrollo de los *sprints* de la metodología SCRUM son descritos en el **Anexo**

D. Desarrollo del prototipo del sistema mediante SCRUM. Este anexo contiene los documentos de la fase de inicio del proyecto y los documentos de planeación de cada uno de los *sprints* (en donde también se presentan los resultados del *Sprint* inmediatamente anterior).

Este capítulo también describe el diseño y desarrollo del *"data set",* el cual fue capturado a partir del módulo de recolección de datos y usado por el módulo de análisis inteligente de información. Finalmente, es descrito el pre-procesamiento, desarrollo y evaluación del modelo de ML siguiendo las fases de la metodología CRISP-DM [16].

5.1. Descripción del prototipo

A continuación, son descritos cada uno de los módulos; objetos físicos y funcionales; y flujos de información presentados en el prototipo del SDIRC (**Figura 9**).

5.1.1. Módulos y objetos físicos del prototipo

El prototipo cuenta con dos módulos: recolección de datos y análisis inteligente de la información. Cada uno de estos (incluyendo los objetos físicos que son contenidos por cada módulo) son explicados a continuación.

5.1.1.1. Módulo de recolección de datos (RD)

Este módulo cuenta con el *hardware* y *software* necesarios para la recolección y almacenamiento de los datos cinemáticos del vehículo. Este proceso abarca tres principales operaciones: la captura de las variables cinemáticas empleando los sensores correspondientes, la obtención de los datos de ubicación y tiempo, y el almacenamiento temporal de los datos recolectados.

El módulo RD (**Figura 10**), esta principalmente representado por el objeto físico denominado "Equipo a bordo del vehículo (EBV)". El EBV contiene tres objetos funcionales: Almacenamiento de los Datos Recolectados (EBV-ADR), Registro Cinemático del Vehículo (EBV-RCV) y Determinación de la Ubicación del Vehículo (EBV-DUV). El EBV es el objeto físico encargado de obtener los datos cinemáticos del vehículo anfitrión, los cuales son capturados a partir del uso de un acelerómetro, giroscopio, magnetómetro y OBD-II. Además, obtiene datos de ubicación y tiempo, obtenidos del GPS y reloj interno del dispositivo.

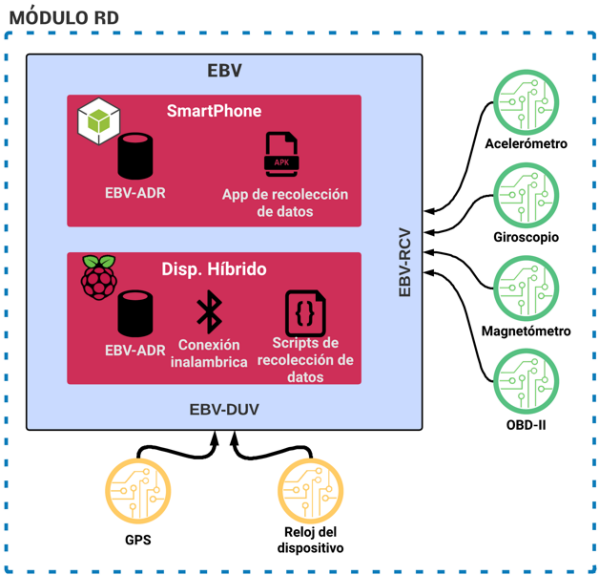

Figura 10 Módulo de recolección de datos del prototipo del SDIRC

Teniendo en cuenta los resultados del análisis de alternativas descritos en la sección 4.3, una tarjeta *Raspberry Pi 3* y un *Smartphone* con sistema *Android* son propuestos como los dispositivos a ser usados para la recolección, procesamiento inicial y almacenamiento de los datos vehiculares. Cada uno de estos dispositivos corresponde a una instancia diferente del EBV. Lo anterior permite evaluar el desempeño de cada uno de estos dispositivos y sus respectivos datos en la detección de *near-crashes.*

Aunque la literatura brinda diferentes perspectivas sobre las ventajas y desventajas del uso de uno u otro dispositivo, en la recolección de datos de ND, su desempeño y particularidades conocidas pueden verse comprometidas cuando el objetivo y contexto de su aplicación varían. Por lo anterior, fue conveniente emplear dos dispositivos diferentes para la recolección de datos (como fue mencionado en la sección 4.3). De esta forma, fue posible conocer la complejidad de desarrollo e implementación de cada dispositivo, las variaciones de los datos recolectados y comparar su desempeño. Todo lo anterior, con base en el contexto y alcance del presente proyecto.

Con el objetivo de definir un mecanismo de almacenamiento temporal de los datos recolectados, estandarizado y común para ambas instancias del EBV (*Rasberry Pi* y *Smartphone*); fue considerado el uso de un motor de base de datos basado en SQL. Este motor de base de datos constituye la base para el desarrollo del objeto funcional EBV-ADR.

5.1.1.2. Módulo de análisis inteligente de la información (AII)

Este módulo contiene los componentes necesarios para el procesamiento de la información recolectada, limpieza de datos, algoritmos de ML y presentación de resultados (**Figura 11**). El módulo está conformado por dos objetos físicos: Sistema de Datos Archivados (SDA), el cual cuenta con la lógica, algoritmos y *software* necesarios para el procesamiento de los datos; y el Sistema de Presentación de Resultados (SPR), donde son presentadas las

zonas con mayor probabilidad de ocurrencia de un AT, en función de los *near-crashes* detectados.

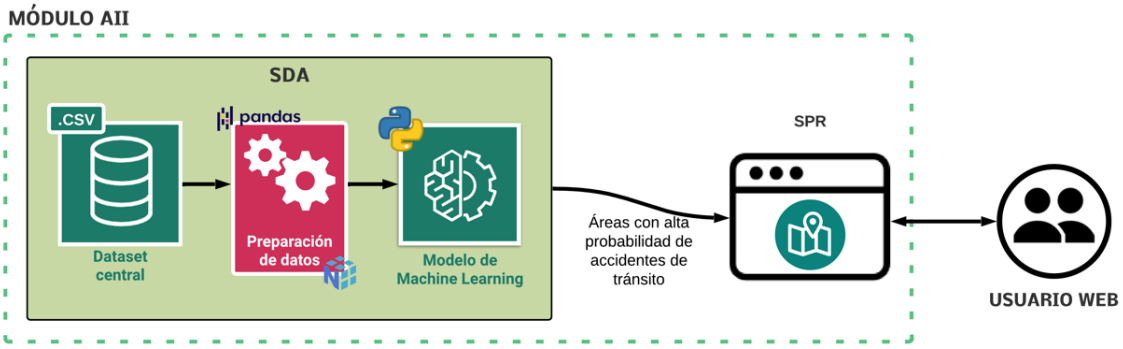

Figura 11. Módulo de análisis inteligente de la información del prototipo del SDIRC

El SDA es el encargado de procesar, analizar y aprender la información obtenida por el EBV. Para esto, toma como punto de partida un *"data set"* central que recopila los datos medidos de los vehículos empleados durante todos los trayectos realizados. Este *"data set"* es pre-procesado y post-procesado haciendo uso de librerías de *Python* adecuadas para estas tareas. Además, está constituido por un modelo de ML que detecta la ocurrencia de un *near-crash*. El SDA contiene tres objetos funcionales: *"data set"* Central (SDA-DC); Filtro y Procesamiento de Datos (SDA-FPD); y Análisis y Aprendizaje de Datos (SDA-AAD).

El SPR presenta los resultados obtenidos por el EBV y el SDA. Está conformado por una aplicación web donde es visualizada intuitivamente la información resultante de la implementación de las pruebas de campo y la salida del modelo de ML. La aplicación web esta implementada con tecnologías como HTML, CSS, JS y el *Backend* con *Flask* [53]. El uso de las anteriores tecnologías permite la presentación de un mapa de calor y un mapa de puntos de la respectiva ciudad, indicando las ubicaciones donde los diferentes *near-crashes* ocurrieron.

El prototipo incluye un objeto físico (actor) por fuera de los anteriores módulos, este es el Administrador de Datos Archivados (ADA). El ADA es el encargado de enviar los datos del módulo RD al AII. El proceso consta de dos pasos: 1) Extracción de los datos temporales de un viaje, los cuales están almacenados en la *Raspberry Pi* o el *Smartphone*. 2) Almacenar en una base de datos central todos los registros de los viajes con sus respectivos datos. En la **Figura 12** se presenta el proceso mencionado.

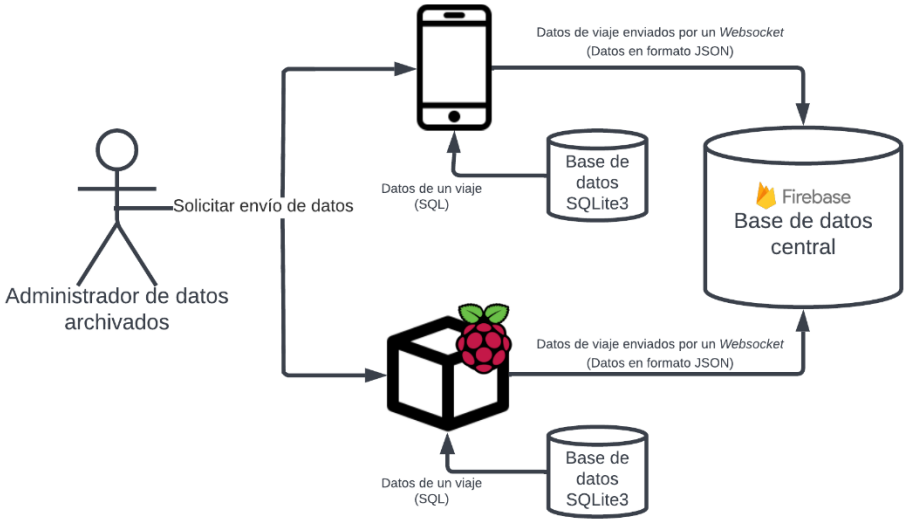

Figura 12. Diagrama para el envío de datos locales a la base de datos central del SDIRC

La base de datos utilizada en el prototipo se encuentra en la nube, concretamente corresponde a una *Realtime Database.* Esta, es una base de datos no relacional de *Firebase* [54]. *Firebase* es el BaaS (*Backend as a Service*) ofrecido por Google, el cual se ha convertido en una plataforma unificada que facilita el desarrollo de aplicaciones Android, iOS y web [55]. Esta base de datos permitió la transformación sencilla a un formato *Comma Separated Values* (CSV), para que la exploración en el módulo AII fuese más sencilla. El formato CSV es usado por su sencillez y su uso masivo en librerías de ML en lenguajes como *Python* [56], [57].

5.1.2. Objetos funcionales del prototipo

Para el prototipo se implementaron seis objetos funcionales, distribuidos en los objetos físicos explicados previamente. Estos objetos funcionales son descritos a continuación.

5.1.2.1. Almacenamiento de datos recolectados (EBV-ADR)

Corresponde a una base de datos *SQLite* instalada tanto en el *Smartphone* como en la tarjeta *Raspberry Pi* (como se muestra en la **Figura 10**). Esta base de datos tiene la función de almacenar los datos provenientes de: los sensores, el sistema OBD-II, el reloj interno del dispositivo y el GPS.

SQLite es una librería escrita en lenguaje C, que implementa un motor de base de datos SQL con muchas ventajas: es pequeño, rápido y confiable [58]. Además, el formato de archivo de base de datos usado por *SQLite* es estable, multiplataforma y soportado por múltiples lenguajes de programación [58]. Estas características y el formato de archivo fueron las razones para seleccionar *SQLite* para el desarrollo del EBV-ADR.

Complementariamente, este objeto funcional incluye un módulo de *software* encargado de recibir los datos cinemáticos, de ubicación y de tiempo, proporcionados por sus respectivas fuentes, además de realizar su debida lectura y escritura en la base de datos mencionada.

5.1.2.2. Registro cinemático del vehículo (EBV-RCV)

Este objeto funcional corresponde a un módulo de *software* encargado de leer las variables cinemáticas medidas por los sensores (acelerómetro, giroscopio y magnetómetro) y el sistema OBD-II en el caso de la *Raspberry*. Posteriormente, suministra estos datos al EBV-ADR, para ser persistidos en la base de datos. Este módulo de *software* es una parte de una aplicación móvil (en el caso del *Smartphone*) y de un programa elaborado en *Python* (en el caso de la tarjeta *Raspberry*).

5.1.2.3. Determinación de la ubicación del vehículo (EBV-DUV)

Software encargado de suministrar al EBV-ADR una marca de tiempo (Ts) correspondiente a cada registro de los datos del vehículo y la ubicación actual del mismo. Estos datos son obtenidos del reloj interno y del GPS incluidos en el EBV. Al igual que el EBV-RCV, forman parte de una aplicación móvil y de un programa elaborado en *Python* según la instancia de EBV.

5.1.2.4. *Data set* central (SDA-DC)

Corresponde a la instancia de *Firebase Realtime Database* y al conjunto de archivos en formato JSON (unidad de almacenamiento manejado por bases de datos no relacionales [59]) que contienen todos los datos recolectados en los diferentes vehículos, los cuales han sido previamente extraídos de sus respectivos EBV-ADR. Estos archivos son accedidos por el objeto funcional SDA-FPD descrito en la siguiente sección y representan el punto de partida para el análisis inteligente de la información.

Aunque la sincronización de datos en tiempo real es la característica principal de la base de datos seleccionada (como su nombre lo indica), esta no fue la razón principal de su elección. Sus diferentes optimizaciones y funcionalidades le otorgan una capacidad de respuesta adecuada para manejar cientos de miles de datos, como lo requiere el presente proyecto [59]. Además, su capa gratuita por uso de servicio es independiente del número de escrituras y lecturas de datos. Esto resulta favorable para un presupuesto limitado en la validación de un prototipo.

5.1.2.5. Filtro y preparación de los datos (SDA-FPD)

Este objeto funcional permite la limpieza y pre-procesamiento de los datos almacenados en el SDA-DC. Los datos son manipulados por la librería *Pandas* (del lenguaje *Python*) que permite reordenar, operar, limpiar datos perdidos, entre otras acciones [57]. Esta librería trabaja en conjunto con otras librerías que pueden ser usadas para la preparación de los datos como: *Numpy*, para los cálculos matemáticos y *Matplotlib* o *Plotly*, para la visualización de los datos [60], [61], [62].

5.1.2.6. Análisis y aprendizaje de datos (SDA-AAD)

Este objeto funcional es el encargado de analizar, procesar y aprender de los datos preparados por el SDA-FPD. En este objeto funcional se encuentra la implementación del

modelo de ML, cuya creación es realizada con la librería *ScikitLearn* [40]. El aprendizaje de este modelo tomó en cuenta lo siguiente: la representación o el espacio de hipótesis, la evaluación de la función objetivo y la optimización [63]. Los algoritmos seleccionados para el modelo de ML fueron: SVM, RF y DT; estos son descritos en la sección 2.4.2, su elección es explicada en la sección 4.3 y su rendimiento es evaluado en la sección 7.2. La salida obtenida al finalizar el proceso de aprendizaje es la clasificación binaria de un evento, como *near crash* o no *near-crash*.

5.1.3. Descripción de los flujos de información

A continuación, son descritos cada uno de los flujos que comunican a los objetos físicos entre sí.

5.1.3.1. Datos cinemáticos del vehículo

Son los valores de las variables cinemáticas usadas para la detección de *near-crashes* como velocidad, aceleración, desaceleración y tasa de rotación además de la intensidad de campo magnético. Estos valores son medidos y suministrados por los sensores que hacen parte del módulo RD. La frecuencia con la que los datos son obtenidos puede variar dependiendo de las tecnologías usadas en la implementación de un prototipo y a conveniencia de los objetivos del trabajo (para el caso del presente trabajo la frecuencia fue igual a 20 Hz y es especificada en la sección 6.1 de este documento). Estas variables son leídas por el EBV-RCV y posteriormente suministradas al EBV-ADR, el cual se encarga de almacenarlas en la base de datos *SQLite*.

5.1.3.2. Ubicación y tiempo

Corresponde a la ubicación geográfica actual del vehículo, incluyendo latitud y longitud; además de la marca de tiempo actual. Una vez estos datos son leídos por el EBV-DUV del reloj interno y el GPS, son suministrados al EBV-ADR para su respectivo almacenamiento en la base de datos *SQLite*. La frecuencia con la que los datos son obtenidos puede variar dependiendo de las tecnologías usadas en la implementación de un prototipo y a conveniencia de los objetivos del trabajo (para el caso del presente trabajo la frecuencia fue igual a 20 Hz y es especificada en la sección 6.1 de este documento).

5.1.3.3. Datos de viajes del vehículo

Representa el conjunto de todos los registros de las variables cinemáticas, de ubicación y del tiempo que han sido almacenados en la base de datos *SQLite* (EBV-ADR) durante uno o más trayectos realizados por un vehículo. Estos datos son extraídos por el ADA, quien los transfiere posteriormente al SDA-DC.

5.1.3.4. Gestión de archivos

Está conformado por dos flujos de información diferentes. El primer flujo corresponde a las solicitudes de gestión de archivo, ejecutadas por el ADA, las cuales pueden ser: solicitudes de importación de los datos de viajes de un vehículo provenientes de su respectivo EBV-

ADR, la ejecución del pre-procesamiento o procesamiento de estos datos, como los procesos de conversión de estos datos del formato de archivo usado por *SQLite* a un formato CSV; la integración de los datos pertenecientes a diferentes vehículos en un solo conjunto de datos; pre-procesamiento y limpieza de estos conjuntos de datos; y a la ejecución de los procesos clasificación de los algoritmos de ML. El segundo flujo, datos de gestión de archivo, corresponde a los resultados a ser presentados al ADA una vez cualquiera de los procesos anteriormente mencionados ha finalizado.

5.1.3.5. Solicitud de análisis de archivos

Este flujo corresponde a la extracción de los "*data sets"* de ND de los diferentes viajes realizados. Estos inicialmente son extraídos del SDA-DC por el SDA-FPD y posteriormente transformados al formato de datos tabulares bidimensionales usado por la librería *Pandas*. Esta transformación permite al SDA-FPD realizar el respectivo pre-procesamiento. Además, los datos suministrados son transformados a los diferentes formatos requeridos por el SDA-AAD.

5.1.3.6. Resultado de análisis de archivos

Corresponde a los datos preparados enviados por el SDA-FPD al SDA-AAD. Este último, usa dichos datos para el entrenamiento de los clasificadores de ML y para la clasificación de los eventos de interés. Además, este flujo incluye los resultados de la clasificación realizada por el modelo de ML los cuales son almacenados en el SDA-DC para ser persistidos y visualizados en el SPR según la demanda de los usuarios.

5.1.3.7. Áreas con alta probabilidad de accidentes

Este flujo de información parte de los resultados de la clasificación realizada por el modelo de ML, los cuales están almacenados en el SDA-DC y son extraídos nuevamente por el SDA-AAD, para realizar su respectiva transformación y enviarlos en un formato que pueda ser visualizado por el usuario. La información contenida en estos datos corresponde a los *near-crashes* detectados, incluyendo su ubicación dentro de la ciudad, su fecha y hora de ocurrencia y su duración.

5.2. Descripción del "*data set"*

A partir del análisis y selección de alternativas realizados en las secciones 4.2 y 4.3, fueron establecidas las variables que utilizó el SDIRC como punto de partida. Estas variables corresponden a las columnas de los "*data set*" generados por los dispositivos del módulo RD.

Debido a que el módulo de RD comprende dos instancias diferentes (*Smartphone* y dispositivo híbrido controlado por la tarjeta *Raspberry* Pi 3 B+), cada uno de estos dispositivos incluyó un "*data set"* diferente. La diferencia entre estos "*data set"* no reside únicamente en el dispositivo donde fueron generados y almacenados, también reside en la interfaz de lectura de sistema OBD-II. Lo anterior hizo que el dispositivo híbrido obtuviera

datos diferentes en la lectura de variables, así como una columna adicional en el "*data set"*. Específicamente las columnas afectadas son velocidad y posición del pedal acelerador. No obstante, se debe tener en cuenta que al tener dispositivos diferentes los datos de las variables, medidas en un instante de tiempo determinado, pueden diferir.

Los "*data sets"* para cada uno de los dos dispositivos del módulo de RD son descritos detalladamente en la **Tabla 7** y **Tabla 8**, donde se especifica el nombre de cada columna, una descripción de esta y su tipo de dato.

5.2.1. *Data set* dispositivo *Smartphone*

A continuación, la **Tabla 7** presenta en detalle cada una de las variables (columnas o *features*) que contiene el "*data set"* para el dispositivo *Smartphone.* Los valores de cada columna pertenecientes a una misma fila corresponden a los datos capturados en un instante de tiempo determinado.

Tabla 7. *Data set Smartphone*

Nombre de columna	Descripción	Tipo
ID dato (*id*)	El identificador único del registro del dato actual (auto incremental).	Numérico
ID viaje (*idTrip*)	El identificador único del registro del viaje (auto incremental).	Numérico
Vehículo (*idVehicle*)	Descripción del vehículo a usar.	Texto
Ruta (*route*)	Descripción de la ruta a realizar.	Texto
Instante de tiempo (*timestamp*)	El momento del tiempo en el cual está siendo almacenada una nueva fila.	*Timestamp*
Velocidad (*speed*)	Rapidez de desplazamiento del vehículo (medido en Km/h).	Numérico
Aceleración X (*accX*)	Aceleración del vehículo en el eje X (medido en m/s^2).	Numérico
Aceleración Y (*accY*)	Aceleración del vehículo en el eje Y (medido en m/s^2).	Numérico
Aceleración Z (*accZ*)	Aceleración del vehículo en el eje Z (medido en m/s^2).	Numérico
Velocidad angular X (*velAngX*)	Movimiento rotacional del vehículo sobre su eje X (medido en rad/s).	Numérico
Velocidad angular Y (*velAngY*)	Movimiento rotacional del vehículo sobre su eje Y (medido en rad/s).	Numérico
Velocidad angular Z (*velAngZ*)	Movimiento rotacional del vehículo sobre su eje vertical (medido en rad/s).	Numérico
Magnetómetro X (*magX*)	Fuerza de la intensidad magnética con respecto al eje X (medido en micro teslas µT).	Numérico
Magnetómetro Y (*magY*)	Fuerza de la intensidad magnética con respecto al eje Y (medido en micro teslas µT).	Numérico
Magnetómetro Z (*magZ*)	Fuerza de la intensidad magnética con respecto al eje Z (medido en micro teslas µT).	Numérico

Latitud (*latitude*)	Coordenada geográfica del GPS con respecto al norte.	Numérico
Longitud (*longitude*)	Coordenada geográfica del GPS con respecto al este.	Numérico
Clase (*eventClass*)	Muestra el evento captado en la prueba, por medio de los números 1 y 0, donde 1 es la ocurrencia de un *near-crash* y 0 el estado de conducción normal.	Booleano

5.2.2. *Data set* dispositivo híbrido

A continuación, la **Tabla 8** detalla cada una de las características (columnas o *features*) que contiene el "*data set"* para el dispositivo híbrido. Para este caso el EBV-RCV contó con un sensor adicional, el dispositivo ELM-327 conectado a la interfaz OBD-II, que añadió una nueva columna al "*data set*".

Tabla 8. *Data set* dispositivo híbrido

Nombre de columna	Descripción	Tipo
ID dato (*id*)	El identificador único del registro del dato actual (auto incremental).	Numérico
ID viaje (*idTrip*)	El identificador único del registro del viaje (auto incremental).	Numérico
Vehículo (*idVehicle*)	Descripción del vehículo a usar.	Texto
Ruta (*route*)	Descripción de la ruta a realizar.	Texto
Instante de tiempo (*timestamp*)	El momento del tiempo en el cual está siendo almacenada una nueva fila.	*Timestamp*
Velocidad (*speed*)	Rapidez de desplazamiento del vehículo, obtenida a través del OBD-II (medido en Km/h).	Numérico
Aceleración X (*accX*)	Aceleración del vehículo en el eje X (medido en m/s^2).	Numérico
Aceleración Y (*accY*)	Aceleración del vehículo en el eje Y (medido en m/s^2).	Numérico
Aceleración Z (*accZ*)	Aceleración del vehículo en el eje Z (medido en m/s^2).	Numérico
Velocidad angular X (*velAngX*)	Movimiento rotacional del vehículo sobre su eje X (medido en rad/s).	Numérico
Velocidad angular Y (*velAngY*)	Movimiento rotacional del vehículo sobre su eje Y (medido en rad/s).	Numérico
Velocidad angular Z (*velAngZ*)	Movimiento rotacional del vehículo sobre su eje vertical (medido en rad/s).	Numérico
Magnetómetro X (*magX*)	Fuerza de la intensidad magnética con respecto al eje X (medido en micro teslas µT).	Numérico
Magnetómetro Y (*magY*)	Fuerza de la intensidad magnética con respecto al eje Y (medido en micro teslas µT).	Numérico
Magnetómetro Z (*magZ*)	Fuerza de la intensidad magnética con respecto al eje Z (medido en micro teslas µT).	Numérico
Posición del pedal acelerador (*accPosition*)	Porcentaje que indica la porción del pedal acelerador que está siendo presionada por el conductor, obtenida a través del OBD-II.	Porcentaje

Latitud (*latitude*)	Coordenada geográfica del GPS con respecto al norte.	Numérico
Longitud (*longitude*)	Coordenada geográfica del GPS con respecto al este.	Numérico
Clase (*eventClass*)	Muestra el evento captado en la prueba, por medio de los números 1 y 0, donde 1 es la ocurrencia de un *near-crash* y 0 el estado de conducción normal.	Booleano

5.3. Desarrollo del prototipo del SDIRC

5.3.1. Desarrollo del módulo de recolección de datos

Teniendo en cuenta el prototipo diseñado, inicialmente se identificó que el desarrollo del módulo de RD incluía la implementación de tres objetos funcionales: EBV-ADR, EBV-RCV, EBV-DUV. Además, estos objetos funcionales interactuaban entre sí a través de tres flujos de información diferentes como son: los datos cinemáticos del vehículo, su ubicación y tiempo, y los datos de viajes del vehículo.

A partir de lo anterior y con base en las épicas, historias de usuarios y tareas identificadas con la metodología SCRUM (ver **Anexo D. Desarrollo del prototipo del sistema mediante SCRUM**), fueron descritas las tareas realizadas durante el desarrollo del módulo RD. Estas fueron:

- Desarrollar un módulo *software* e implementar el módulo *hardware* encargados de obtener los datos cinemáticos del vehículo.
- Desarrollar un módulo *software* e implementar el módulo *hardware* encargados de obtener la ubicación y tiempo.
- Desarrollar una interfaz de usuario (UI) que permita la activación y desactivación del proceso de recolección de datos y la lectura en tiempo real de los datos que están siendo capturados.
- Modificar la UI, de modo que permita ajustar ciertas preferencias como la frecuencia de recolección de datos, la ruta a ser recorrida y el vehículo en el cual se hace el recorrido.
- Desarrollar una UI y su respectivo componente *software* para la gestión de los recorridos realizados, incluyendo la visualización de su historial, eliminación y extracción.

Como fue mencionado en la sección 5.1.1, en el prototipo diseñado fueron desarrolladas dos instancias diferentes del EBV, este es el componente central del módulo RD. Estas dos instancias correspondieron a: un *Smartphone*, basado en una aplicación móvil para *Android*, y un dispositivo híbrido, basado en una *Raspberry Pi*; las dos tuvieron un funcionamiento similar, pero fueron independiente entre sí. Es decir, las tareas de desarrollo anteriormente mencionadas fueron realizadas tanto para el *Smartphone,* como para el

dispositivo híbrido. De este modo, cada dispositivo logró cumplir con las funcionalidades requeridas, a través del uso de diferentes tecnologías y dispositivos electrónicos complementarios.

Básicamente, las funcionalidades del módulo RD fueron clasificadas en dos grupos: las relacionadas con el control del proceso de recolección de datos y las relacionadas con la gestión de los datos recolectados en los viajes. Por lo anterior, fueron implementadas dos interfaces diferentes en cada instancia del módulo RD, cada una de ellas orientada a brindar al operador de este módulo el acceso a un grupo de funcionalidades.

En este sentido, **la Figura 13a y Figura 13b** muestran el desarrollo final de la aplicación móvil del *Smartphone*. La primera figura muestra la interfaz que permite al operador configurar, iniciar, monitorear y detener el proceso de recolección de datos; y la segunda muestra la interfaz que permite ver los viajes o trayectos realizados, enviar los datos de un viaje a la base de datos central o eliminarlos de la base de datos local.

Figura 13. a) Interfaz de recolección de datos del *Smartphone*, b) Interfaz de gestión de datos del *Smartphone*.

Por otra parte, el desarrollo final de la aplicación del dispositivo híbrido puede apreciarse en la **Figura 14a** y la **Figura 14b,** estas muestran las interfaces que contiene el dispositivo. Estas interfaces fueron accedidas a través de un servidor web desarrollado con el fin de brindar al operador mecanismo más intuitivo para controlar este dispositivo. Las funcionalidades de estas interfaces fueron similares a las de la aplicación móvil del *Smartphone* mencionadas anteriormente.

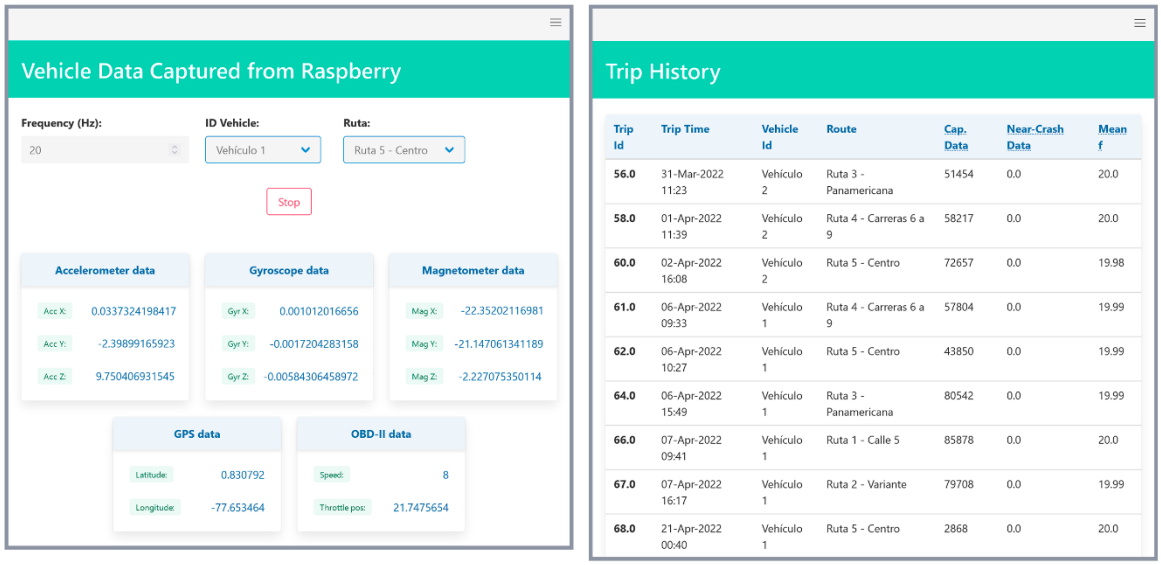

Figura 14. a) Interfaz de recolección de datos del dispositivo híbrido, b) interfaz de gestión de datos del dispositivo híbrido

Adicionalmente, en el dispositivo hibrido fue necesario realizar un desarrollo de tipo *hardware*. Este proceso consistió básicamente en la integración y conexión de todos los componentes y sensores necesarios para cumplir a cabalidad con las funcionalidades requeridas. Su componente central encargado de la lectura, pre-procesamiento, gestión y almacenamiento de los datos recolectados corresponde a una tarjeta Raspberry Pi 3 B+. A ella fueron conectados:

- Un escáner OBD-II ELM 327 con conexión Bluetooth fue empleado para la lectura de las variables velocidad y posición del pedal acelerador.
- Un módulo *multi-chip* MPU 9250 que contiene una IMU y un magnetómetro. Utilizado para leer los valores cinemáticos del vehículo (aceleración, velocidad angular e intensidad de campo magnético, en cada uno de sus ejes “X”, “Y”, y “Z”) [64]. Este módulo fue conectado usando los pines SDA (GPIO2) y SCL (GPIO3) correspondientes a la interfaz de comunicación I^2C de la *Raspberry*.
- Un módulo GPS de referencia Neo6M. Utilizado para obtener la localización geoespacial del dispositivo híbrido (latitud y longitud). Este fue conectado a través de los pines TXD (GPIO14) y RXD (GPIO15) de la *Raspberry*.
- Un pulsador eléctrico, conectado a un pin GPIO de la *Raspberry* configurado como entrada. Este fue utilizado para el etiquetado de datos durante las pruebas de recolección de datos para el entrenamiento del algoritmo. Su funcionalidad es descrita en la sección 6.1.

El desarrollo hardware anteriormente mencionado se puede apreciar de forma resumida en la **Figura 15** y se puede evidenciar en la **Figura 16**, mostradas a continuación:

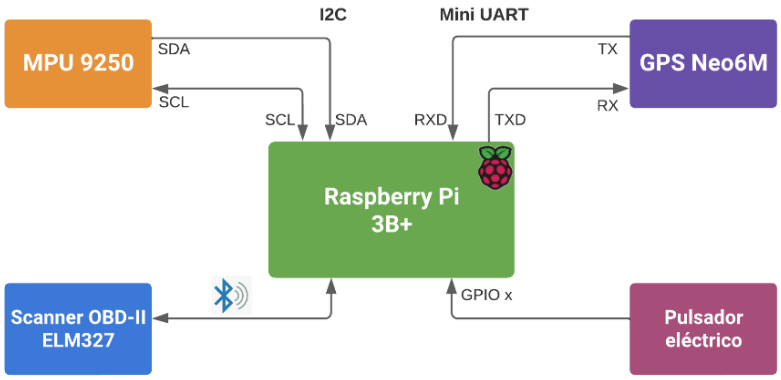

Figura 15. Diagrama de bloques dispositivo híbrido

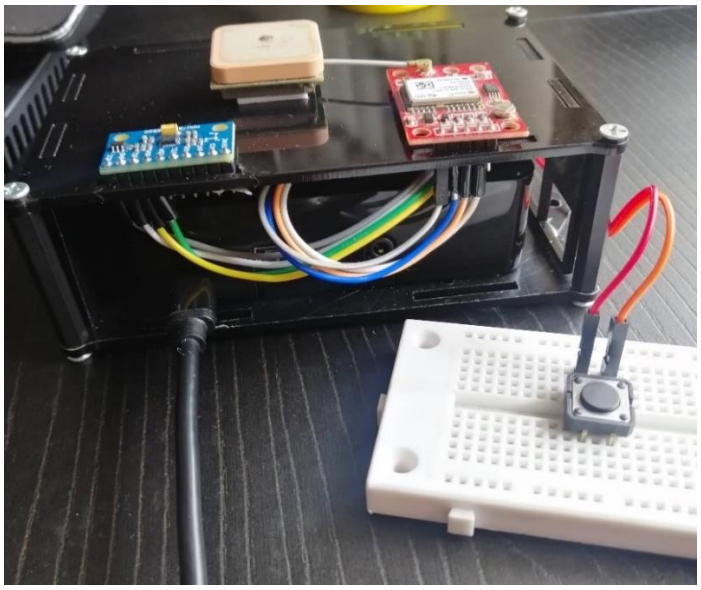

Figura 16. Dispositivo híbrido del SDIRC

El proceso de desarrollo detallado, que incluye una descripción más completa de las herramientas, tecnologías y dispositivos empleados en la implementación de las dos instancias del EBV puede ser consultada en el **Anexo E. Desarrollo del prototipo del SDIRC**.

Igualmente, los códigos tanto de la aplicación móvil desarrollada como del componente software del dispositivo híbrido fueron almacenado en repositorios de GitHub especificados en el **Anexo F. Aplicación móvil para la recolección de datos** y **Anexo G. Aplicación para la recolección de datos en el dispositivo híbrido.**

5.3.2. Sistema de coordenadas para el SDIRC

En los artículos mencionados en la revisión de la literatura (sección 3.1) fueron encontradas variaciones en el sistema de coordenadas que manejaban los dispositivos de recolección de datos. Por lo tanto, fue necesario establecer una convención en el sistema de coordenadas de los sensores IMU que contienen el *Smartphone* y el dispositivo híbrido, así como el sistema de coordenadas que usa el vehículo.

Habitualmente las IMU cuentan con 3 sensores, cada uno con tres ejes distintos “X”, “Y” y “Z” apuntando a un sistema de coordenadas establecido por el desarrollador o fabricante. Los sistemas de coordenadas del dispositivo smartphone e híbrido pueden consultarse con mayor detalle en el **Anexo E. Desarrollo del prototipo SDIRC.**

Teniendo en cuenta los sistemas de coordenadas de ambos dispositivos, así como las convenciones de referencia mundial para definir la orientación de cuerpos rígidos [65], fue posible definir la convención ENU (East, Norte y Arriba) como el sistema de coordenadas a usar en el desarrollo del EBV-RCV. Además, este sistema estableció la posición en la cual fueron ubicados los dispositivos a bordo del vehículo, ya que era necesario que la pantalla del *Smartphone* estuviera apuntando hacia arriba (permitiendo controlar la aplicación adecuadamente). En la **Figura 17** es presentado un automóvil en movimiento y un dispositivo móvil con orientación horizontal (el dispositivo móvil representa a la MPU-9250 y el *Smartphone*, debido a que los dos dispositivos pueden ser ubicados en la misma forma). Las flechas marcadas con rojo representan los ejes del acelerómetro, así como los ejes del magnetómetro; las flechas azules representan únicamente los ejes del giroscopio.

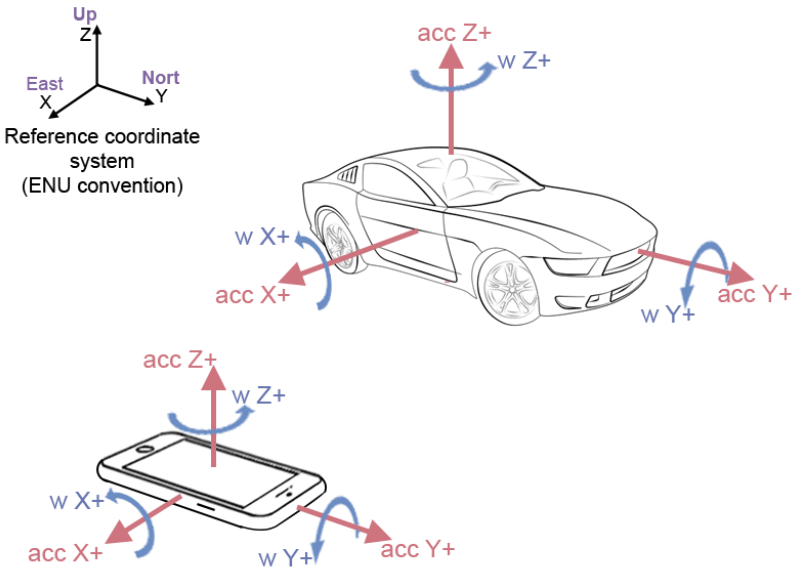

Figura 17. Sistema de coordenadas definido para el SDIRC

5.3.3. Desarrollo del módulo de análisis inteligente de información

El desarrollo del AII tuvo en cuenta todos los objetos físicos, objetos funcionales y flujos de información relacionados con este módulo (sección 5.1). Cada una de estas partes abarca

uno o varios requisitos necesarios para cumplir con la totalidad del desarrollo. Cada requisito es especificado en el **Anexo D. Desarrollo del prototipo del sistema mediante SCRUM** a través de historias de usuario y épicas.

En general las tareas detectadas para el desarrollo de este módulo fueron:

- Desarrollar una plataforma web que permita a un administrador de datos consultar y gestionar la información recolectada y enviada a la base de datos central por los módulos RD durante los viajes.
- Desarrollar una plataforma web que permita visualizar los datos recolectados, observando medidas estadísticas que permitan entender y familiarizarse con los datos.
- Desarrollar un modelo clasificador de ML haciendo uso de la metodología CRISP-DM que permita determinar eventos de *near-crash*.
- Desarrollar una web que a través de un mapa permita observar los eventos de *near-crash* detectados por el algoritmo.

Completar el desarrollo de cada uno de los requisitos permitió construir una plataforma web (denominada ICRDS web por las siglas en inglés *Intelligent Collision Risk Detection System*) que contiene dos componentes: subsistema para la gestión y análisis de los datos de viajes y el subsistema de visualización de resultados. El código fuente puede ser consultado en el **Anexo H. Plataforma ICRDS web**.

5.3.3.1. Subsistema para la gestión y análisis de los datos de viajes

Los componentes de este subsistema permiten al administrador de datos realizar: la lectura de la base de datos central (SDA-DC) o *Firebase Realtime Database*, el proceso de filtrado y pre-procesamiento de los datos (SDA-FDP) y la ejecución del modelo de ML para clasificación de los eventos de conducción (SDA-AAD). Cada una de las funcionalidades son ejecutadas por la plataforma web a través de su *Backend*.

La **Figura 18** presenta la interfaz que permite administrar los datos recolectados en los trayectos vehiculares. Para ello cuenta con las siguientes funciones: filtros por fecha, vehículo y ruta (trayecto); visualización de los datos recolectados por el *Smartphone* o por el dispositivo híbrido (*Raspberry*) a través de pestañas; eliminación de los datos de un trayecto o viaje; y exploración en detalle los datos de los trayectos.

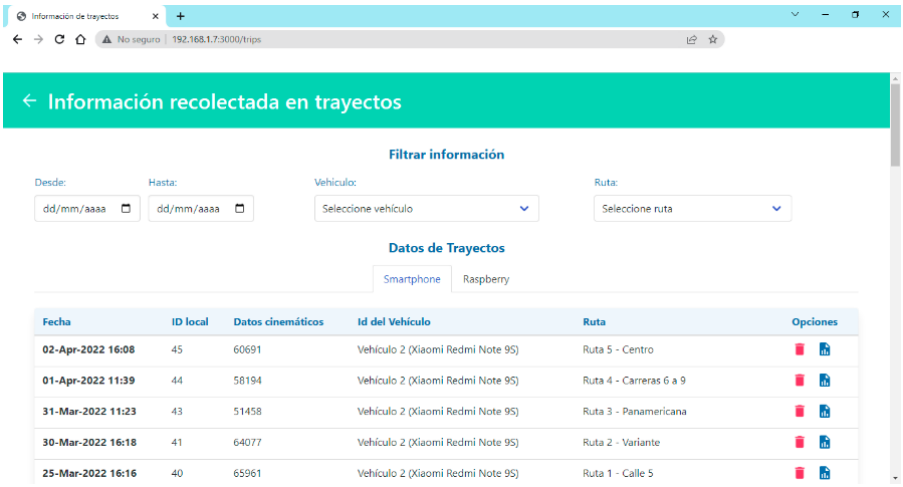

Figura 18. Interfaz de información de los trayectos cargados a la base de datos central del SDIRC

La **Figura 19** muestra la interfaz que contiene la información detallada y permite la exploración y visualización de los datos de un viaje. En esta se visualiza la hora, dispositivo, datos capturados, el vehículo, la ruta y el identificador de dicho viaje.

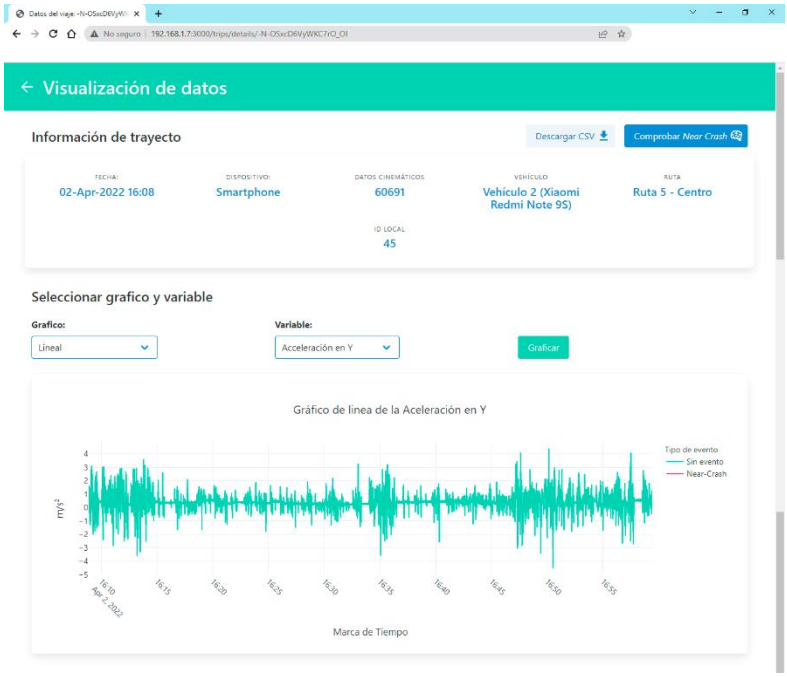

Figura 19. Interfaz de información de los datos de trayectos recolectados en el SDIRC

Además de lo anterior, hay una sección dentro de esta interfaz que permite visualizar gráficas que ayudan a tener un mayor entendimiento de los datos. Estas graficas son: grafica lineal, histograma, grafico de pastel y matriz de dispersión (**Figura 20a ,Figura 20b, Figura 20c, Figura 20d**; respectivamente). Por medio de esta interfaz es posible visualizar cada una de las características capturadas por el módulo RD (características seleccionadas en el "*data set"* sección 5.2).

El gráfico lineal es una serie de puntos conectados por segmentos de líneas rectas, estos son usados normalmente para rastrear los cambios temporales de un proceso o comparar trayectorias del espacio o el estado [66]. El eje Y corresponde a la característica seleccionada en el campo de entrada “variable” y el eje X corresponde al *timestamp* del recorrido.

El histograma es usado para mostrar la distribución de probabilidad de un conjunto de datos, son construidos usando un número de *bins* (la división del eje X en un rango de números representada por una barra) que cubre el rango de valores del conjunto de datos [66].

El grafico de pastel o torta permite apreciar los porcentajes o proporciones de los dos tipos de eventos a los cuales pueden corresponder los datos recolectados: evento de conducción normal o *near-crash* [66]. Este gráfico, resultó particularmente útil durante la fase de entrenamiento del modelo de ML, ya que permitía visualizar si el “*data set*” contenía clases no balanceadas.

La matriz de dispersión es una gráfica que brinda una forma de explorar relaciones entre múltiples variables. Esto lo hace a través de una cuadricula o matriz de gráficos, donde cada variable es comparada con cada una de las otras variables [29].

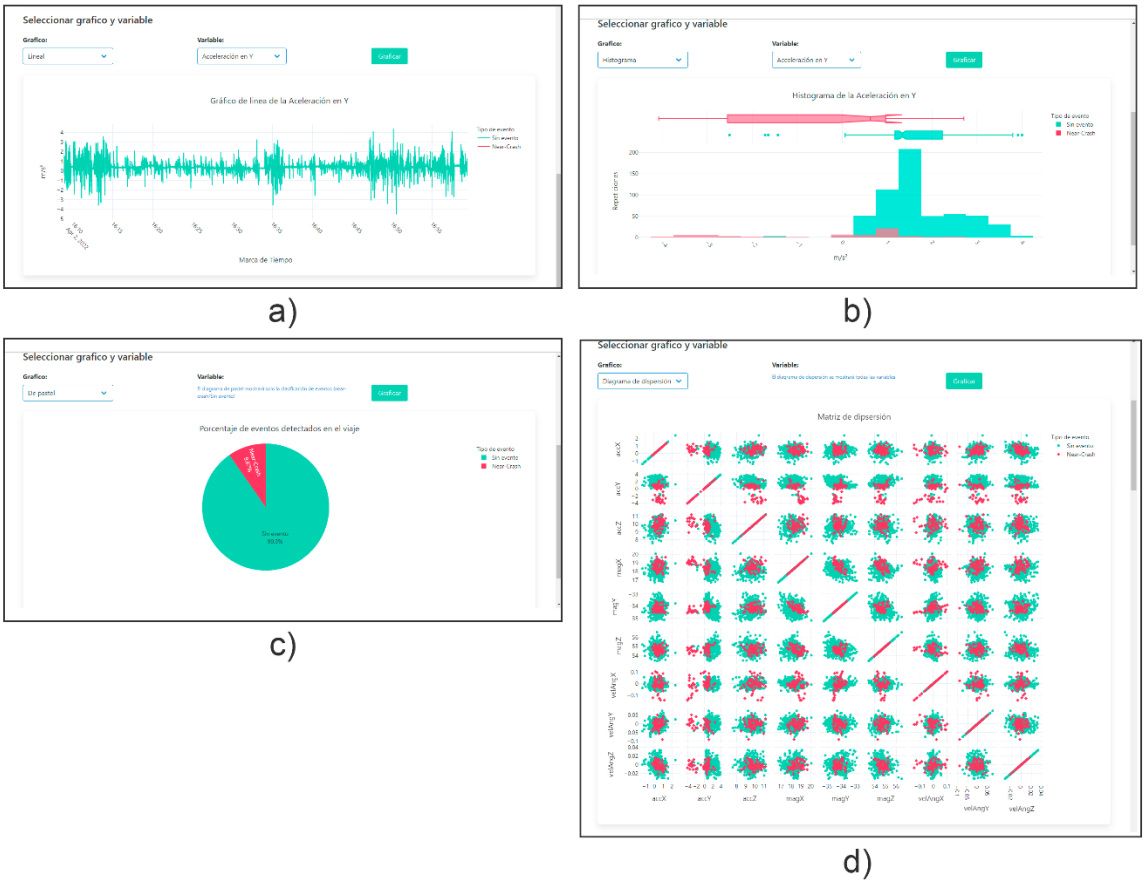

Figura 20. Visualización de diagramas de datos del SDIRC: a) grafico lineal, b) histograma, c) grafico de pastel, d) matriz de dispersión

Las dos últimas funcionalidades disponibles en este subsistema son: la descarga de los datos de un viaje en formato CSV y la comprobación de *near-crashes*. Esta última es de las más importantes ya que permite a un administrador de los datos iniciar el proceso de análisis de los datos recolectados durante un viaje. Durante este proceso, dichos datos son suministrados al modelo de ML, el cual realiza la clasificación de los mismo en busca de los eventos de *near-crash* (sección 5.4.6).

El resultado del proceso de comprobación de *near-crash* es igualmente almacenado en la base de datos central y contiene información relevante sobre cada evento detectado. Esta incluye su ubicación (latitud y longitud), su fecha y hora de ocurrencia, el dispositivo con el que fueron recolectados los datos, la ruta a la que pertenece y cada uno de los registros que conforman el *near-crash*. Sin embargo, el usuario únicamente puede observar la lista de los *near-crash* detectados con su ubicación, fecha y hora de ocurrencia (**Figura 21**).

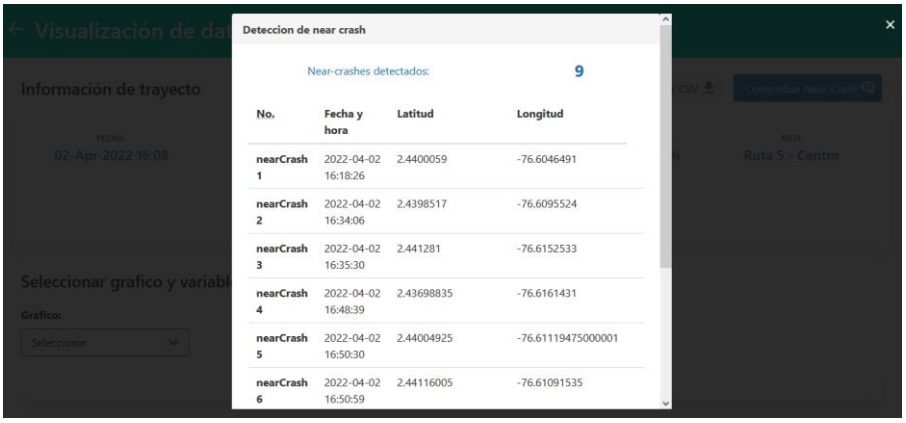

Figura 21. Resultados de un proceso de comprobación de *near-crash* en el SDIRC desarrollado

5.3.3.2. Subsistema de visualización de resultados

El principal objetivo de este subsistema es presentar a un usuario los resultados de todo el proceso de análisis de datos recolectados, principalmente del proceso de comprobación de *near-crashes,* de forma más representativa y concorde al objetivo general del sistema propuesto. Esto es logrado haciendo uso de un mapa de calor (**Figura 22**) y un mapa de puntos (**Figura 23**) en los cuales son localizados los diferentes *near-crashes* detectados, permitiendo apreciar la distribución de su ocurrencia sobre diferentes zonas de la ciudad de estudio. Además, brinda al usuario la opción de elegir los datos que son mostrados en el mapa mediante la selección del dispositivo y una ruta en específico.

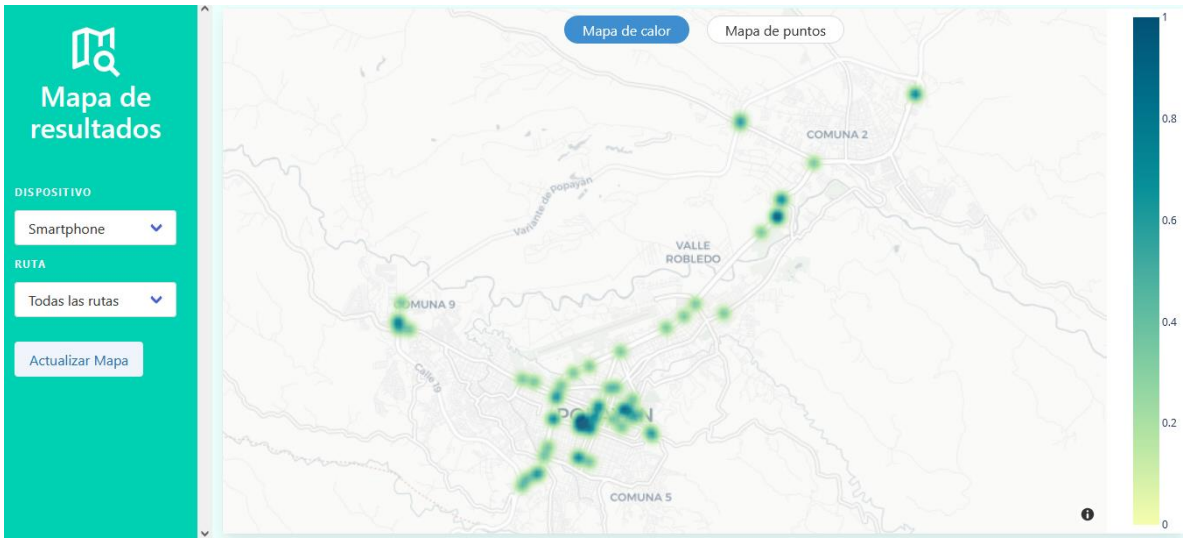

Figura 22. Ejemplo de mapa de calor del subsistema de visualización de resultados

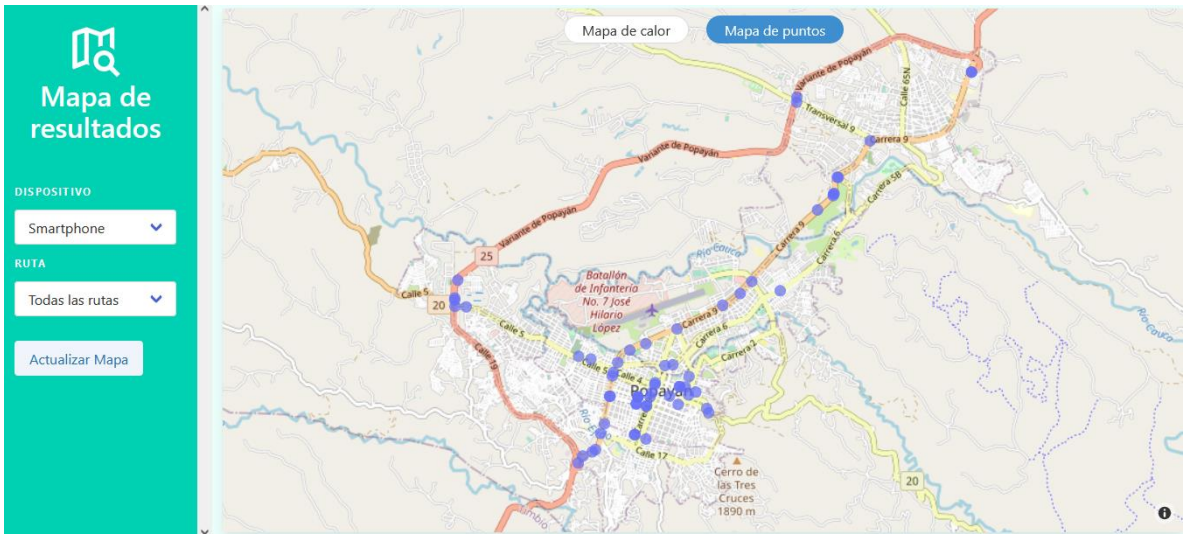

Figura 23. Ejemplo de mapa de puntos del subsistema de mapa de puntos

5.4. Desarrollo del modelo de ML

Como fue mencionado en la sección 1.4.3, el desarrollo del modelo de ML tuvo como base el uso de la metodología CRISP-DM [15]. Esta metodología tiene 6 fases que definieron el desarrollo del modelo de ML usado por los objetos funcionales SDA-FPD y SDA-AAD del módulo AII.

5.4.1. Fase I: Comprensión del negocio

En esta fase fueron determinados los objetivos a lograr con el desarrollo del modelo de ML, los resultados y plan previstos que permitieron el logro de los objetivos. Además, son descritas las herramientas a usar para la manipulación de los datos y el proceso de construcción del modelo de ML [16].

Inicialmente fue descrito el contexto de la problemática que debía abordar el algoritmo. En este caso, las secciones 1.1 y 1.2 exponen el trasfondo de la situación. Por lo tanto, el objetivo a lograr con el desarrollo del modelo de ML fue: “Identificar por medio de la cinemática vehicular zonas en donde puedan presentarse eventos de conducción relacionados con los *near-crashes*”.

Para lograr cumplir con este objetivo fue necesario revisar el estado actual de conocimiento (secciones 3.1 y 3.2) identificando la forma en la que muchos trabajos determinaban los eventos de accidentalidad. De esta manera fueron encontrados trabajos como el de [26] y [34] en el cual son reconocidos aquellos comportamientos de conducción que pueden determinar un AT; estos comportamientos son definidos como maniobras agresivas de giros, aceleraciones o frenadas. Otros trabajos como [39] y [67] determinan los eventos de accidentalidad con base a la acción del freno de un vehículo. Por último, trabajos como [12] y [23] analizan los posibles AT a través de eventos cinemáticos poco comunes y condiciones externas al vehículo.

Debido a la limitación de los datos obtenidos por el SDIRC muchas de las condiciones externas al vehículo no fueron consideradas para lograr el objetivo del modelo de ML. Así que para lograr alcanzar el desarrollo del objetivo fue necesario determinar eventos de conducción en los que se presenten maniobras agresivas o poco comunes relacionadas con los eventos de *near-crash*. En este caso, las maniobras fueron: **aceleración repentina, frenada repentina, giro derecho e izquierdo agresivo y cambio de línea agresivo**. Estas maniobras fueron ejecutadas por un conductor designado y habilitado para realizar este tipo de eventos, los detalles sobre esta prueba se presentan en la sección 6.1.

El plan para cumplir con el objetivo del desarrollo del modelo de ML consistió en los siguientes pasos:

- Extraer datos donde se presenten las maniobras de conducción anteriormente mencionadas.
- Limpiar y filtrar los datos obtenidos al ejecutar las maniobras de conducción.
- Seleccionar aquellas características (*features* del “*data set”*) que mejor representen cada maniobra realizada.
- Realizar el proceso de extracción de las características para determinar el “*data set”* final de entrada del algoritmo clasificador.
- Entrenar y optimizar cada uno de los clasificadores para seleccionar aquel que obtenga el mejor rendimiento clasificando las maniobras de conducción.
- Seleccionar el clasificador que obtuvo mejor rendimiento para generar el modelo de ML que permita la detección de las diferentes maniobras.
- Desplegar el clasificador en la plataforma ICRDS web.

Este plan fue ejecutado utilizando herramientas que permitieron el análisis de datos, manipulación y construcción de modelos de ML. El lenguaje seleccionado para este desarrollo fue *Python* debido a su gran ecosistema de paquetes dedicado a actividades relacionadas con la ciencia de datos (en las secciones 5.1.2.5 y 5.1.2.6 son especificados los paquetes usados en el desarrollo del modelo de ML). La creación de los códigos dedicados a la limpieza, filtrado, revisión y generación del modelo fue realizada a partir de la herramienta de computación interactiva *Jupyter Notebook* [68].

Finalmente, en esta fase fueron descritos los algoritmos, técnicas y métricas utilizadas en la creación del modelo de ML. Los algoritmos clasificadores seleccionados para el modelo de ML fueron SVM, DT y RF (las secciones 2.4.2, 4.2.3 y 4.3 detallan la definición y la razón de su selección). La selección y extracción de características tuvieron como base los documentos [26] y [34] donde se especifican las características que definen a cada una de las maniobras, así como la técnica de la ventana deslizante para la extracción de datos (las secciones 5.4.2 y 5.4.3 detallan el proceso de selección y extracción de características). Por último, las métricas utilizadas para evaluar los algoritmos fueron: "*precision"*, "*recall"*, *"F1-score"* y "AUC" (los detalles sobre estas métricas se encuentran en la sección 2.4.3).

5.4.2. Fase II: Comprensión de los datos

En esta fase fueron realizadas las actividades para la familiarización de los datos, estas consistieron en: comprender como los datos fueron adquiridos, analizar el "*data set"* inicial, explorar los datos a través de técnicas de visualización, descubrir información de interés y verificar la calidad en los datos [16].

Los datos para la creación del modelo de ML fueron adquiridos a través de la ejecución de pruebas controladas de las maniobras de conducción anormales (aceleración repentina, frenada repentina, giro derecho e izquierdo agresivo y cambio de línea agresivo); la sección 6.1 detalla la ejecución de estas pruebas. En este sentido el módulo de recolección de datos tanto del dispositivo *Smartphone* como del dispositivo híbrido fueron los encargados de recolectar la información y almacenarla en distintos "*data sets"*.

Al final de este proceso de recolección, se logró obtener un total de 68 archivos de datos almacenados en formato CSV, en cada uno fue especificado: el nombre del dispositivo, fecha de creación, tipo de maniobra e ID establecido por la base de datos central (Ej. "*raspberry_02-Feb-2022-20-07_frenada repentina_Data-Mv2GjLnIwNMl56-k3B4.csv*"). En total, el "***data set"*** formado por los datos en crudo (o *raw)* obtuvo 36353 filas; de 18 columnas para archivos registrados por el *Smartphone* y 19 columnas para archivos registrados por el dispositivo híbrido (cada archivo CSV puede observarse en el **Anexo I. *Data sets* de maniobras de conducción**). La **Tabla 9** muestra el número total de registros obtenidos por cada maniobra de conducción en los diferentes dispositivos.

Tabla 9. Registros de datos para cada maniobra

Maniobra	Dispositivo	Filas	Columnas
Aceleración repentina	*Smartphone*	8079	18

	Híbrido	2312	19
Frenada repentina	*Smartphone*	3893	18
	Híbrido	1878	19
Giro derecho agresivo	*Smartphone*	4310	18
	Híbrido	1174	19
Giro izquierdo agresivo	*Smartphone*	3144	18
	Híbrido	1523	19
Cambio de línea agresivo-derecha	*Smartphone*	3253	18
	Híbrido	1241	19
Cambio de línea agresivo-izquierda	*Smartphone*	3733	18
	Híbrido	1813	19

La herramienta usada para la lectura y visualización de los datos fue la plataforma ICRDS web (el desarrollo de la plataforma es detallado en el **Anexo E. Desarrollo del prototipo SDIRC**), esta permitió conocer por medio de sus graficas **(Figura 20** y **Figura 24**) el comportamiento que presentan los datos. En este caso la teoría planteada en los trabajos de [24], [26], [34], [37] y [51] pudo ser corroborada a través del análisis visual del ICRDS web. Esta teoría expone que no todas las variables y ejes proyectan cambios al realizar las diferentes maniobras de conducción.

La **Figura 24** muestra la relación que tienen las maniobras de conducción con respecto a cada una de las características cinemáticas almacenadas en el "*data set"*; la zona marcada en rojo presenta el momento en que la maniobra fue ejecutada. Tomando como base la teoría anteriormente planteada y el análisis grafico realizado por medio del ICRDS web, fue posible plantear la hipótesis de que cada maniobra de conducción puede ser definida a partir de solo algunas características cinemáticas. De esta manera es posible descartar aquellas características que no presenten un cambio significativo en el momento de su ejecución.

Por medio de lo anterior, resultó posible demostrar la hipótesis planteada. Como ejemplo, se tiene que: la maniobra de aceleración y frenado repentino presentan variaciones significativas en las características de velocidad y aceleración en "Y", mientras que las otras variables se mantienen indiferentes a este tipo de eventos, presentando variaciones insignificantes o únicamente ruido. Para la maniobra de giro agresivo, tanto de derecha como de izquierda, las características con mayor relevancia fueron la aceleración en los ejes "X" y "Y", su velocidad angular en "Z" y la fuerza magnética en los ejes "X" y "Y". Por último, la maniobra de cambio de línea agresivo, tanto de derecha como de izquierda, presenta variaciones significativas en la aceleración del eje "X" y la velocidad angular del eje "Z".

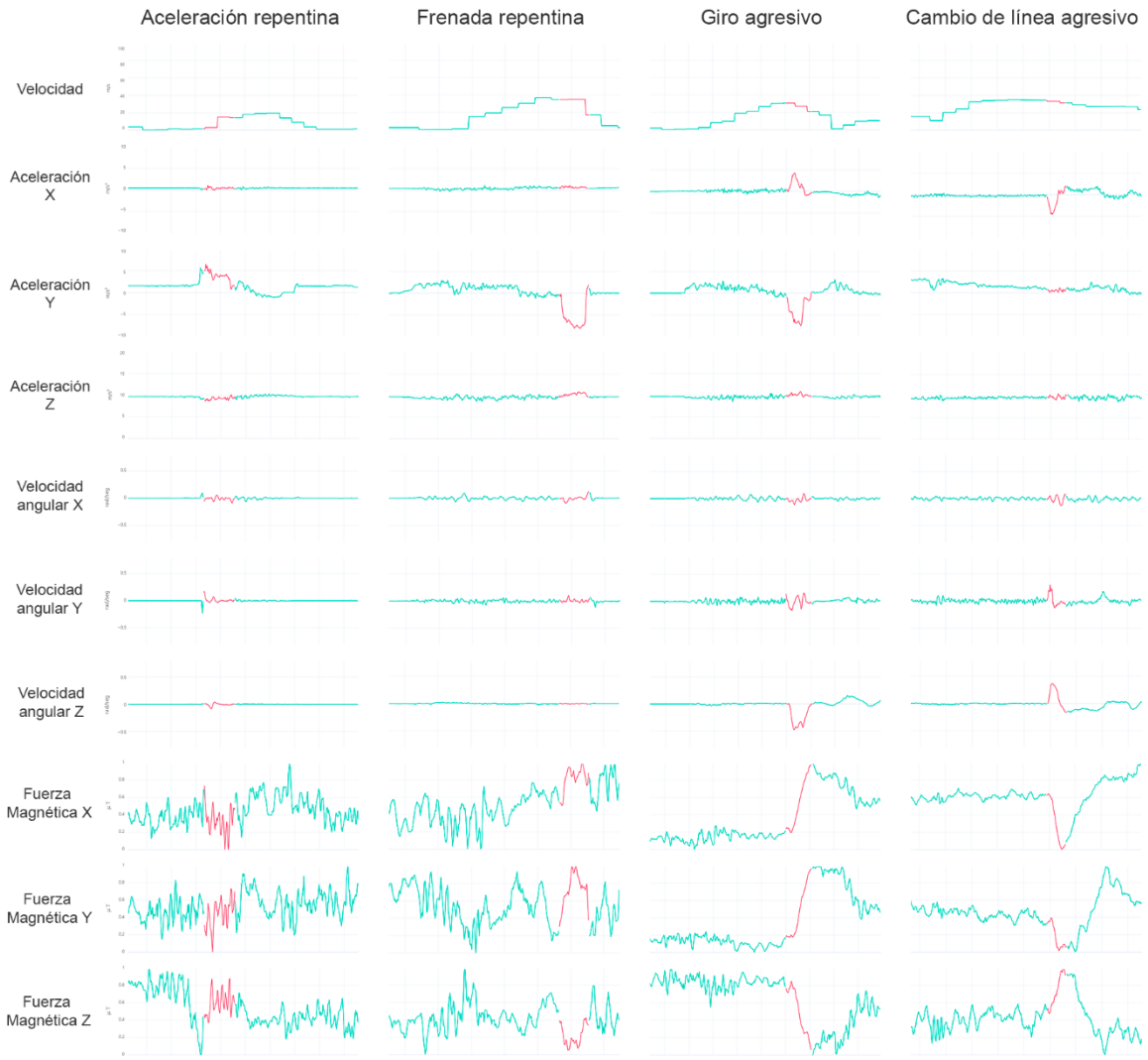

Figura 24. Visualización de la relación entre variables cinemáticas y maniobras de conducción, durante la fase de comprensión de datos de la metodología CRISP-DM

Además, con base a lo anterior también fueron identificadas características que son independientes de cualquier maniobra realizada en el vehículo. Estas son: la aceleración en "Z", que marca constantemente la aceleración de la gravedad; la velocidad angular en el eje "X" y "Y", que definen movimientos tridimensionales como el cabeceo (*pitch*) y rotación (*roll*), los cuales no se ven gravemente afectados al ejecutar las diferentes maniobras; y por último la fuerza magnética en "Z", que presenta el campo magnético inducido en este eje.

Finalmente, la calidad de los datos fue verificada y a partir de esta fue construido un reporte resumido en la **Tabla 10**, en esta tabla son detallados: 1) Los inconvenientes detectados en los "*data sets"*, 2) El comportamiento de los datos a raíz del problema y 3) La posible solución.

Tabla 10. Revisión de la calidad de los datos

Inconveniente	Comportamiento de los datos	Posible solución
Afección de las características de la velocidad y posición del acelerador en el *"data set"* del dispositivo híbrido debido a la desconexión del escáner ELM 327.	El último registro obtenido por el ELM 327 fue añadido hasta que la conexión se restableciera.	Obtener nuevos puntos por medio del método de interpolación.
Etiquetado incorrecto de un evento de conducción por parte del operador. Provocando un desfase o errores en la marcación de un evento de *near-crash*.	El *target* del *"data set"* (columna eventClass) es marcado con el valor de 1 en momentos donde el evento de *near-crash* aun no ocurre.	Identificar los momentos donde el etiquetado fue erróneo y cambiar el registro de la columna "eventClass" por un 0.
Desplazamiento en los datos de aceleración debido a la presencia de inclinación en el lugar donde fueron ubicados los dispositivos (*Smartphone* o híbrido).	Los registros de la aceleración en sus 3 ejes se encontraban desplazados por un pequeño valor numérico.	Eliminar el desplazamiento restando al valor capturado el valor desplazado u *offset*.
Presencia de ruido en los datos obtenidos por la IMU debido a factores como el motor o fricción de las ruedas.	Las características cinemáticas tenían valores con ruido o presentaban variaciones con picos anormales.	Aplicar un filtro en todos los datos cinemáticos. Según la revisión de la literatura el filtro adecuado para datos de cinemática vehicular es el filtro Kalman.

5.4.3. Fase III: Preparación de los datos

En esta fase fueron detalladas las actividades necesarias para generar el *"data set"* final, el cual contiene los datos que sirvieron como entrada del modelo de ML. Las actividades de esta fase consistieron en: seleccionar las características de interés para cada maniobra de conducción, limpiar los datos del "*data set" raw* con base en el reconocimiento de la anterior fase, realizar el proceso de extracción de características a través de la ejecución de una ventana de tiempo deslizante y finalmente, describir el *"data set"* que fue introducido como entrada en el modelo de ML.

5.4.3.1. Selección de características

Como fue especificado en la fase I de la metodología CRISP-DM (ver sección 5.4.1), el desarrollo del modelo de ML tomó como punto de partida la identificación de distintas maniobras de conducción; estas son las que representaron a un evento de *near-crash*. Por lo tanto, para la selección de características fue necesario identificar las variables que tuvieron: mejor correlación en la matriz de dispersión de la plataforma web ICRDS (**Figura 20d**); cambios notables en los datos según la maniobra ejecutada, observados a través del grafico lineal de la plataforma web ICRDS (**Figura 24**); e información relevante con base en el estado actual del conocimiento ([24], [26], [34] y [51]).

Al finalizar la revisión se creó la **Tabla 11,** donde se indican las variables escogidas para definir finalmente las diferentes maniobras de conducción. Hay un total de 8 maniobras de conducción definidas para cada uno de los dispositivos (*Smartphone* e híbrido).

Tabla 11. Selección de características

Maniobra de conducción	**Características Cinemáticas**											
	speed	speed OBD-II	acc Position	acc X	acc Y	accZ	vel Ang X	vel Ang Y	vel Ang Z	mag X	mag Y	mag Z
Aceleración repentina (*Smartphone*)	✓				✓							
Frenado repentino (*Smartphone*)	✓				✓							
Giro agresivo (*Smartphone*)				✓	✓				✓	✓	✓	
Cambio de línea agresivo (*Smartphone*)				✓					✓			
Aceleración repentina (*híbrido*)		✓	✓		✓							
Frenado repentino (*híbrido*)		✓			✓							
Giro agresivo (*híbrido*)		✓		✓	✓				✓	✓		
Cambio de línea agresivo (*híbrido*)				✓					✓			

Con base en lo anterior, las características descartadas totalmente del *"data set" raw* fueron: accZ, debido a que constantemente marca la fuerza de la gravedad y no tiene variaciones significativas en la ejecución de las maniobras; velAngX y velAngY, debido a que las variaciones observadas en los 68 archivos fueron minúsculas en comparación con la velAngZ; y por último, magZ, es descartada porque nunca tuvo como referencia el norte magnético de la tierra, debido al sistema de coordenadas usado (sección 5.3.2) donde el eje "Z" siempre se mantuvo ortogonal al plano del suelo.

5.4.3.2. Limpieza de datos

La actividad de limpieza de datos la conformaron dos partes, una parte manual y otra automatizada por medio de un código en *Python*. En la primera parte fueron ejecutadas las correcciones necesarias para los 3 primeros inconvenientes de la **Tabla 10** y en la segunda parte el ultimo inconveniente relacionado con la aplicación del filtro Kalman. Cada una de estas partes son descritas a continuación:

- **Limpieza de datos primera parte**. Para lograr la limpieza de datos en los 68 *"data set"* fue utilizado un cuaderno de *Jupyter* y la librería *Pandas* de *Python,* el código estructurado en este archivo permitió a un controlador revisar cada característica del *"data set"* y modificar sus datos según el inconveniente identificado (el código puede observarse en el **Anexo H. Plataforma ICRDS web** con el nombre de "data_cleaning.ipynb").

El primer inconveniente identificado, en el cual el escáner ELM 327 sufría, en algunos casos, desconexiones fue solventado por medio de la ejecución de un método de interpolación incluido en la librería *Pandas*. En este método fue necesario definir por medio del ID del registro en el *"data set"* el intervalo donde ocurría el problema (este intervalo fue definido en código como "*min_zone*" y "*max_zone"*), así como la columna del "*data set"* en la cual se presentaba el problema (en el código esta información fue definida como "*var_to_interpolate*").

El segundo inconveniente, en el que los datos fueron etiquetados incorrectamente fue solucionado aplicando un cambio simple en el valor de la columna "eventClass". Para lo anterior fue necesario definir por medio del ID del "*data set"* el intervalo o los intervalos donde ocurría este problema (en el código los intervalos fueron definidos como "min_ID" y "max_ID").

Por último, para solucionar el inconveniente del desplazamiento de los datos fue necesario eliminar el *offset* que presentaban las características capturadas por el acelerómetro. Por lo tanto, fueron identificados en el *"data set"* instantes en los cuales el carro estuvo en un estado de reposo, estos instantes eran definidos como tiempo de espera (*standby*). Por medio del *standby* y la característica con desfase (en el código cada elemento es nombrado como "*standby*" y "*var_with_offset*", respectivamente) fue creada la **ecuación (5) y (6)** la cual permitió eliminar el desfase de los datos.

$$offset = media(standby_{datos}) \quad \textbf{(5)}$$

$$caracteristica = caracteristica\ - offset \quad \textbf{(6)}$$

El valor del *offset* fue obtenido calculando la media de los datos contenidos en el tiempo de espera (**ecuación (5)**). Finalmente, a las características que contenían desfase les fue restado el *offset* previamente calculado y posteriormente fueron sobrescritos los datos sin desfase (**ecuación (6)**).

Finalizar la primera fase de la limpieza permitió generar "*data sets"* libres de los tres primeros errores, cada uno de estos "*data sets"* fueron guardados y almacenados en el **Anexo I. *Data sets* de maniobras de conducción**. Además fue creado un archivo en formato JSON en el cual se reportan todos los cambios realizados en los "*data sets" raw*, este archivo fue nombrado como: "near_crash_correction.json" y es posible verlo en el **Anexo H. Plataforma ICRDS web**.

- **Limpieza de datos segunda parte**. El ultimo inconveniente presentado en la **Tabla 10** fue solucionado por medio de un algoritmo de *Python* ejecutado en un cuaderno de *Jupyter*. Este algoritmo seleccionaba los 68 "*data set"* y aplicaba un filtro Kalman a todas las columnas que contuvieran datos cinemáticos (el código puede observarse en el **Anexo H. Plataforma ICRDS web** con el nombre de "data_filter.ipynb").

El filtro Kalman es un algoritmo no supervisado encargado de identificar el estado real de una secuencia de mediciones ruidosas [26]. Implementar un filtro Kalman en mediciones de variables cinemáticas permite reducir picos provocados por ruidos externos a los sensores, estos picos normalmente conducen a la detección de mayores falsos positivos por lo que es importante mitigarlos.

La librería usada para la ejecución del filtro Kalman fue *pykalman*, esta permitía ejecutar el algoritmo de filtrado sin necesidad de especificar los estados o parámetros del filtro. El algoritmo no supervisado EM (por sus siglas en ingles *Expectation Maximization*) fue el encargado de la optimización de los estados o parámetros del filtro [69]. Usar este algoritmo permitió conseguir los mejores estados o parámetros adecuados para realizar el filtro, pero aumento drásticamente el tiempo de ejecución del modelo de ML, así como el costo computacional.

La **Figura 25** ilustra como el filtro Kalman logró eliminar con éxito el ruido producido por factores externos a los sensores IMU. Los datos filtrados son presentados en color azul y los datos sin filtrar son presentados en color verde.

Figura 25. Eliminación de ruido mediante el uso del Filtro Kalman

Además, en la etapa de filtrado fue realizada la normalización de las características del magnetómetro, ya que el valor de la magnitud de la intensidad del campo geomagnético era variante en cada prueba, pero su comportamiento era similar. La **ecuación (7)** muestra la formula usada para la normalización, esta es comúnmente conocida como normalización por reescalado o "*min-max normalization"*.

$$X_{normalizada} = (X + X_{min})/(X_{max} - X_{min}) \qquad \textbf{(7)}$$

Al finalizar la ejecución del algoritmo fueron generados 68 nuevos archivos que pueden observarse en el **Anexo I. *Data sets* de maniobras de conducción**, estos archivos serían usados para la extracción de características.

5.4.3.3. Extracción de características

La ventana de tiempo deslizante fue la técnica usada para la extracción de características. este tipo de técnica es frecuentemente usada para caracterizar datos de series temporales

(los detalles de esta técnica son descritos en la sección 2.5.3). Las razones por la cual esta técnica fue escogida para la extracción de características son las siguientes:

- La revisión sistemática permitió conocer que muchos de los estudios relacionados con el desarrollo de modelos de ML hacían uso de ventanas de tiempo deslizante en la caracterización de sus datos. Los estudios de [26] y [34] demostraron que la ventana de tiempo deslizante permitía conseguir un mayor rendimiento en sus modelos de ML (los rendimientos de estos estudios pueden ser observados en la **Tabla 4**).

- Los comportamientos de conducción atípicos representan solo una pequeña parte en los "*data set"* relacionados con análisis viales, por lo que usar una ventana de tiempo deslizante permitirá recaudar más información sobre el contexto de los eventos de conducción, lo que a su vez conlleva a una menor probabilidad de que el modelo se sub-ajuste [32].

Como fue mencionado en la sección 2.5.3 la ventana de tiempo deslizante contiene dos características de ajuste: el tamaño de la ventana, y el intervalo deslizante. Para el desarrollo de este trabajo solo fue variado el tamaño de la ventana, tomando valores de 20 o 40 muestras de datos tal y como fue realizado por algunos autores en el estado del arte [34] (considerando que la frecuencia de muestreo en las pruebas desarrolladas en la sección 6.1 fue establecida en 20 Hz, el tamaño de la ventana también puede ser definido como 1 o 2 segundos, respectivamente). Además, el intervalo deslizante de la ventana de tiempo fue establecido constante con un valor igual a 1 para el desarrollo total del presente trabajo.

El proceso de extracción de características fue desarrollado a partir de un algoritmo que tomó como punto de partida los "*data sets"* limpios obtenidos en la anterior sección, al finalizar la extracción fue obtenido un "*data set"* final o el *"data set"* de entrada para el modelo de ML. La **Figura 26** muestra el proceso de extracción de características realizado por el algoritmo.

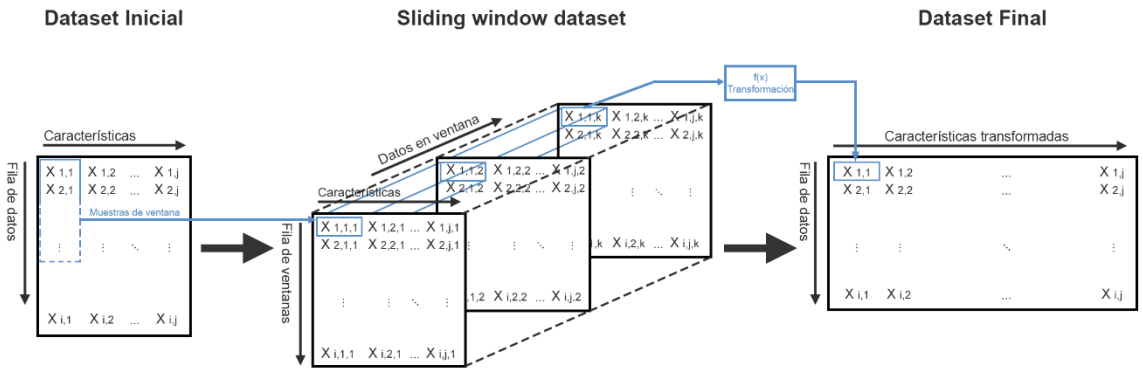

Figura 26. *Data sets* en el proceso de extracción de características

En este proceso se manejaron 3 "*data sets"*: *"data set"* inicial, *"data set"* generado por la ventana de tiempo y el *"data set"* final. Cada *"data set"* puede ser visto como una matriz de

datos que cuentan con tamaños y dimensiones diferentes. En este sentido las matrices pueden definirse de la siguiente manera:

- **Matriz del "*data set*" inicial.** En ella se encuentran los datos capturados por el módulo de recolección de datos preprocesados y limpiados por el administrador de datos archivados (secciones 5.4.3.1 y 5.4.3.2). Esta matriz cuenta con 2 dimensiones representada por filas y columnas, el tamaño de cada dimensión es el siguiente:
    - Filas (eje i): Número total de los registros obtenidos en el proceso de captura de datos, este número varía en función del tiempo que duró la recolección de datos.
    - Columnas (eje j): Número de características cinemáticas, este número varía en función de las características seleccionadas en las diferentes maniobras de conducción.
- **Matriz del "*data set*" de la ventana de tiempo deslizante.** Es la matriz resultante de aplicar la técnica de ventana de tiempo deslizante (ver sección 2.5.3). Esta matriz cuenta con 3 dimensiones representadas por filas, columnas y páginas, cada dimensión es definida como:
    - Filas (eje i): Numero de ventanas de tiempo creadas a partir de los registros del "*data set*" inicial. El tamaño de esta dimensión es calculado a través de la **ecuación (8)**, donde: "m" es el número total de registros obtenidos en el *"data set"* inicial y "*ws*" es el tamaño de la ventana de tiempo (este puede ser 20 o 40). Es importante notar que el tamaño de esta dimensión siempre es menor al tamaño de la dimensión equivalente en la matriz anterior, debido al recorte sufrido por el tamaño de la ventana de tiempo.

$$i = m - ws + 1 \qquad \textbf{(8)}$$

    - Nota: la **ecuación (8)** solo aplica para ventanas de tiempo con intervalo deslizante igual a 1.
    - Columnas (eje j): Al igual que en la anterior matriz esta dimensión contiene las características cinemáticas seleccionadas en las diferentes maniobras de conducción.
    - Paginas (eje k): Son las muestras incluidas dentro de una ventana de tiempo, el tamaño de esta dimensión es igual al tamaño de la ventana.
- **Matriz del "*data set*" final.** Esta matriz surge como resultado de aplicar en la matriz anterior (matriz de ventana de tiempo) las funciones de transformación. Estas funciones constan de cálculos matemáticos que permiten generar una mayor ganancia de información de los eventos atípicos presentados en datos de conducción. Con base a la revisión sistemática (sección 3.1) fue posible definir las

funciones de transformación a usar en el presente trabajo, estas son: la media (**ecuación (9)**), mediana (**ecuación (10)**), desviación estándar (**ecuación (11)**), máximo valor (**ecuación (12)**), mínimo valor (**ecuación (13)**) y tendencia (**ecuación (14)**).

$$\bar{X} = \frac{\left(\sum_{i=1}^{k} x_i\right)}{k} \quad \textbf{(9)}$$

$$mediana = Md(x_1, x_k) \quad \textbf{(10)}$$

$$std = \sqrt{\frac{\sum_{i=1}^{k}(x_i - \bar{X})^2}{k}} \quad \textbf{(11)}$$

$$maximo = \max(x_1, x_k) \quad \textbf{(12)}$$

$$minimo = \min(x_1, x_k) \quad \textbf{(13)}$$

$$T_i = \frac{\bar{X}_i}{\bar{X}_{i-1}} \quad \textbf{(14)}$$

Cada una de las anteriores funciones de transformación fueron aplicadas al eje k de la matriz de ventana de tiempo deslizante, exceptuando la ecuación de la tendencia la cual utilizo el eje i de la matriz. En el archivo "machine_learning.ipynb" del **Anexo H. Plataforma ICRDS web** se encuentra el código que permitió los cálculos de cada una de las función de transformación.

La matriz resultante al finalizar el proceso de extracción de características vuelve a ser una matriz bidimensional, debido a que cada función de transformación convirtió a todos los datos del eje k en un solo valor. Las dimensiones de esta matriz son representadas por medio de filas y columnas y su tamaño es el siguiente:

- Filas (eje i): Numero de ventanas de tiempo creadas a partir de los registros del "*data set"* inicial. Este eje es heredado de la matriz de la ventana de tiempo, por lo tanto, su tamaño también es calculado por medio de la **ecuación (8)**.
- Columnas (eje j): Es el número de funciones de transformación multiplicado por las características seleccionadas en las diferentes maniobras de conducción.

5.4.3.4. *Data set* de la fase de preparación

Completar cada uno de los anteriores procesos permitió generar 12 "*data sets"* que sirven como entrada para cada clasificador de ML, estos "*data sets"* definieron las diferentes maniobras de conducción (**Tabla 9**). Las dimensiones de cada *"data set"* las conformaron las filas y columnas del "*data set"* final presente en la sección 5.4.3.3. Además, fueron agregadas 3 columnas extras: ID de la primera muestra, ID de la última muestra y la etiqueta

(*target*). Los detalles de las columnas de los "*data sets"* finales son explicados en la **Tabla 12**.

Tabla 12. *Data set* de la fase de preparación

Nombre de columna	**Descripción**	**Tipo**
ID inicial de la muestra (*first_id*)	El identificador único del primer registro de la ventana de tiempo.	Numérico
ID final de la muestra (*last_id*)	El identificador único del último registro de la ventana de tiempo.	Numérico
Media (*mean*) (*)	La media calculada a partir de todos los registros contenidos por la ventana de tiempo.	Numérico
Mediana (*median*) (*)	La mediana calculada a partir de todos los registros contenidos por la ventana de tiempo.	Numérico
Desviación estándar (*std*) (*)	La desviación estándar calculada a partir de todos los registros contenidos por la ventana de tiempo.	Numérico
Máximo valor (*max_val*) (*)	El máximo valor en los registros de la ventana de tiempo.	Numérico
Mínimo valor (*min_val*) (*)	El mínimo valor en los registros de la ventana de tiempo.	Numérico
Tendencia (*tendency*)	La división entre la media de los registros actuales en la ventana de tiempo y la anterior media de los registros.	Numérico
Clase (*target*)	La moda de la clase calculada a partir de todos los registros contenidos por la ventana de tiempo.	Booleano
(*) Esta columna es multiplicada dependiendo las características de la maniobra de conducción realizada (**Tabla 11**). Ej. si el evento es un frenado repentino tendremos 2 columnas representando a la media estas son: media de la velocidad y media de la aceleración en "Y".		

5.4.4. Fase IV: Modelado

En esta fase fueron realizadas las actividades relacionadas con la construcción de los algoritmos clasificadores seleccionados en la fase I (sección 5.4.1), la optimización de los parámetros de cada clasificador y la creación de las pruebas de rendimiento de los clasificadores.

La sección 4.3 presenta el método usado para seleccionar los algoritmos en el modelo de ML, estos son: SVM, RF, DT. Para crear cada uno de los algoritmos fue necesario hacer uso de la herramienta para análisis predictivos de datos *ScikitLearn*, Esta herramienta se encuentra desarrollada en *Python* y está construida sobre las librerías *Numpy*, *Scipy* y *Matplotlib*. Además, hacer uso de esta herramienta permite realizar la evaluación, optimización y despliegue de cada uno de los clasificadores [40].

Los clasificadores en ScikitLearn son definidos como una clase de un algoritmo de ML especifico. Una clase es el componente central en programación orientada a objetos, esta cuenta diferentes métodos y variables que definen la función del objeto [70]. A continuación, son especificadas las clases que representan a los 3 algoritmos:

- **sklearn.svm.SVC:** Clase para construir el clasificador SMV, su definición es detallada en la sección 2.4.2.1. Los hiper-parámetros de esta clase usados en el desarrollo del modelo de ML fueron: el kernel, el parámetro de regularización (“C”) y el grado de curvatura (*gamma*).

- **sklearn.tree.DecisionTreeClassifier:** Clase para la construcción del clasificador DT, su definición es detallada en la sección 2.4.2.2. Los hiper-parámetros de esta clase usados en el desarrollo del modelo de ML fueron: criterio de selección, máxima profundidad del árbol, máximo número de nodos de bajo nivel peso del *target* (ayuda a balancear los datos tomados en cada nodo del árbol).

- **sklearn.ensemble.RandomForestClassifier:** Clase para la construcción del clasificador RF, su definición es detallada en la sección 2.4.2.3. Los hiper-parámetros usados en el desarrollo del modelo de ML fueron: máximo porcentaje de muestras, número de árboles estimadores y peso del *target* (ayuda a balancear los datos dentro de cada árbol).

Los anteriores clasificadores cuentan con el método “*fit()”* que permite entrenar el clasificador a partir de datos de entrenamiento. Los datos de entrenamiento usados en el desarrollo del presente trabajo fueron los 12 “*data sets”* finales presentados en la sección 5.4.3.4. Los datos de entrenamiento fueron obtenidos a partir de la división aleatoria de las muestras de un “*data set”*, esta división permitió obtener un subconjunto de datos de entrenamiento y prueba, este último es usado para la evaluación del algoritmo. Al finalizar el entrenamiento el clasificador era capaz de realizar predicciones a través del método “*predict()”.*

A partir de las predicciones del clasificador fue posible probar el rendimiento preliminar de cada uno de los clasificadores. En estas pruebas de rendimiento fueron usadas las métricas de evaluación especificadas en la sección 2.4.3 (*precision, recall, F1-score, AUC*) y la matriz de confusión. Esta última métrica de evaluación permite visualizar el rendimiento del algoritmo a partir de una tabla en la que se indican las cuatro medidas fundamentales (Vp, Fp, Vn, Fn) [29].

Basado en los resultados obtenidos por la evaluación preliminar, en esta evaluación los hiper-parámetros no fueron ajustados, los rendimientos de los clasificadores no fueron los óptimos. Debido a lo anterior fue necesario aplicar un método de optimización en los clasificadores. Esta optimización fue realizada a partir del método “*GridSearchCV()*” y de la variación del tamaño de la ventana (sección 5.4.3.3).

El método “*GridSearchCV”* es usado para realizar una búsqueda exhaustiva de los mejores hiper-parámetros definidos en una cuadricula por el usuario [71]. Para aplicar este método fue necesario:

- Especificar el clasificador a optimizar, en el presente trabajo los 3 clasificadores fueron optimizados para cada una de las maniobras y dispositivos.

- Definir los hiper-parámetros de cada clasificador y los valores definidos procurando que el rango de valores seleccionado para cada hiper-parámetro evitara el sobreajuste. A continuación, son indicados los valores que tomaron cada hiper-parámetros de los clasificadores:

    - **Clasificador SVM**. Regularización: 0.01, 0.05, 0.1, 1, 10, 100, 1000; *gamma*: 0.001, 0.005, 0.01, 0.05, 0.1, 1, 10. La regularización y *gamma* fueron definidas a escala logarítmica empezando por 10E-2 y 10E-3 respectivamente, la escala logarítmica permite ver un mayor espectro de datos y evita sobrecostos computacionales. El hiper-parámetro de regularización "C" tiende al sobre-ajustarse con valores pequeños y ser más tolerante a errores a valores grandes. Para el hiper-parámetro *gamma* el sobre-ajuste aparece en valores grandes [31].

    - **Clasificador DT.** Criterio de selección: "Gini"; máxima profundidad: 2, 3, 4, 5, 6, 7, 8, 9, 10; máximo número de nodos de bajo nivel: 5, 6, 10, 15, 20, 25, 30, 40, 50, 60; peso del *target*: balanceado o no balanceado. Los rangos de ajuste son variantes en este clasificador ya que normalmente dependen del tipo de datos, en el caso de la evaluación preliminar el árbol de decisión tenía una profundidad variante entre 5 a 12 y nodos de bajo nivel u hojas entre 20 y 50. Como fue mencionado en la sección 2.4.2.2 el DT tiende a sobre-ajustarse cuando no es aplicada una poda en el árbol, por lo tanto los valores pequeños de los hiper-parámetros permitieron que el árbol no llegara a sobre-ajustarse.

    - **Clasificador RF.** Máximo porcentaje de muestras: 90%, 95%, 100%; número de árboles: 100, 200, 300; peso del *target*: balanceado o no balanceado. La lógica del clasificador RF es evitar el sobre-ajuste visto en el DT. Esto se logra agregando un número grande de árboles y una pequeña variación en las muestras. Aunque aumentar el número de árboles resulta beneficioso también puede hacer que los costos computacionales se eleven demasiado, por eso fue prudente establecer valores que no fueran sumamente grandes pero que lograran marcar diferencias.

- Definir un esquema de validación cruzada, este método utiliza una división de datos para evaluar el rendimiento del conjunto de datos total. En este sentido los datos son divididos en "K pliegues", cada pliegue representa el subconjunto de datos, donde cada K-1 pliegues corresponden a los datos de entrenamiento y solo uno de los pliegues representa los datos de prueba (datos con los cuales es realizada la evaluación de rendimiento) [42]. En el presente trabajo fue tomado el valor de K=5 y la división de datos fue realizada de manera aleatoria.

- Por último, fueron definidas las métricas a usar en la optimización, estas son: *precision*, *recall, f1-score* y *AUC*.

Una vez finalizada la optimización de los clasificadores fueron obtenidos los valores de los hiper-parámetros que lograban un mejor rendimiento basado en la métrica de evaluación optimizada. Esto permitió generar una tabla con 248 registros de puntajes de evaluación (ver **Anexo J. Rendimiento de los algoritmos**) donde son especificados los valores de los hiper-parámetros y su rendimiento en la validación cruzada. En la siguiente sección es evaluada la tabla de puntajes con el fin de seleccionar el algoritmo que mejor representa los eventos de *near-crashes*.

5.4.5. Fase V: Evaluación

En esta fase fueron evaluados los clasificadores generados en la fase anterior con el fin de seleccionar aquel clasificador que tenga el mejor rendimiento. Además, fue comprobado por medio de algunas pruebas de campo si el modelo cumple con el objetivo de la fase I. Finalmente, fueron determinadas las posibles acciones a realizar en futuras iteraciones.

La tabla presentada en el **Anexo J. Rendimiento de los algoritmos** fue usada para la evaluación de cada uno de los clasificadores junto con sus hiper-parámetros. En esta tabla fueron conformados conjuntos de evaluación de la siguiente forma: 1) clasificador de ML, 2) dispositivo, 3) maniobra de conducción, 4) tamaño de la ventana deslizante y 5) Métrica optimizada. Los resultados obtenidos fueron presentados en la sección 7.2, estos resultados permitieron definir los clasificadores que mejor rendimiento tuvieron al momento de clasificar las distintas maniobras de conducción; así como la ventana de tiempo y los hiper-parámetros con los que se obtuvo un mejor rendimiento.

De esta manera, el algoritmo RF y una ventana de tiempo de 40 muestras fueron escogidos para el desarrollo de cada uno de los clasificadores en el modelo de ML. Cada uno de los clasificadores conto con diferentes hiper-parámetros especificados en la **Tabla 13**.

Tabla 13. Hiper-parámetros para cada clasificador

Maniobras (clasificadores: RF, tamaño ventana: 40	**Dispositivo**	**Hiper-parámetros**		
		Porcentaje de muestras (%)	**Número de árboles estimadores**	**Peso del *target***
Aceleración repentina	*Smartphone*	95	200	No balanceado
	Híbrido	100	300	Balanceado
Frenada repentina	*Smartphone*	100	300	No balanceado
	Híbrido	95	200	Balanceado
Giro derecho agresivo	*Smartphone*	95	100	Balanceado
	Híbrido	95	300	Balanceado
Giro izquierdo agresivo	*Smartphone*	100	300	No balanceado
	Híbrido	90	300	No balanceado
Cambio de línea agresivo-derecha	*Smartphone*	95	300	Balanceado
	Híbrido	100	200	No balanceado

Cambio de línea agresivo-izquierda	*Smartphone*	95	100	No balanceado
	Híbrido	95	100	No balanceado

Una vez desarrollados los 12 clasificadores de ML fue posible comprobar a partir de una prueba preliminar (en la que nuevos datos fueron recolectados) que el reconocimiento de estas maniobras era realizado con éxito. Por lo tanto, el objetivo de la fase I de CRIPS-DM (sección 5.4.1) fue completado, dicho de otra manera, los 12 clasificadores de ML estaban listos para la creación del modelo de ML y su respectivo despliegue.

Finalmente, en esta fase se determinó las futuras acciones a realizar en futuras versiones del modelo de ML. Estas fueron: realizar un cambio en el filtro Kalman, ya que el tiempo de pre-procesamiento en los "*data sets"* fue demasiado largo; y actualizar el modelo a partir de nuevos datos adquiridos en las pruebas de ND (sección 6.2), esto permitirá generar un modelo de ML con mejor rendimiento que el actualmente disponible.

5.4.6. Fase VI: Despliegue

En la última fase de CRISP-DM fue desarrollada la forma en la que la plataforma ICRDS web usaría el modelo de ML. Para esto fue necesario hacer uso de la herramienta *joblib* que permitió crear clasificadores de ML persistentes en el disco [72].

Con *joblib* los 12 clasificadores de ML fueron generados en archivos con extensión "*.joblib"* (estos archivos se encuentran en la carpeta "*./machine_learning/built_classifiers*" del **Anexo H. Plataforma ICRDS web**) para que posteriormente la plataforma ICRDS web los cargara y ejecutara un algoritmo que permitiera la detección de *near-crashes*. La **Figura 27** muestra el procedimiento realizado por el ICRDS web con el fin de lograr la detección y subida de los datos de *near-crash*. La detección se realiza de esta manera: 1) los datos son descargados de la base de datos central, 2) se realiza el proceso de preparación de datos (sección 5.4.3), 3) los 12 clasificadores (dependiendo del dispositivo) clasifican la ocurrencia de una maniobra, 4) cada detección de una maniobra es almacenada como un *near-crash* 5) los eventos de *near-crash* repetidos son unidos, 6) los eventos son enviados a la base de datos central.

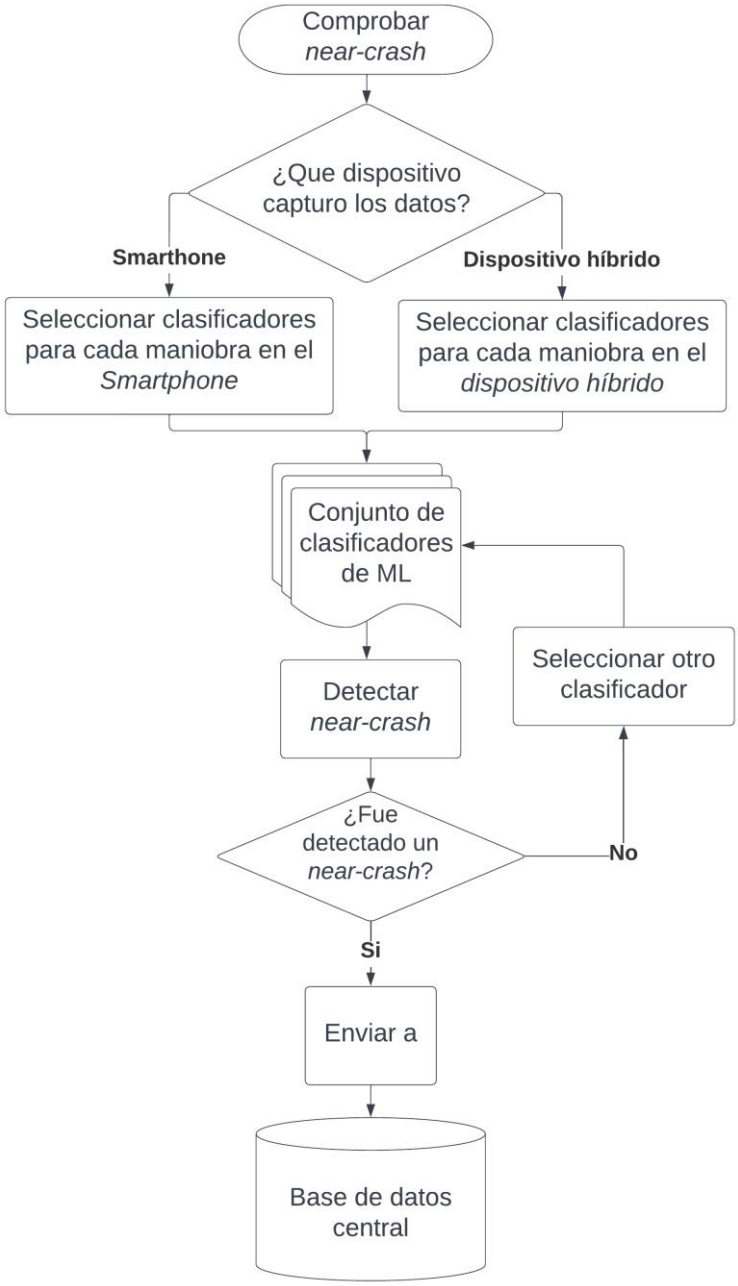

Figura 27. Diagrama de flujo predicción *near-crash*

Finalizar cada una de las fases de CRISP-DM dio paso a ejecutar las pruebas de campo descritas en la sección 6.2.

Capítulo 6

6. Diseño y desarrollo de las pruebas

Las pruebas de campo a desarrollar fueron diseñadas tomando como base el proceso de desarrollo del modelo de ML y el propósito de identificar zonas con alta probabilidad de accidentes de tránsito. Lo anterior condujo a diseñar y realizar dos tipos de pruebas de campo: controladas y no controladas.

Las pruebas de campo controladas estuvieron enfocadas en la fase de entrenamiento y creación del modelo de ML. Por otro lado, las no controladas estuvieron orientadas a validar tanto la hipótesis del presente trabajo como el funcionamiento del modelo de ML previamente entrenado. Esto último a través del uso de la metodología de ND (sección 2.1) y la detección de *near-crashes* usando el modelo de ML desarrollado.

A continuación, son descritas con mayor detalle las pruebas de campo realizadas:

6.1. Pruebas de campo de entrenamiento (controladas)

El objetivo por lograr con el modelo de ML correspondía con la solución de un problema de clasificación mediante aprendizaje supervisado. Como fue mencionado en la sección 5.4.1, el modelo de ML estuvo dirigido a la detección de *near-crashes* a través de maniobras de conducción asociadas a estos eventos (aceleración repentina; frenada repentina; giro derecho e izquierdo agresivo; cambio de línea derecho e izquierdo agresivo).

Con base en lo anterior, fueron diseñadas las pruebas controladas, las cuales estuvieron dirigidas a la recolección de datos de conducción que incluyan las maniobras de interés y eventos de conducción normal (sección 5.4.2). El resultado deseado con estas pruebas corresponde a un *"data set"* de entrenamiento con datos etiquetados, elemento requerido en todos los problemas de aprendizaje supervisado. Este *"data set"* sería un soporte principal para el desarrollo del modelo de ML.

Para la ejecución de estas pruebas fue utilizado únicamente un vehículo *Nissan March modelo 2016* y un único conductor realizó cada maniobra reiteradamente. Las maniobras fueron realizadas en vías planas donde el tránsito de vehículos fue casi nulo. Esto ayudo a evitar posibles accidentes durante la ejecución de las diferentes maniobras debido a su alto riesgo. Los dispositivos *Smartphone* e hibrido fueron ubicados en la zona intermedia del piso de los pasajeros, la **Figura 28** muestra un ejemplo de la ubicación.

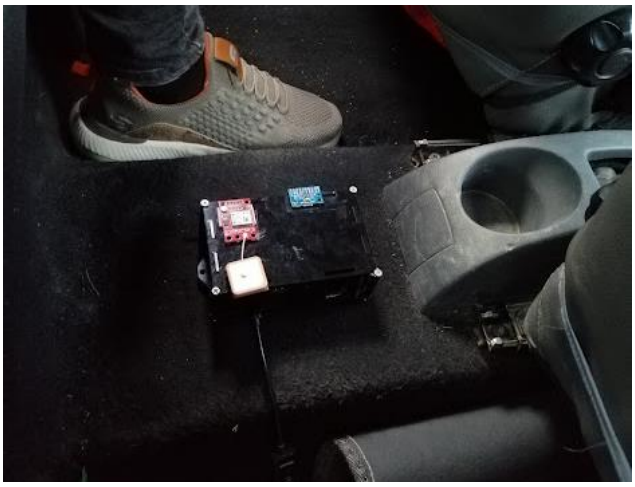

Figura 28. Ubicación de los dispositivos en el vehículo

La frecuencia seleccionada para la ejecución de las pruebas fue 20 Hz, la razón de esta selección estuvo fundamentada en el tiempo que tomo realizar cada una de las maniobras descritas en la **Tabla 9**. Estas maniobras tuvieron una duración aproximada de 0.7 segundos a 2.5 segundos, por lo que obtener 20 muestras por segundo serían suficiente para cumplir con la condición de frecuencia de muestreo de Nyquist. Además, en pruebas preliminares de los dispositivos fue comprobado que establecer una alta frecuencia en la captura de datos producía errores de inestabilidad tanto en la aplicación como de los sensores IMU, estos últimos soportaron frecuencias de 50 Hz antes de presentar el error.

La etiqueta en los datos recolectados representa la ocurrencia o no de una maniobra de interés y por ende de un *near-crash*. Para lograr este etiquetado durante el proceso de recolección de datos, ambos dispositivos de recolección fueron desarrollados con una funcionalidad que facilitara esta tarea. En el caso del *Smartphone* esto fue posible mediante un botón ubicado en la interfaz principal de la aplicación (**Figura 13**); en el caso del dispositivo híbrido, mediante un pulsador eléctrico conectado a un pin GPIO (**Figura 16**).

Durante el proceso de recolección de datos, los botones descritos fueron presionados por los operadores del módulo RD cuando el conductor inició la ejecución de una maniobra. De esta forma, el valor del *target* o etiqueta de cada registro cambiaba al valor correspondiente a un *near-crash*. Los nuevos registros seguían siendo guardados con dicha etiqueta hasta que los botones de cada dispositivo fueran nuevamente presionados, indicando el fin de la maniobra de conducción asociada al *near-crash* y volviendo etiquetar los nuevos registros como eventos de conducción normal.

A continuación, se describe el procedimiento ejecutado para realizar cada una de las maniobras:

- **Aceleración repentina.** Con el vehículo en estado de reposo o en movimiento, el conductor realizó aceleraciones drásticas que incrementaron la velocidad del vehículo.

- **Frenada repentina.** El conductor presiono el pedal de freno bruscamente para obligar al vehículo a parar o disminuir drásticamente su velocidad, en algunas ocasiones el vehículo presento leves derrapes.

- **Giro derecho e izquierdo agresivo.** Con el vehículo en movimiento, el conductor realizó un giro brusco (derecho e izquierdo), el giro consistió en realizar un movimiento brusco en el volante haciendo virar rápidamente al vehículo y lo detuviera.

- **Cambio de línea a la derecha e izquierda agresivo.** Con el vehículo en movimiento, el conductor cambió repentinamente de carril por medio de un movimiento leve en el volante y sin detener al vehículo, también se realizaron pruebas en las cuales el conductor cambiaba de carril y nuevamente volvía al carril inicial.

La **Tabla 14** relaciona cada maniobra ejecutada con su respectiva cantidad de repeticiones. A partir de los datos generados en la ejecución de estas maniobras fue construido el modelo de ML de la sección 5.4.

Tabla 14. Cantidad de maniobras ejecutadas

Maniobra	Repeticiones
Aceleración repentina	12 (+ 8 repeticiones donde se presentaron aceleraciones leves)
Frenada repentina	9
Giro derecho agresivo	8
Giro izquierdo agresivo	7
Cambio de línea agresivo-derecha	9
Cambio de línea agresivo-izquierda	11

6.2. Pruebas de campo de validación (no controladas)

Las pruebas de campo que se ejecutaron constituyeron un componente determinante en la evaluación del uso de los métodos de ND y de los *near-crashes*, como herramientas para identificar zonas con alta probabilidad de AT. Con lo anterior y teniendo en cuenta que el prototipo desarrollado busca ayudar en la identificación estas zonas, usando los *near-crash*es como medida sustituta de los choques, es importante resaltar que su funcionamiento y los resultados obtenidos a partir él, están basados en la frecuencia de ocurrencia de *near-crashes* en determinadas zonas.

Lograr el anterior objetivo mediante el prototipo desarrollado implica la construcción de un "*data set*" de variables cinemáticas vehiculares. Dichos datos deben ser recolectados realizando recorridos sobre diferentes rutas clave de la ciudad en cuestión, usando un determinado número de conductores, vehículos y definiendo otros factores relevantes para dichas pruebas o recorridos.

El método seleccionado para realizar los recorridos a través de la ciudad (pruebas de campo) sigue el enfoque propuesto por los estudios de conducción naturalista (ND). Los estudios de ND se caracterizan por proporcionar una visión sobre el comportamiento del conductor y sus maniobras a lo largo de diferentes viajes realizados sin ningún control experimental, manteniendo su estilo habitual de conducción. Lo anterior es logrado gracias a la recolección de datos asociados al conductor, el vehículo y su entorno [19].

Antes de diseñar las pruebas de campo, fueron examinados los trabajos del estado del arte que realizaron recolección de datos empleando la metodología de conducción naturalista. Posteriormente, unas pruebas de campo convenientes para el presente trabajo fueron diseñadas, considerando, además de la anterior revisión, algunos datos relacionados con la movilidad y la seguridad vial de Popayán.

Dentro de los documentos que conforman el estado del arte del presente trabajo, cinco de ellos realizaron procesos de recolección de datos mediante conducción naturalista. Algunos aspectos identificados en ellos son mencionados de forma general a continuación:

- El número de conductores usados en las pruebas varió considerablemente, tres estudios ([11], [37], [39]) emplearon cantidades mayores a 30 conductores y dos estudios ([24] y [26]), cantidades menores o iguales a 5.

- El número de vehículos utilizados en las pruebas osciló en cantidades bajas, donde la mayor cantidad de vehículos usados fue de 5 en ([26]). Además, usar vehículos repetitivamente con diferentes conductores y en diferentes rutas fue una idea respaldada por los experimentos de algunos trabajos ([11], [26] y [37]).

- Los tipos de vías o carreteras donde fueron realizadas las pruebas cambió de un trabajo a otro según la infraestructura vial de los lugares de realización de dichos estudios. Como menciona [11], cada tipo de carretera influye de forma distinta en el comportamiento del conductor, por presentar diferentes condiciones de tráfico, límites de velocidad etc.

- La duración y distancia recorrida totales de las pruebas examinadas mantuvieron de forma general valores altos. [26] presenta la menor cantidad con un aproximado de 20 horas y 1200 km de conducción naturalista.

- Solo un documento ([39]) dio libertad para que los conductores escogieran las rutas e igualmente un documento ([24]) empleó cinco *Smartphones* diferentes.

La revisión de las pruebas de ND en los documentos del estado del arte es descrita con mayor detalle en el **Anexo K. Revisión y diseño de pruebas de campo de ND**. Es posible mencionar que ninguno de los trabajos revisados brindó detalles sobre los criterios o razones consideradas al momento de seleccionar las diferentes variables, como número de conductores o vehículos, rutas, tipos de vías o distancias a recorrer, entre otras.

La revisión realizada permitió conocer que una ventaja de realizar estudios de conducción naturalista extensos radica en un aumento en la probabilidad de registrar un mayor número

de eventos de interés [11], [37], [39]. Aunque lo anterior podía ser favorable para estudiar diferentes factores de seguridad vial y movilidad de la ciudad, excedía el alcance y los límites de tiempo del presente trabajo. Esto llevó a realizar una inclusión más selectiva de las vías de la ciudad a través de las cuales los datos fueron recolectados. Así mismo, el número de vehículos y conductores participes de estas pruebas debió ser reducido.

Teniendo en cuenta estos limitantes, para maximizar la probabilidad de registrar *near-crashes* durante las pruebas de campo, fue necesario realizar un reconocimiento de los puntos críticos y de las vías de la ciudad que podrían presentar mayores conflictos de movilidad; descartando inicialmente aquellas vías y zonas de la ciudad donde la ocurrencia de los siniestros de movilidad fuera poco probable.

Para cumplir con la anterior tarea, los diferentes análisis y caracterizaciones de la movilidad y la seguridad vial comprendidos en el plan de movilidad de Popayán [73] fueron tomados como base. Como resultado de lo anterior, fueron identificados 15 puntos críticos [74], los cuales son listados a continuación y pueden ser apreciados en la **Figura 29**:

1. Calle 5 con variante (salida al Tambo)
2. Calle 5 con Carrera 37
3. Calle 5 con Carrera 25
4. Calle 5 con Carrera 14
5. Calle 5 con Carrera 9
6. Carrera 17 con Calle 8
7. Transversal 9N sector Terminal de transportes
8. Calle 13 con Carrera 9
9. Carrera 9 con Calle 4
10. Carrera 9 con Calle 24N
11. Intersección de Campanario
12. Carrera 6 con Calle 25N
13. Carrera 9 sector Universidad Antonio Nariño (cerca a la piedra norte)
14. Carrera 9 con Calle 63N, sector Bella vista
15. Transversal 9A N con Variante.

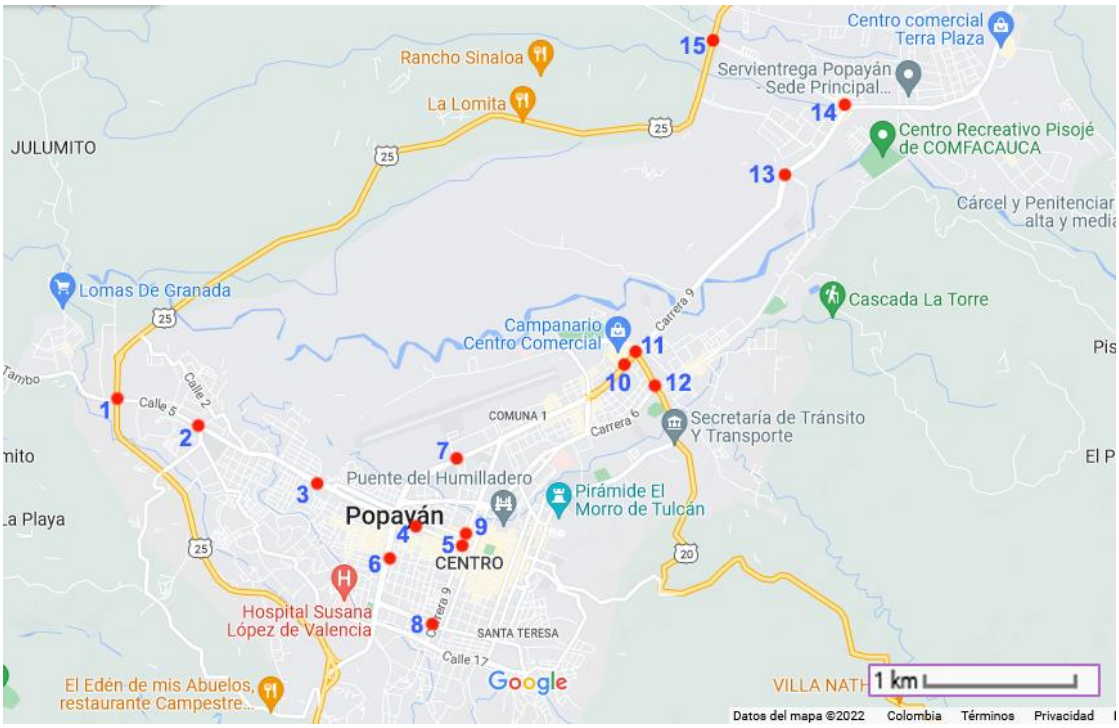

Figura 29. Puntos críticos de movilidad en Popayán (mapa tomado de *Google Maps*)

Buscando contemplar las vías y puntos de movilidad más críticos fueron definidas 5 rutas que corresponden a los trayectos donde fueron realizadas las pruebas de campo de validación. Los datos de interés tomados del Plan de Movilidad de Popayán ([73] y [74]) y la descripción de las 5 rutas son mencionados en detalle en el **Anexo K. Revisión y diseño de pruebas de campo de ND**.

La ejecución de estas pruebas fue realizada mediante el recorrido de las rutas definidas anteriormente. Para esto, cada ruta fue recorrida por dos conductores, cada uno en su respectivo vehículo (*Renault Logan modelo 2007* y *KIA Picanto Ion modelo 2014*). Ambos vehículos empleados en estas pruebas cumplieron los dos requisitos necesarios: pertenecer a la categoría de automóviles y ser compatibles con el escáner OBD-II: ELM 327.

La decisión de realizar los recorridos de las rutas únicamente con dos conductores y dos vehículos se debió a las siguientes limitantes: el tiempo para finalización del proyecto, el presupuesto corto (para gasto de combustible y remuneración para los conductores), la dificultad de encontrar voluntarios y la dificultad de encontrar vehículos compatibles con los *scanners* OBD-II ELM 327 utilizados.

Para la ejecución de cada recorrido, momentos antes de iniciar cada uno de ellos, se solicitó a los conductores mantener su estilo habitual de conducción. Y durante los recorridos, los operadores de los dispositivos de recolección de datos acompañaron al conductor evitando al máximo influir en su conducción.

La recolección de datos fue realizada simultáneamente por cada uno de los dispositivos considerados: un *Smartphone* Xiaomi Redmi Note 9S con la aplicación móvil instalada y el

dispositivo híbrido desarrollado. Estos dispositivos fueron ubicados en la zona intermedia del piso de los pasajeros (**Figura 28**). Considerando los recursos disponibles (conductores, vehículos y cantidad de dispositivos) en el desarrollo de las pruebas no fue posible realizar recorridos de forma simultánea, lo cual solamente afectó en que el tiempo para recoger los datos fuera más amplio.

Con las anteriores condiciones, la recolección de datos fue realizada para un total de 10 trayectos, y para cada uno de estos trayectos fueron creados dos "*data sets"* correspondientes a cada uno de los dos dispositivos de recolección de datos. Esto permitió comparar su funcionamiento y desempeño en condiciones similares. La **Tabla 15** describe de forma resumida la ejecución de las pruebas de campo de validación.

Tabla 15. Lista de recorridos de pruebas de campo de validación

No. De recorrido	Vehículo	Ruta	Fecha	Hora inicio	Duración aproximada	No. de registros aprox.
1	Kia Picanto Ion 2014	1	25/Mar/2022	4:16 PM	55 min.	65900
2		2	30/Mar/ 2022	4:18 PM	53 min.	64040
3		3	31/Mar/ 2022	11:23 AM	43 min.	51456
4		4	01/Abr/ 2022	11:39 AM	48 min.	58206
5		5	02/Abr/2022	4:08 PM	51 min.	60691
6	Renault Logan 2007	1	07/Abr/2022	9:41 AM	72 min.	85884
7		2	07/Abr/2022	4:17 PM	66 min.	79730
8		3	06/Abr/2022	3:49 PM	67 min.	80560
9		4	06/Abr/2022	9:33 AM	48 min.	57824
10		5	06/Abr/2022	10:27 AM	36 min.	43909

El número de registros aproximado mostrado en la **Tabla 15**, corresponde con un promedio del número de registros almacenados por el dispositivo híbrido y por el *Smartphone*. Esto debido a que sincronizar de forma exacta tanto el inicio como finalización de los procesos de recolección de datos en los dos dispositivos fue complicado. Sin embargo, la diferencia en el número de registros de ambos dispositivos no fue significativa.

Todos los recorridos fueron realizados durante momentos con condiciones climáticas soleadas, nubladas y parcialmente nubladas. Ningún recorrido fue realizado durante lluvias o lloviznas. Es importante afirmar que no hubo intensión alguna en seleccionar o evitar uno de estos tipos de condiciones climáticas, puesto que no fue una variable para considerar en el presente proyecto.

Por último, las condiciones del tráfico variaron en cada recorrido según su hora y día de ejecución. Tanto el día como la hora de un recorrido dependieron del tiempo disponible por parte de los conductores. Teniendo en cuenta las fechas listadas en la **Tabla 15** es posible afirmar que los recorridos fueron realizados de miércoles a sábado.

Capítulo 7

7. Resultados

Este capítulo presenta y evalúa los resultados obtenidos del proceso de desarrollo del prototipo para el sistema SDIRC, y los obtenidos de emplear el prototipo durante las pruebas de campo de validación (sección 6.2).

7.1. Prototipo para el sistema SDIRC

En esta sección tanto el módulo RD como el módulo AII fueron verificados a través de una comprobación del cumplimiento de los requisitos del sistema. Al emplear la metodología SCRUM, estos requisitos fueron descritos como épicas e historias de usuario.

Para comprobar el desarrollo del sistema fue necesario realizar pruebas unitarias que permitieran verificar las funcionalidades de los requisitos del sistema. Al igual que corroborar que todas las vistas sean presentadas correctamente al cliente. Los fallos presentados en el código fueron corregidos y cada requisito de alta prioridad fue realizado. En este sentido las historias de usuario fueron definidas como implementadas, no implementadas o parcialmente implementadas, dependiendo del resultado de las pruebas.

La **Tabla 16** lista todos los títulos de las historias de usuario identificadas junto con su criterio de completitud y algunos comentarios (para conocer la descripción completa de las historias de usuario ver **Anexo D. Desarrollo del prototipo del sistema mediante SCRUM**).

Tabla 16. Verificación cumplimiento historia de usuarios

No.	Historia	¿Imple-mentado?	Comentarios
1	Activación/Desactivación de recolección de datos del dispositivo hibrido.	Sí	A través de un botón presente en la interfaz web (**Figura 14a**) el operador puede controlar el proceso de recolección de datos en el dispositivo híbrido.
2	Verificación de datos recolectados del dispositivo híbrido.	Sí	La interfaz web del dispositivo hibrido desarrollado permite ver una lista de los viajes que han sido realizados y están almacenados en la base de datos local (**Figura 14b**). Además, puede ver los datos que están siendo obtenidos durante del proceso de recolección (**Figura 14a**).
3	Ajuste de parámetros del dispositivo híbrido	Sí	La interfaz web del dispositivo hibrido desarrollado permite ingresar una frecuencia a la cual son recolectadas las variables, seleccionar el vehículo y la ruta en los donde es realizado un viaje (**Figura 14a**).

4	Activación/Desactivación de recolección de datos en el *Smartphone*	Sí	La interfaz principal de la aplicación móvil desarrollada permite al operador controlar el proceso de recolección de datos con un *Smartphone* mediante un botón (**Figura 13a**).
5	Visualización en tiempo real de los datos capturados en el *Smartphone*	Sí	La interfaz principal de la aplicación móvil fue programada para actualizar su estado durante un proceso de recolección de datos y así mostrar los datos que están siendo obtenidos (**Figura 13a**).
6	Ajuste de velocidad de muestro de datos en el *Smartphone*	Sí	A través de un campo de entrada de la interfaz principal el operador puede ingresar la frecuencia a la cual son recolectados los datos (**Figura 13a**).
7	Verificación de datos recolectados en el *Smartphone*	Sí	La interfaz de historial de viajes desarrollada en la aplicación móvil permite a un operador observar los viajes realizados y guardados el base de datos local (**Figura 13b**).
8	Consulta de datos recolectados en trayectos	Sí	Las interfaces del dispositivo hibrido y la aplicación móvil permiten al operador obtener una información breve y descriptiva de los datos de cualquier viaje almacenado en sus bases de datos locales. Además, permiten eliminar los datos de un viaje completo.
9	Extracción de datos recolectados en trayectos	Sí	Las interfaces del dispositivo hibrido y del *Smartphone* tienen implementado un botón que permite enviar los datos de un viaje a una base de datos central (*Firebase Realtime Database*).
10	Consulta información recolectada (en la base de datos central)	Sí	El subsistema de gestión y análisis de los viajes del módulo AII presenta una lista de los viajes presentes en la base de datos central, permitiendo al operador filtrarlos según un rango de fechas, su vehículo y ruta; eliminarlos o ver información detallada sobre el viaje. Además, este subsistema permite al operador graficar los datos de las diferentes variables consideradas (**Figura 20**).
11	Visualizar lugar de recolección	No	Esta historia de usuario fue descartada debido a su bajo valor y prioridad asignada. Su desarrollo implicaba graficar todo el recorrido de una ruta a partir de los datos recolectados. Debido al gran número de datos recolectados, el mapa resultante podría quedar muy saturado y la información brindada a un usuario no sería muy diferente de la obtenida al ver una imagen de las rutas diseñadas (las imágenes de las rutas pueden ser observadas en el **Anexo K. Revisión y diseño de las pruebas de campo de ND**).
12	Consultar información de *near-crashes* (en la base de datos central)	Sí	Consultar la información relacionada con la ocurrencia de un *near-crash* implica realizar una clasificación previa de los datos recolectados durante un trayecto mediante el modelo de ML. Por lo tanto, el subsistema de gestión y análisis de los datos de los datos de viajes desarrollado permite al administrador de datos archivados la consulta de esta información siempre y cuando los datos de un viaje hayan sido analizados (**Figura 21**).

			Además, al usar el filtro por fechas, rutas o vehículos desarrollado en la lista de viajes de la base de datos central, implícitamente estaban siendo filtrados los datos de ocurrencia de *near-crashes*, ya que estos datos siempre pertenecen a un determinado viaje.
13	Visualizar lugares de ocurrencia de near-crashes	Sí	Debido a que esta historia de usuario guardó mucha relación con la historia de usuario número 17, fueron combinadas y desarrolladas en el subsistema de visualización de resultados.
14	Clasificar eventos como near-crash o no near-crash	Sí	El modelo de ML desarrollado e integrado en el módulo de AII permite al administrador de datos archivados realizar el análisis de los datos recolectados durante un viaje con el fin de detectar eventos de *near-crash.* Esta funcionalidad puede ser ejecutada desde la interfaz de información de un viaje (**Figura 19**).
15	Interactuar con los hiper-parámetros del modelo	Parcial	Aunque en las interfaces del módulo AII no existe un apartado que permita al administrador interactuar con los hiper-parámetros del modelo de ML para refinarlo, esta tarea es posible hacerla a nivel de código modificando el archivo *machine_learning.ipynb* para generar un nuevo modelo.
16	Evaluar el modelo	Parcial	Al igual que la historia de usuario 15, aunque no fue desarrollada un interfaz que permita observar la evaluación y rendimiento del modelo a partir de nuevos datos, es posible realizar esta tarea a nivel de código usando el archivo *machine_learning.ipynb.*
17	Observar mapa de puntos	Sí	El subsistema de visualización de resultados del módulo AII (sección 5.3.3.2) permite observar a través de un mapa de puntos las ubicaciones donde un *near-crash* fue detectado por el modelo de ML. Además, permite filtrar estos eventos detectados por dispositivo y ruta.
18	Observar mapa de calor	Sí	El subsistema de visualización de resultados del módulo AII (sección 5.3.3.2) permite observar a través de un mapa de calor las zonas donde los *near-crashes* detectados por el modelo de ML ocurrieron con mayor frecuencia. Igualmente, permite filtrar esta información por dispositivo y ruta.
19	Interactuar con el mapa	Sí	Los mapas desarrollados permiten realizar desplazamientos, acercamientos y alejamientos con el fin de visualizar de mejor forma los lugares y zonas de ocurrencia de un *near-crash*. En el caso del mapa de puntos es posible observar el momento exacto (fecha y hora) de ocurrencia de un *near-crash* apuntado.

7.2. Algoritmos de ML

En esta sección son presentados los resultados obtenidos al evaluar el rendimiento de cada uno de los algoritmos de ML seleccionados. El rendimiento fue obtenido por medio de la técnica de validación cruzada y la optimización realizada en la fase IV de CRISP-DM (sección 5.4.4). Las métricas de evaluación usadas para cuantificar el rendimiento de los algoritmos fueron: *precision, recall, f1-score* y *AUC;* la definición y el cálculo de cada métrica son explicados en la sección 2.4.3 del presente documento.

Los resultados del rendimiento de cada algoritmo sirvieron como base para definir el algoritmo usado en el modelo de ML del sistema SDIRC, estos resultados presentan una comparación entre los algoritmos, ventana de tiempo deslizante y métrica optimizada por el método "*GridSearchCV*" (la funcionalidad de este método es explicada en la sección 5.4.4 y cada uno de los resultados obtenidos a partir de la optimización son mostrados en el **Anexo J. Rendimiento de los algoritmos**).

7.2.1. Rendimiento en el dispositivo *Smartphone*

En esta subsección es presentado el rendimiento que los algoritmos lograron al clasificar eventos de *near-crash*, usando como entrada los datos recolectados por el dispositivo *Smartphone.* Los resultados del rendimiento son presentados como un diagrama de caja, en el cual se puede observar la distribución de los resultados al realizar la técnica de validación cruzada. Además, son indicados por medio de color rojo los resultados obtenidos por una ventana de tiempo de 40 muestras y en color azul los resultados de una ventana de tiempo de 20 muestras.

La **Tabla 17** y la **Figura 30** muestran el rendimiento de los algoritmos al ejecutar la optimización de la métrica de "*precision"*.

Tabla 17. Rendimiento de la métrica "*precision*" para el dispositivo *Smartphone*

Clasificador	Muestras en ventana de tiempo	Máximo valor	Valor medio	Mínimo valor
SVM	20	1	0.9548	0.8857
	40	1	0.9764	0.9326
DT	20	0.9661	0.9367	0.696
	40	0.9688	0.9473	0.8893
RF	20	0.9899	0.9739	0.9521
	40	0.9919	0.9732	0.9593

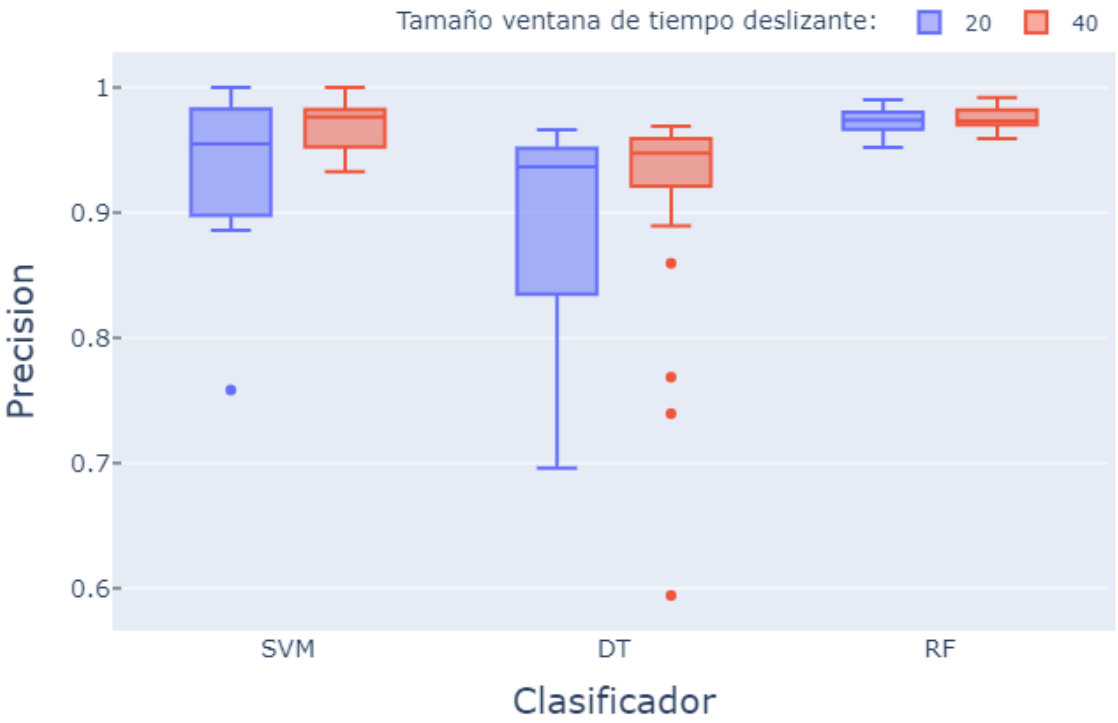

Figura 30. Gráfico de cajas, rendimiento de la métrica "*precision*" para el dispositivo *Smartphone*

La **Tabla 18** y la **Figura 31** muestran el rendimiento de los algoritmos al ejecutar la optimización de la métrica de "*recall"*.

Tabla 18. Rendimiento de la métrica "*recall*" para el dispositivo *Smartphone*

Clasificador	Muestras en ventana de tiempo	Máximo valor	Valor medio	Mínimo valor
SVM	20	0.9536	0.8615	0.077
	40	0.9662	0.9131	0.1909
DT	20	0.9774	0.9490	0.8417
	40	0.9851	0.9504	0.9092
RF	20	0.9728	0.9503	0.9208
	40	0.9778	0.9676	0.9515

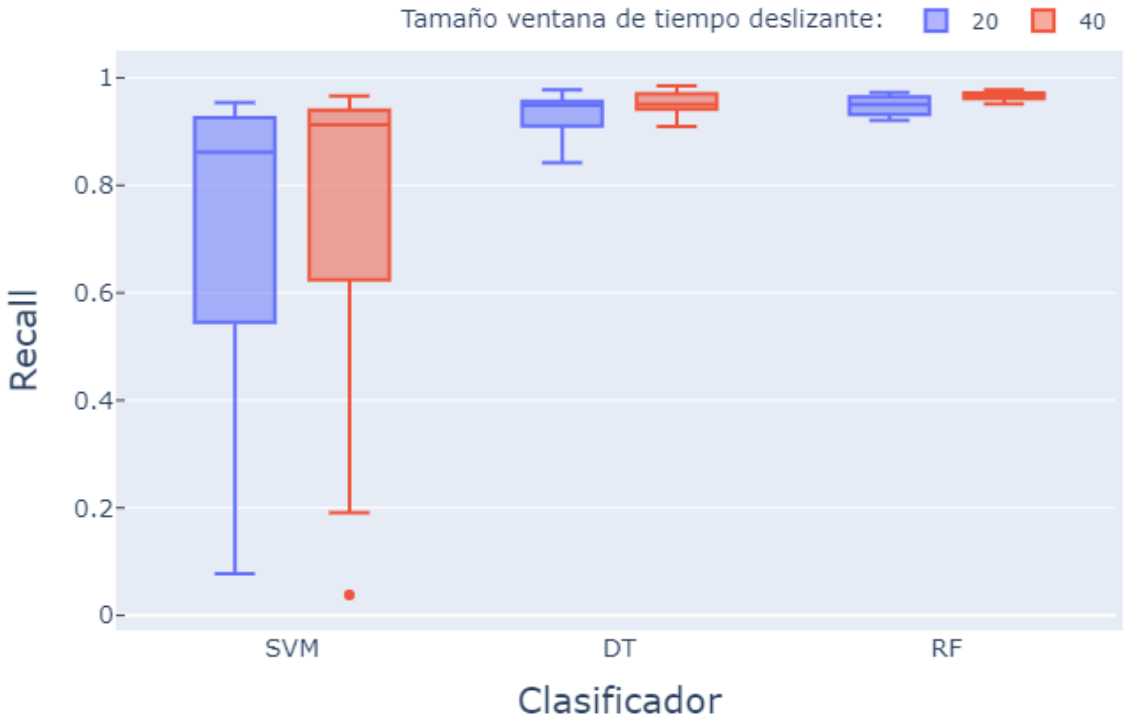

Figura 31. Gráfico de cajas, rendimiento de la métrica "*recall"* para el dispositivo *Smartphone*

La **Tabla 19** y la **Figura 32** muestran el rendimiento de los algoritmos al ejecutar la optimización de la métrica de "*F1-score"*.

Tabla 19. Rendimiento de la métrica "*F1-score*" para el dispositivo *Smartphone*

Clasificador	Muestras en ventana de tiempo	Máximo valor	Valor medio	Mínimo valor
SVM	20	0.9569	0.8775	0.4397
	40	0.9727	0.9305	0.7269
DT	20	0.9608	0.9253	0.8030
	40	0.9734	0.9375	0.9168
RF	20	0.9826	0.9589	0.9461
	40	0.9835	0.9686	0.9584

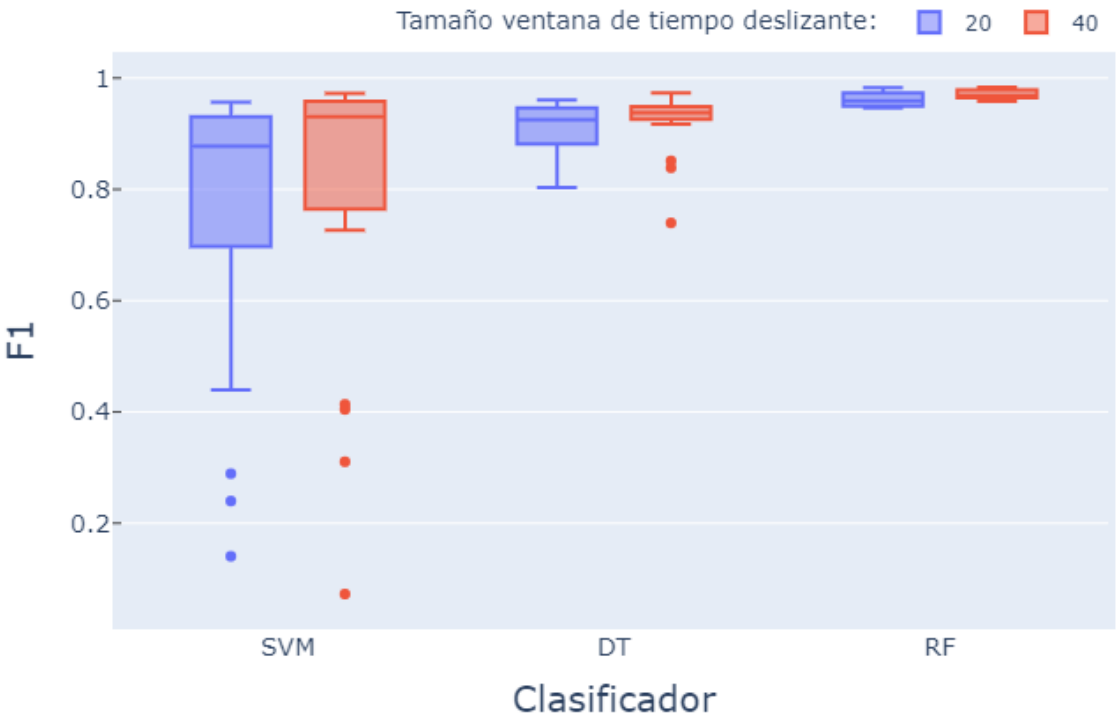

Figura 32. Gráfico de cajas, rendimiento de la métrica "*F1-score*" para el dispositivo *Smartphone*

La **Tabla 20** y la **Figura 33** muestran el rendimiento de los algoritmos al ejecutar la optimización de la métrica de "*AUC"*.

Tabla 20. Rendimiento de la métrica "AUC" para el dispositivo *Smartphone*

Clasificador	Muestras en ventana de tiempo	Máximo valor	Valor medio	Mínimo valor
SVM	20	0.9987	0.9902	0.9841
	40	0.9997	0.9950	0.9618
DT	20	0.9802	0.9713	0.9499
	40	0.9878	0.9718	0.9596
RF	20	0.9998	0.9993	0.9973
	40	0.9999	0.9996	0.9987

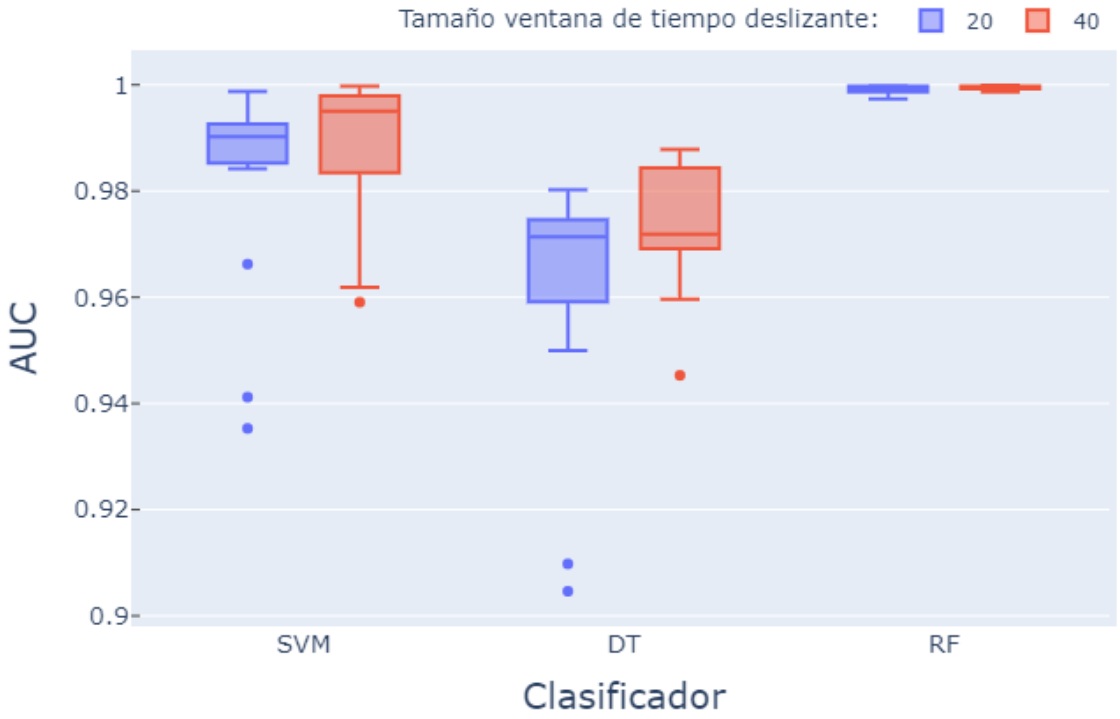

Figura 33. Gráfico de cajas, rendimiento de la métrica "AUC" para el dispositivo *Smartphone*

7.2.2. Rendimiento en el dispositivo híbrido

En esta subsección son presentados los rendimientos que los algoritmos lograron al clasificar eventos de *near-crashes*, usando como entrada los datos del dispositivo híbrido (*Raspberry Pi*). Al igual que en la anterior subsección, los resultados del rendimiento son presentados como un diagrama de caja, en este se puede observar la distribución de los resultados al realizar la técnica de validación cruzada. Además, son indicados por medio de color rojo los resultados obtenidos por una ventana de tiempo de 40 muestras y en color azul los resultados de una ventana de tiempo de 20 muestras.

La **Tabla 21** y la **Figura 34** muestran el rendimiento de los algoritmos al ejecutar la optimización de la métrica de *precision*.

Tabla 21. Rendimiento de la métrica "*precision*" para el dispositivo híbrido

Clasificador	Muestras en ventana de tiempo	Máximo valor	Valor medio	Mínimo valor
SVM	20	1	0.9791	0.9168
	40	1	0.9761	0.9424
DT	20	0.9802	0.9526	0.8536
	40	0.9833	0.9702	0.9215
RF	20	0.9906	0.9757	0.9497
	40	0.9920	0.9814	0.9542

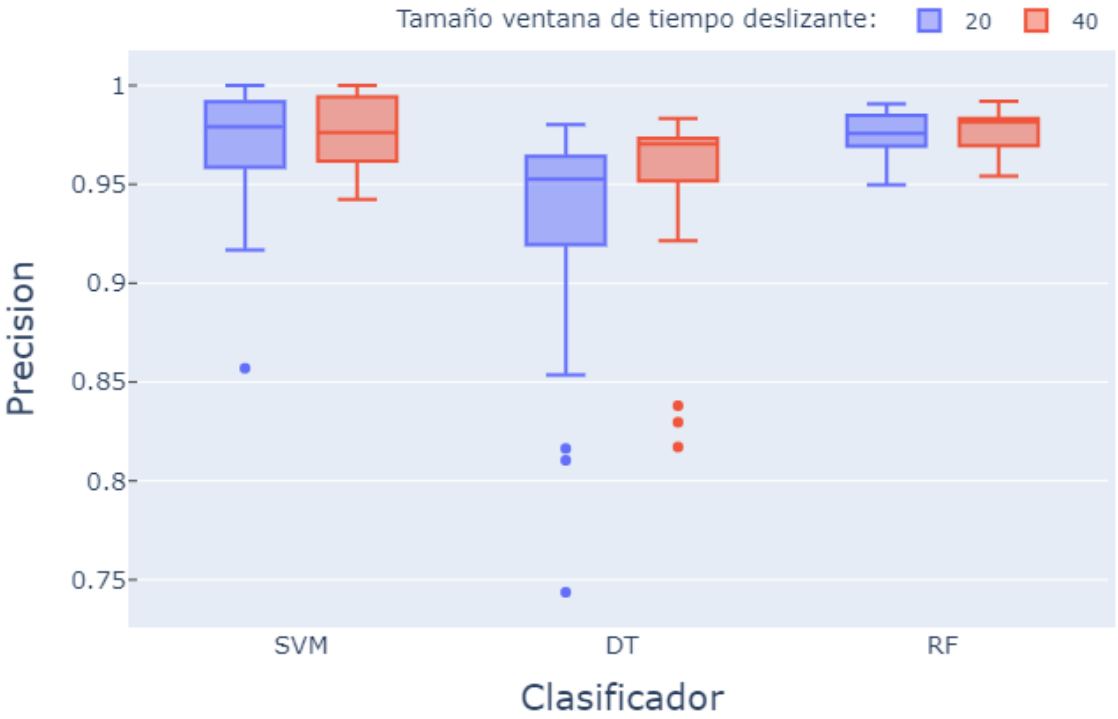

Figura 34. Gráfico de cajas, rendimiento de la métrica "*precision*" para el dispositivo híbrido

La **Tabla 22** y la **Figura 35** muestran el rendimiento de los algoritmos al ejecutar la optimización de la métrica de "*recall"*.

Tabla 22. Rendimiento de la métrica "*recall*" para el dispositivo híbrido

Clasificador	Muestras en ventana de tiempo	Máximo valor	Valor medio	Mínimo valor
SVM	20	0.9948	0.9617	0.5919
	40	0.9941	0.9563	0.5769
DT	20	0.9947	0.9707	0.9469
	40	0.9935	0.9635	0.9441
RF	20	0.9823	0.9714	0.9609
	40	0.9853	0.9746	0.9534

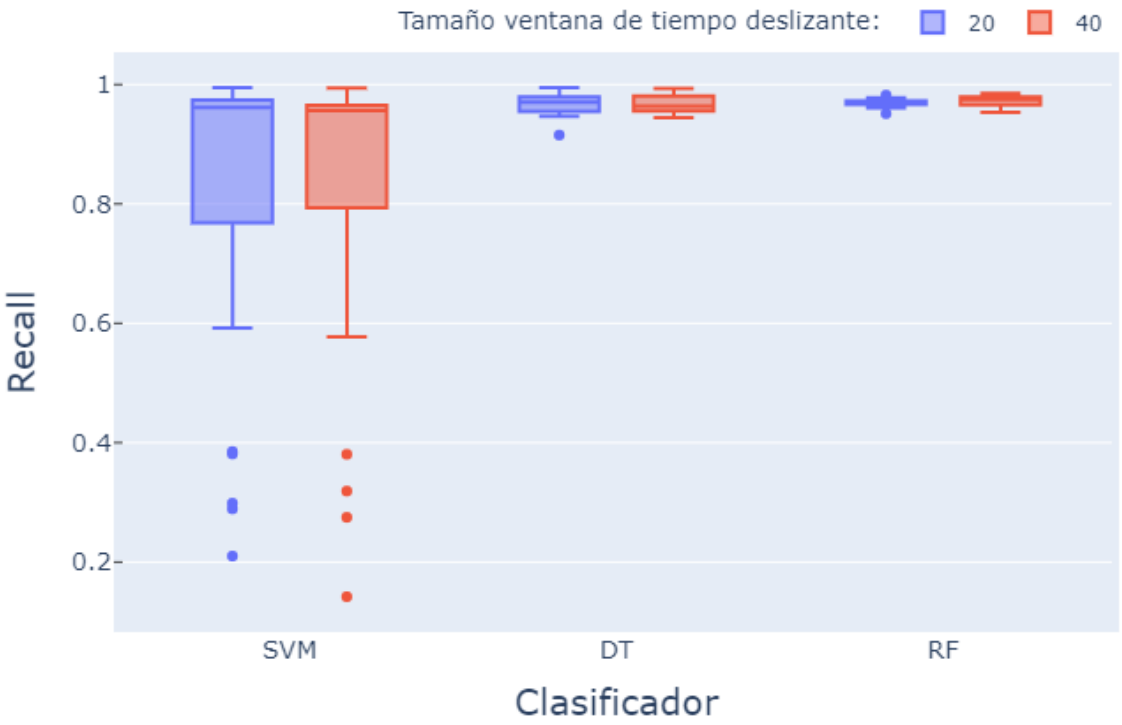

Figura 35. Gráfico de cajas, rendimiento de la métrica "*recall*" para el dispositivo híbrido

La **Tabla 23** y la **Figura 36** muestran el rendimiento de los algoritmos al ejecutar la optimización de la métrica de "*F1-score"*.

Tabla 23. Rendimiento de la métrica "*F1-score*" para el dispositivo híbrido

Clasificador	Muestras en ventana de tiempo	Máximo valor	Valor medio	Mínimo valor
SVM	20	0.9833	0.9596	0.7435
	40	0.9903	0.9585	0.7843
DT	20	0.9714	0.9565	0.9349
	40	0.9834	0.9590	0.9416
RF	20	0.9811	0.9707	0.9614
	40	0.9822	0.9759	0.9556

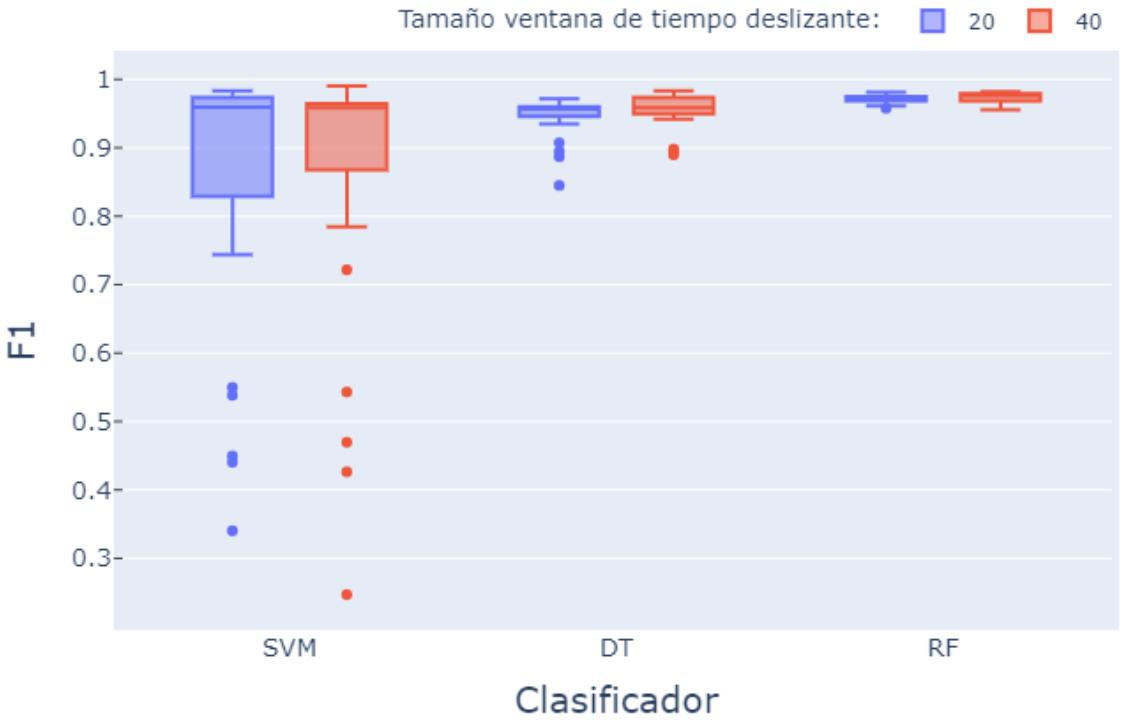

Figura 36. Gráfico de cajas, rendimiento de la métrica "*F1-score*" para el dispositivo híbrido

La **Tabla 24** y la **Figura 37** muestran el rendimiento de los algoritmos al ejecutar la optimización de la métrica de "*AUC"*.

Tabla 24. Rendimiento de la métrica "AUC" para el dispositivo híbrido

Clasificador	Muestras en ventana de tiempo	Máximo valor	Valor medio	Mínimo valor
SVM	20	0.9999	0.9985	0.9924
	40	0.9998	0.9981	0.9895
DT	20	0.9922	0.9831	0.9760
	40	0.9955	0.9804	0.9678
RF	20	0.9999	0.9996	0.9992
	40	0.9999	0.9997	0.9993

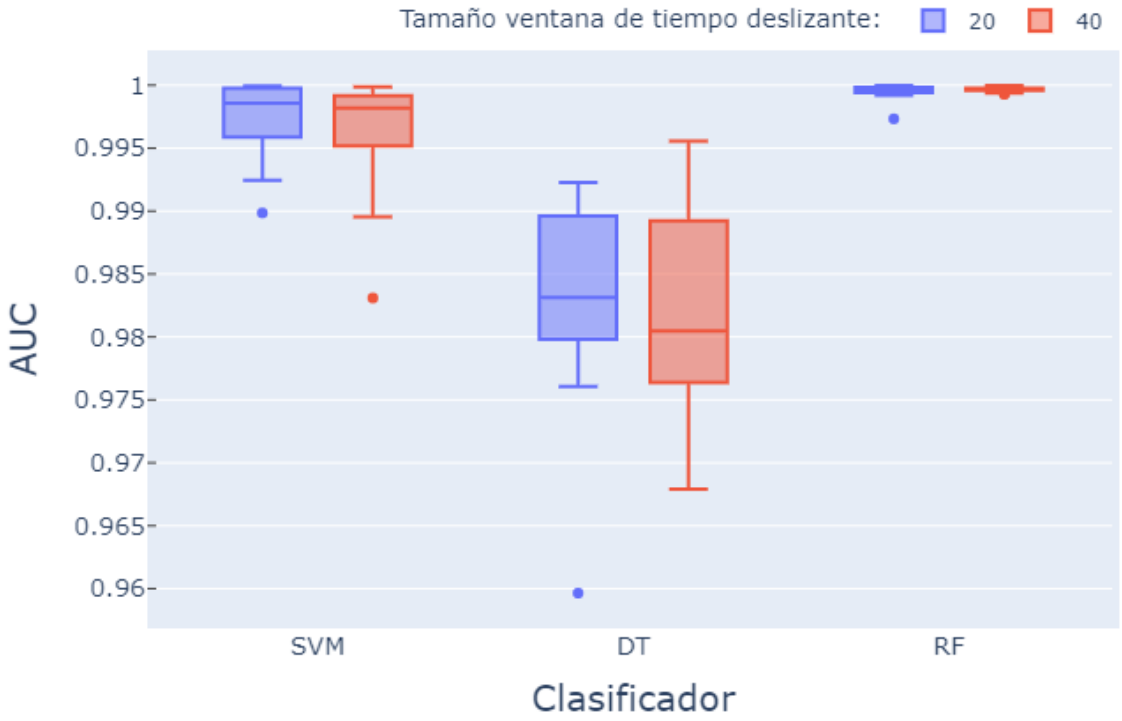

Figura 37. Gráfico de cajas, rendimiento de la métrica "AUC" para el dispositivo híbrido

7.2.3. Análisis de resultados

Para el dispositivo *Smartphone,* los resultados presentados en la sección 7.2.1 permitieron determinar las siguientes afirmaciones:

- El algoritmo SVM presentó una "*precision"* perfecta en el valor máximo del diagrama de caja, pero el algoritmo RF presentó una mejor media en esta métrica.
- El algoritmo SVM presentó muchas más variaciones en la distribución de los valores del rendimiento de la "*precision",* a diferencia del algoritmo RF donde los valores se mantuvieron cercanos.
- El algoritmo RF obtuvo el mejor rendimiento en los resultados de la métrica de "*recall"* haciendo uso de una ventana de tiempo de 40 muestras.
- El algoritmo SVM obtuvo el peor resultado de entre todas las métricas, este fue presentado en el valor mínimo del diagrama de caja para la métrica de *recall*, su valor fue aproximadamente 0.0770.
- Los resultados presentados en las métricas "*recall"* y "*F1-score"* fueron similares para los tres algoritmos, pero en el caso de *F1-score* los algoritmos SVM y DT presentaron varios valores atípicos (un valor atípico es aquel que supera la varianza esperada, o los valores máximos o mínimos del diagrama de caja).
- El rendimiento presentado por los tres algoritmos en los resultados de la métrica AUC obtuvo valores estables y elevados no bajaron de 0.9.

Para el dispositivo híbrido los resultados presentados en la sección 7.2.2 permitieron determinar las siguientes afirmaciones:

- El algoritmo SVM presentó una "*precision"* perfecta en el valor máximo del diagrama de caja, pero el algoritmo RF presentó una mejor media para el caso de una ventana de tiempo con 40 muestras.
- Los algoritmos DT y RF obtuvieron un buen rendimiento en las métricas "*recall"* y "*F1-score"*, mientras que el algoritmo SVM obtuvo un bajo rendimiento.
- El algoritmo SVM presentó una gran cantidad de valores atípicos en los resultados de las métricas "*recall"* y "*F1-score"*.
- El rendimiento del AUC fue nuevamente el mejor para cada uno de los algoritmos, presentando valores por encima de 0.96 y con pocos valores atípicos.

La alta dispersión que sufrieron algunos algoritmos en sus métricas de evaluación permitió determinar que el ajuste de los hiper-parámetros afecta considerablemente el rendimiento del algoritmo. Igualmente, los valores atípicos presentados en los resultados de los algoritmos SVM y DT indicaron que el ajuste de los hiper-parametros debía ser realizado cuidadosamente para evitar sobre-ajustes.

Otra observación realizada en el análisis de resultados fue la aparición de sesgo en los datos (la media de los datos no estaba centrada en el diagrama). La aparición de este sesgo se debe a las muestras utilizadas para el análisis, ya que fueron escogidos los resultados de los rendimientos obtenidos para cada una de las maniobras de conducción.

El "*recall"* fue una de las métricas más críticas para el algoritmo SVM, que presentó varios valores atípicos y un rendimiento bajo comparado con los demás algoritmos. Además, este algoritmo presentó distribuciones no uniformes que generaron diagramas de caja demasiado grandes.

El algoritmo DT mantuvo un rendimiento medio con respecto a los otros algoritmos, en algunos casos se equiparo al rendimiento presentado por el RF, pero no logró superarlo. Además, presentó muchos más valores atípicos que el algoritmo RF.

Por lo anterior, el algoritmo RF fue escogido para el desarrollo del modelo de ML presentado en la sección 5.4. Debido al rendimiento presentado (el promedio de todas las métricas optimizadas dio un valor igual a 0.9801), poca presencia de valores atípicos y distribución uniforme en sus resultados.

El tamaño de ventana deslízate de 40 muestras obtuvo mejor rendimiento que la ventana de 20 muestras, esto permitió demostrar los resultados presentados por [34]. Por esta razón la ventana de tiempo de 40 muestras fue seleccionada como parámetro para el desarrollo del modelo de ML presentado en la sección 5.4.

7.3. Resultados de las pruebas de campo de validación

Inicialmente, las pruebas de campo de validación (sección 6.2) permitieron construir 10 "*data sets"* de ND para cada uno de los dispositivos usados en la recolección de datos (*Smartphone* y dispositivo híbrido). La estructura de los "*data sets*" es igual a la mostrada en la sección 5.2. Cada "*data set*" corresponde al recorrido de una ruta en un determinado vehículo. En total, los 10 "*data sets*" de un dispositivo representan 539 minutos o aproximadamente 9 horas de ND; y una distancia de 175.2 Km, con base en las distancias estimadas para cada ruta (ver **Anexo K. Revisión y diseño de las pruebas de campo de ND**). Los "*data sets*" recolectados pueden ser consultados en el **Anexo L. Data sets de ND**.

Mediante el uso del modelo de ML desarrollado fue posible analizar cada "*data set*" buscando identificar *near-crashes* ocurridos durante los viajes realizados. Una vez este proceso finalizó correctamente, fue posible observar gráficamente los resultados a través de los mapas de calor y mapas de puntos del subsistema de visualización de resultados (sección 5.3.3.2). La **Figura 38, Figura 39, Figura 40, Figura 41** muestran los mapas con las ubicaciones de los eventos detectados.

Un total de 349 *near-crashes* fueron detectados en los datos recolectados por ambos dispositivos. Del total de eventos detectados, 81 de ellos fueron detectados en los datos recolectados por el *Smartphone*. Esta cantidad es superada significativamente por los 268 *near-crashes* detectados a partir de los "*data sets*" del dispositivo híbrido. La diferencia entre los valores de *near-crash* identificados por cada dispositivo es posible observarla a simple vista al comparar la **Figura 38** con la **Figura 40** y la **Figura 39** con la **Figura 41.**

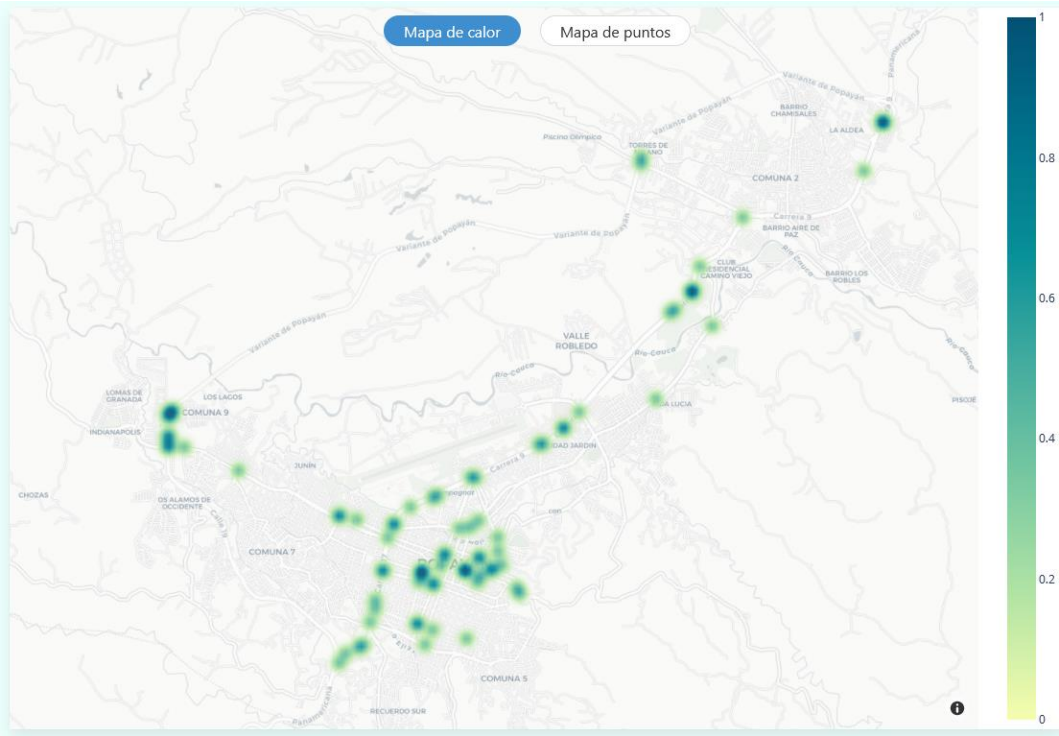

Figura 38. Mapa de calor de los *near-crashes* detectados en datos del *Smartphone*

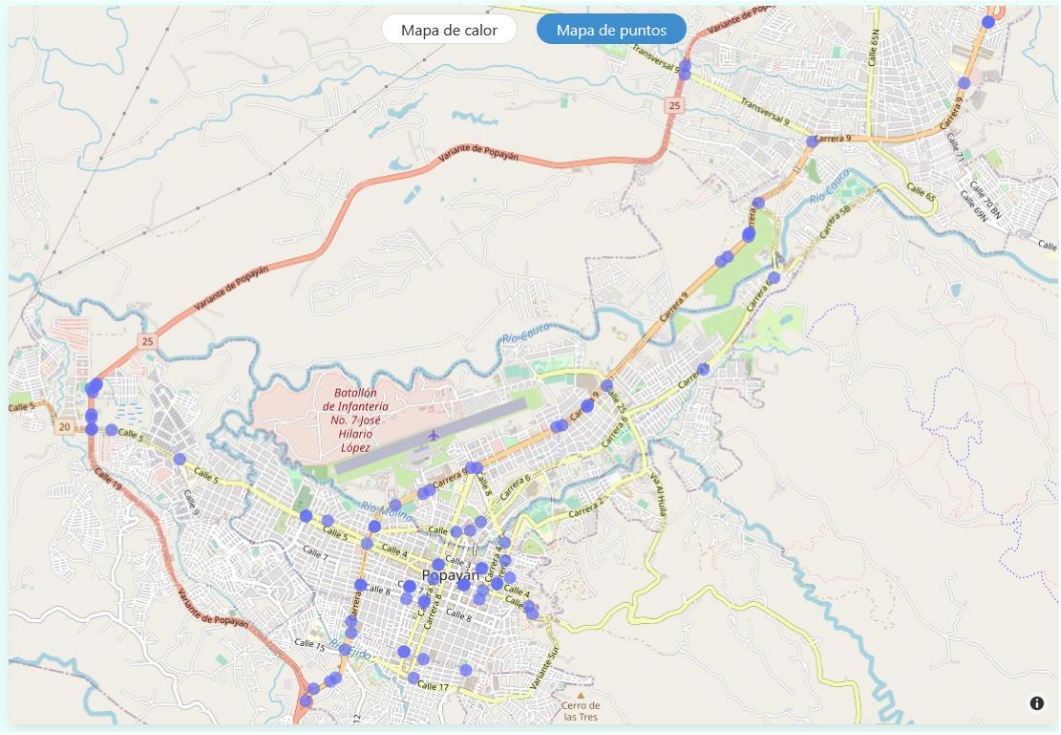

Figura 39. Mapa de puntos de los *near-crashes* detectados en datos del *Smartphone*

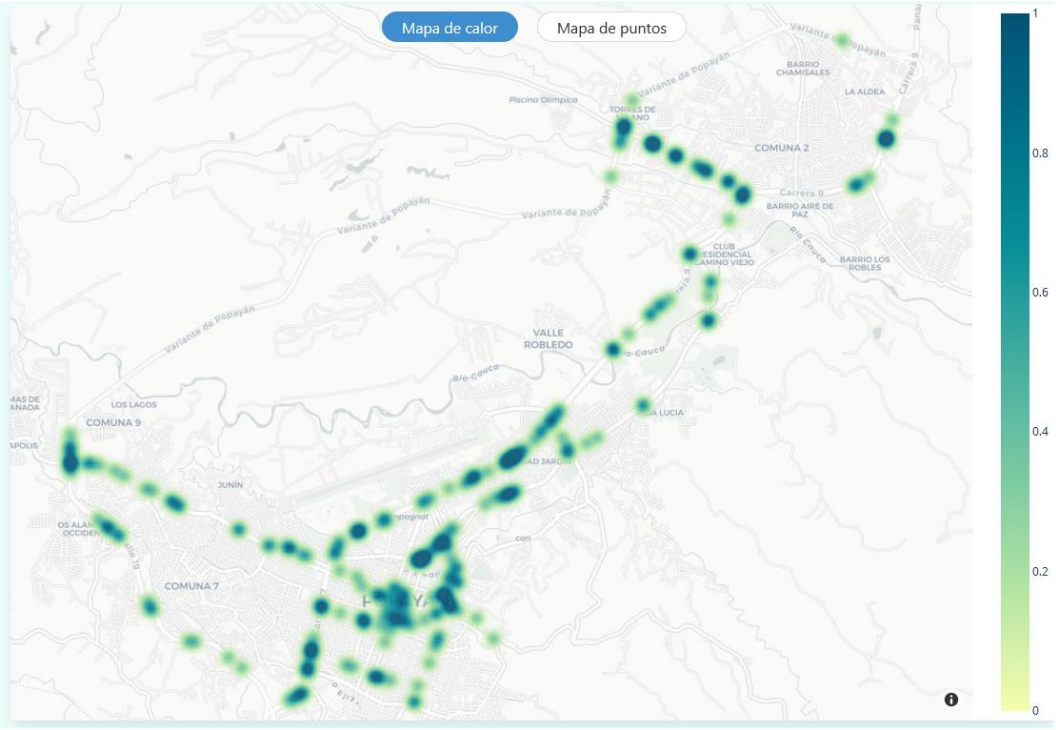

Figura 40. Mapa de calor de los *near-crashes* detectados en datos del dispositivo hibrido

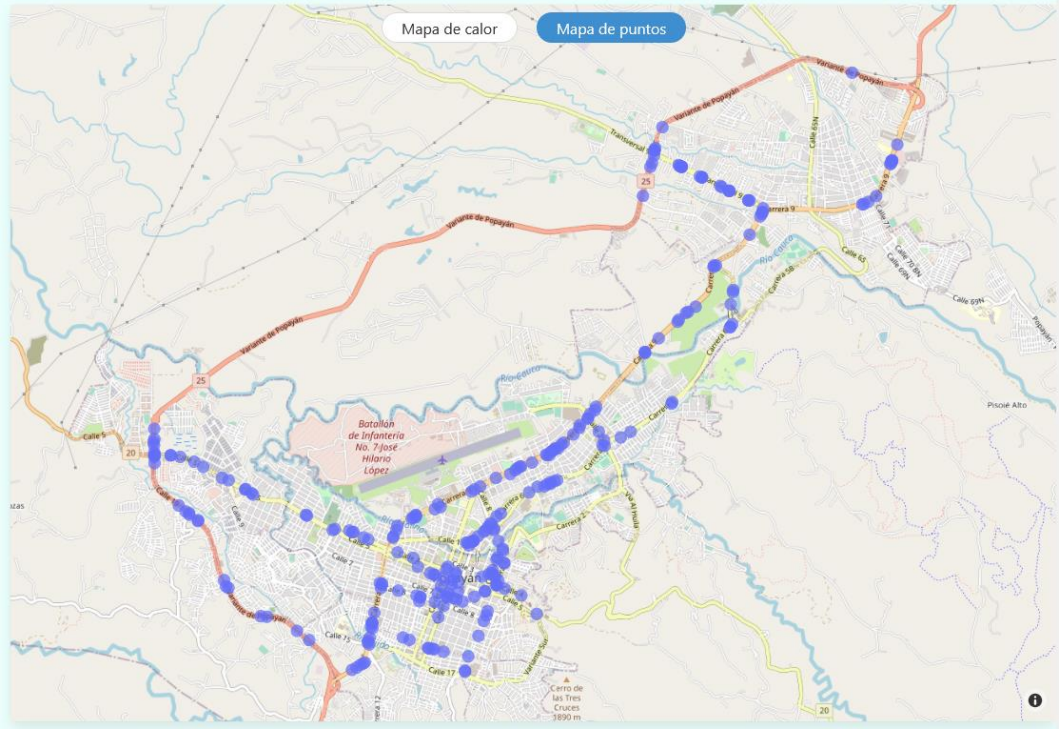

Figura 41. Mapa de puntos de los *near-crashes* detectados en datos del dispositivo hibrido

Para realizar una evaluación más objetiva de los resultados obtenidos a partir del sistema SDIRC, fue llevada a cabo una comparación de estos resultados con estadísticas de accidentalidad de los años 2020, 2021 e inicios del 2022, suministradas por la Secretaría de Transito de Popayán.

La evaluación de los resultados del análisis, realizado a los "*data sets*" de cada dispositivo, estuvo basada en la comparación de la cantidad de *near-crashes* detectados en los dispositivos con la cantidad de accidentes ocurridos en una misma zona de la ciudad.

Las zonas de la ciudad que permitieron la anterior comparación fueron definidas dividiendo el área de la ciudad de Popayán en un sistema de pequeñas cuadriculas. Para esto inicialmente fue definida el área completa de la ciudad dentro de un cuadrado delimitado por la intersección de un rango de latitudes y un rango de longitudes. El rango de latitudes fue establecido desde 2.424 hasta 2.504; y el rango de longitudes fue establecido desde -76.642 hasta -76.550. Esta área puede apreciarse con mayor claridad en la **Figura 42**.

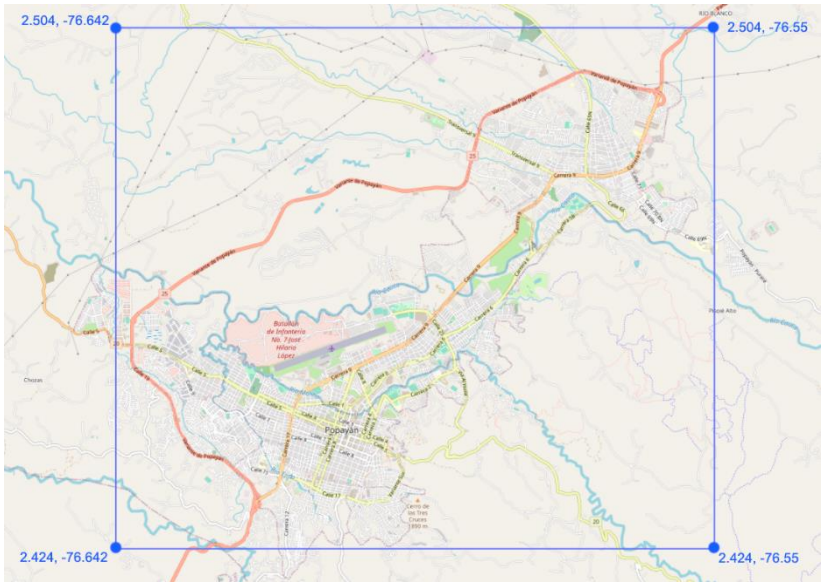

Figura 42. Área completa del sistema de cuadriculas definida para la ciudad de Popayán

Con un área total definida fue posible dividir la ciudad en varias zonas, correspondientes a cuadriculas más pequeñas, cuyas extensiones a lo alto y ancho abarcaron 0.004 grados de latitud y longitud respectivamente. De esta forma, todas las cuadriculas tuvieron el mismo tamaño, permitiendo distribuir uniformemente el área total definida. Tomando como base la herramienta de medición de distancia disponible en Google Maps, el lado de cada cuadricula midió aproximadamente 445 metros, estableciendo un área aproximada de 198.025 m^2 para cada una de ellas. Para una mejor comprensión de lo anterior, es posible observar algunas de estas zonas en las **Figura 43, Figura 44, Figura 45**.

A partir de lo anterior, una zona o cuadricula pudo ser identificada por dos latitudes y dos longitudes delimitadoras y la combinación de cada latitud con cada una de las longitudes hizo posible conocer los 4 vértices de la cuadricula. Siguiendo esta lógica, son listadas las 15 zonas con mayor número de *near-crashes* detectados tanto para los datos del *Smartphone* como del dispositivo híbrido (**Tabla 25 y Tabla 26**, respectivamente).

Tabla 25. Zonas con mayor número de *near-crashes* detectados según análisis de datos de *Smartphone*

Zona	Rango de latitudes	Rango de Longitudes	Near-crashes detectados
1	(2.44, 2.444]	(-76.606, -76.602]	8
2	(2.456, 2.46]	(-76.642, -76.638]	6
3	(2.44, 2.444]	(-76.61, -76.606]	5
4	(2.472, 2.476]	(-76.582, -76.578]	4
5	(2.44, 2.444]	(-76.614, -76.61]	4
6	(2.488, 2.492]	(-76.562, -76.558]	3
7	(2.444, 2.448]	(-76.618, -76.614]	3
8	(2.432, 2.436]	(-76.614, -76.61]	3
9	(2.444, 2.448]	(-76.622, -76.618]	3
10	(2.444, 2.448]	(-76.606, -76.602]	3
11	(2.448, 2.452]	(-76.606, -76.602]	2
12	(2.452, 2.456]	(-76.598, -76.594]	2
13	(2.452, 2.456]	(-76.642, -76.638]	2
14	(2.456, 2.46]	(-76.598, -76.594]	2
15	(2.468, 2.472]	(-76.586, -76.582]	2

Tabla 26. Zonas con mayor número de *near-crashes* detectados según análisis de datos de dispositivo híbrido

Zona	Rango de latitudes	Rango de Longitudes	Near-crashes detectados
1	(2.444, 2.448]	(-76.606, -76.602]	18
2	(2.44, 2.444]	(-76.61, -76.606]	17
3	(2.44, 2.444]	(-76.606, -76.602]	16
4	(2.452, 2.456]	(-76.598, -76.594]	14
5	(2.456, 2.46]	(-76.594, -76.59]	8
6	(2.432, 2.436]	(-76.618, -76.614]	8
7	(2.48, 2.484]	(-76.578, -76.574]	8
8	(2.484, 2.488]	(-76.59, -76.586]	7
9	(2.44, 2.444]	(-76.614, -76.61]	6
10	(2.452, 2.456]	(-76.602, -76.598]	6
11	(2.484, 2.488]	(-76.586, -76.582]	6
12	(2.448, 2.452]	(-76.606, -76.602]	6
13	(2.448, 2.452]	(-76.638, -76.634]	6
14	(2.452, 2.456]	(-76.642, -76.638]	6
15	(2.436, 2.44]	(-76.618, -76.614]	6

Siguiendo el mismo sistema de cuadriculas fueron identificadas las zonas de la ciudad que presentaron un mayor número de accidentes entre el 2020 y marzo del 2022, según la información recolectada por la Secretaría de Tránsito de Popayán. Estas zonas son listadas en la **Tabla 27** y fueron la referencia comparativa para validar la existencia o no de una

relación entre las zonas con mayor número de *near-crashes* y las zonas donde han ocurrido más accidentes de tránsito.

Tabla 27. Zonas con mayor número de accidentes de tránsito según datos de Secretaría de Tránsito de Popayán

Zona	Rango de latitudes	Rango de Longitudes	Choques detectados
1	(2.452, 2.456]	(-76.598, -76.594]	41
2	(2.44, 2.444]	(-76.61, -76.606]	40
3	(2.456, 2.46]	(-76.594, -76.59]	37
4	(2.44, 2.444]	(-76.606, -76.602]	30
5	(2.456, 2.46]	(-76.59, -76.586]	26
6	(2.444, 2.448]	(-76.61, -76.606]	24
7	(2.472, 2.476]	(-76.582, -76.578]	24
8	(2.452, 2.456]	(-76.602, -76.598]	23
9	(2.448, 2.452]	(-76.602, -76.598]	21
10	(2.448, 2.452]	(-76.606, -76.602]	21
11	(2.448, 2.452]	(-76.61, -76.606]	21
12	(2.444, 2.448]	(-76.614, -76.61]	21
13	(2.444, 2.448]	(-76.622, -76.618]	21
14	(2.452, 2.456]	(-76.638, -76.634]	21
15	(2.436, 2.44]	(-76.61, -76.606]	20
16	(2.452, 2.456]	(-76.642, -76.638]	20
18	(2.444, 2.448]	(-76.606, -76.602]	18
17	(2.444, 2.448]	(-76.618, -76.614]	18
19	(2.436, 2.44]	(-76.606, -76.602]	17
20	(2.48, 2.484]	(-76.566, -76.562]	15

Para obtener una representación más clara de los datos listados en las **Tabla 25, Tabla 26, Tabla 27,** fueron elaborados mapas donde las zonas descritas en estas tablas fueron representadas mediante las pequeñas cuadriculas descritas anteriormente, a cada una de los cuales fue asignado un color de acuerdo con su respetivo número de eventos (accidentes el caso de la **Tabla 27** y *near-crashes* en el caso de las **Tabla 25 y Tabla 26**). Una gama de colores que va desde tonos rojizos hasta tonos amarillos, pasando por tonos naranja, fue utilizada para diferenciar las zonas que presentaron un mayor número de eventos de aquellas con cantidades medias y con las cantidades más bajas. Estas representaciones son presentadas en las **Figura 43, Figura 44, Figura 45**.

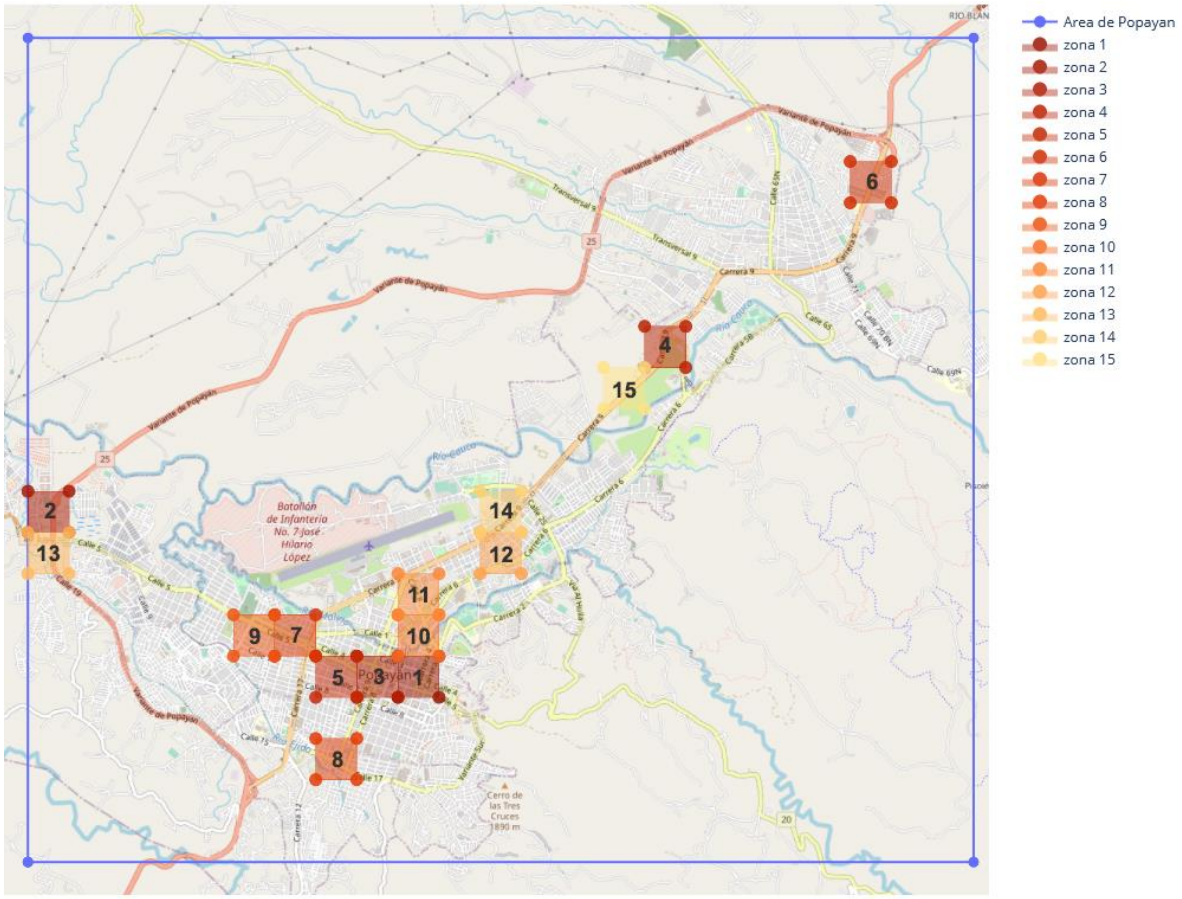

Figura 43. Mapa de zonas con mayor número de *near-crashes* según análisis de datos de *Smartphone*

La **Figura 43** permite observar que las zonas con mayor número de *near-crashes* identificados a partir del análisis de los "*data sets*" del *Smartphone,* corresponden a sectores centro histórico o zonas 1 y 3 (carreras 3° a 9° y calles 7° a 3°), la unión de la calle 5 con la variante (salida al Tambo) o zonas 2 y 13, los sectores de la piedra norte o zona 4 y Pandiguando o zonas 7 y 9. Después de las anteriores zonas, con una concentración mucho más baja de *near-crashes* están los sectores del Barrio Bolívar y la Lotería (zona 10), la glorieta de Toscana (zona 11) y Catay (zonas 12 y 14).

Es posible apreciar que varios puntos críticos de la ciudad según [74], mencionados en el **Anexo K. Revisión y diseño de las pruebas de campo de ND**, se encuentran dentro las zonas anteriormente identificadas. Aunque estos resultados podrían dar validez al sistema y metodologías del presente proyecto, es necesario tener en cuenta los resultados de comparación inicialmente planteados, los cuales son descritos más adelante.

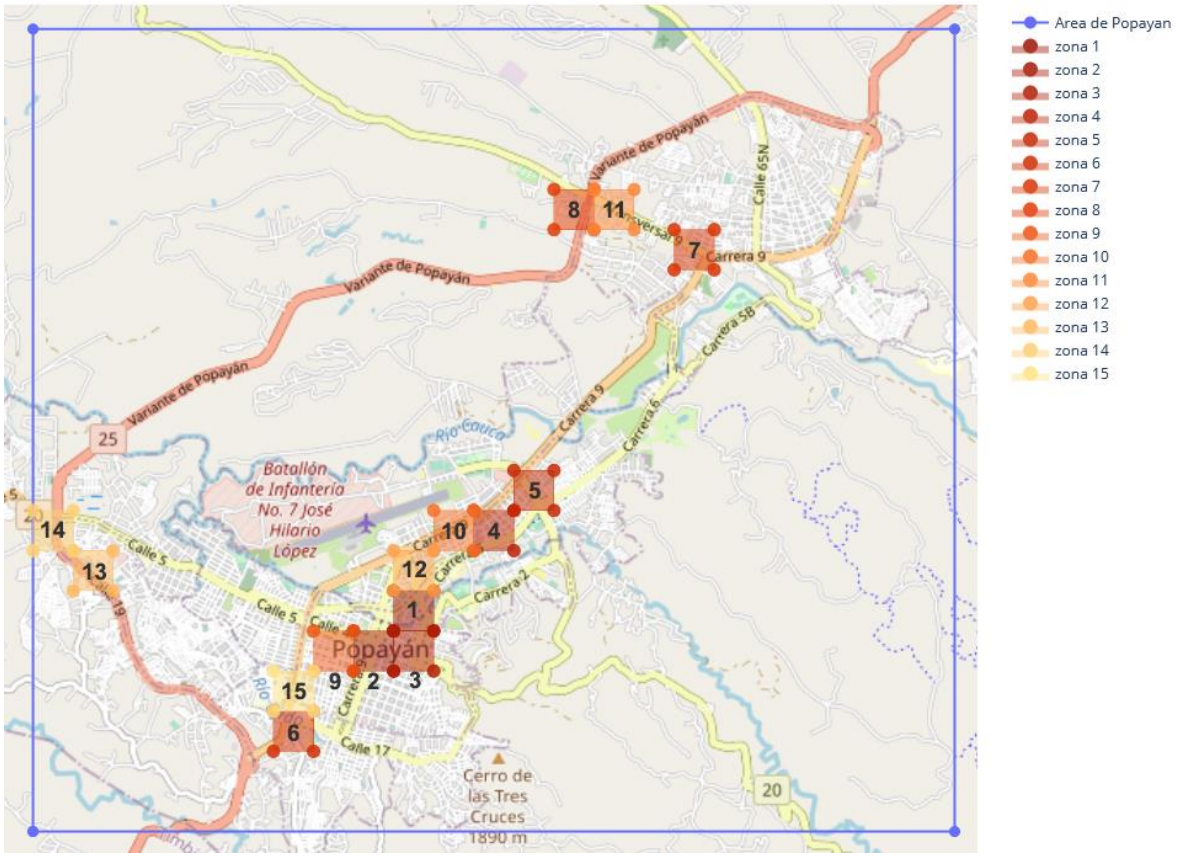

Figura 44. Mapa de zonas con mayor número de *near-crashes* según análisis de datos de dispositivo híbrido

La **Figura 44** evidencia que los resultados obtenidos a partir de los "*data sets*" del dispositivo hibrido presentaron una similitud clara con los del *Smartphone* al coincidir en el sector del centro histórico (zonas 2 y 3 según la **Figura 44**) como una de las áreas con más *near-crashes* detectados. Sin embargo, en el caso del dispositivo híbrido, aparecen otras zonas de alta concentración de *near-crashes*. La zona 1 es la más sobresaliente y abarca los sectores del Barrio Bolívar, el centro comercial la Estación y la glorieta de la Lotería; después de esta zona están los sectores cercanos a Catay (zonas 4 y 10), Campanario (zona 5), la Piedra sur (zona 6), Bellavista (zona 7) y la unión de la variante con la transversal 9A norte (zonas 8 y 11).

El sector de la salida al Tambo (zona 14) también está presente en la **Figura 44** pero con una concentración relativamente más baja en comparación con las anteriormente mencionadas.

Al igual que con los resultados del *Smartphone,* muchos puntos de movilidad críticos, mencionados en el **Anexo K. Revisión y diseño de las pruebas de campo de ND**, también pertenecen a las zonas con mayor concentración de *near-crashes* identificadas a partir de los "*data sets*" del dispositivo híbrido.

A través de las figuras **Figura 38** y **Figura 40** es posible apreciar que el dispositivo híbrido y los modelos ML utilizados (para este tipo de dispositivo), tienden a ser mucho más sensibles en la detección de *near-crashes*. A pesar de ello, llama la atención apreciar que

en la unión de la vía Panamericana con la vía variante norte (zona 6 de la **Figura 43**) fue detectada una cantidad significativa de *near-crashes* a partir de los "*data sets*" y modelos de ML del *Smartphone*, mientras que con los del dispositivo híbrido no fue detectado ninguno.

Una vez identificadas y graficadas las zonas con mayor número de *near-crashes* para ambos dispositivos, fueron graficadas las zonas con mayor número de AT (teniendo en cuenta la **Tabla 27**), permitiendo así evaluar alguna relación cuantitativa entre la ocurrencia de *near-crashes* en una determinada zona y la posible aparición de un verdadero AT.

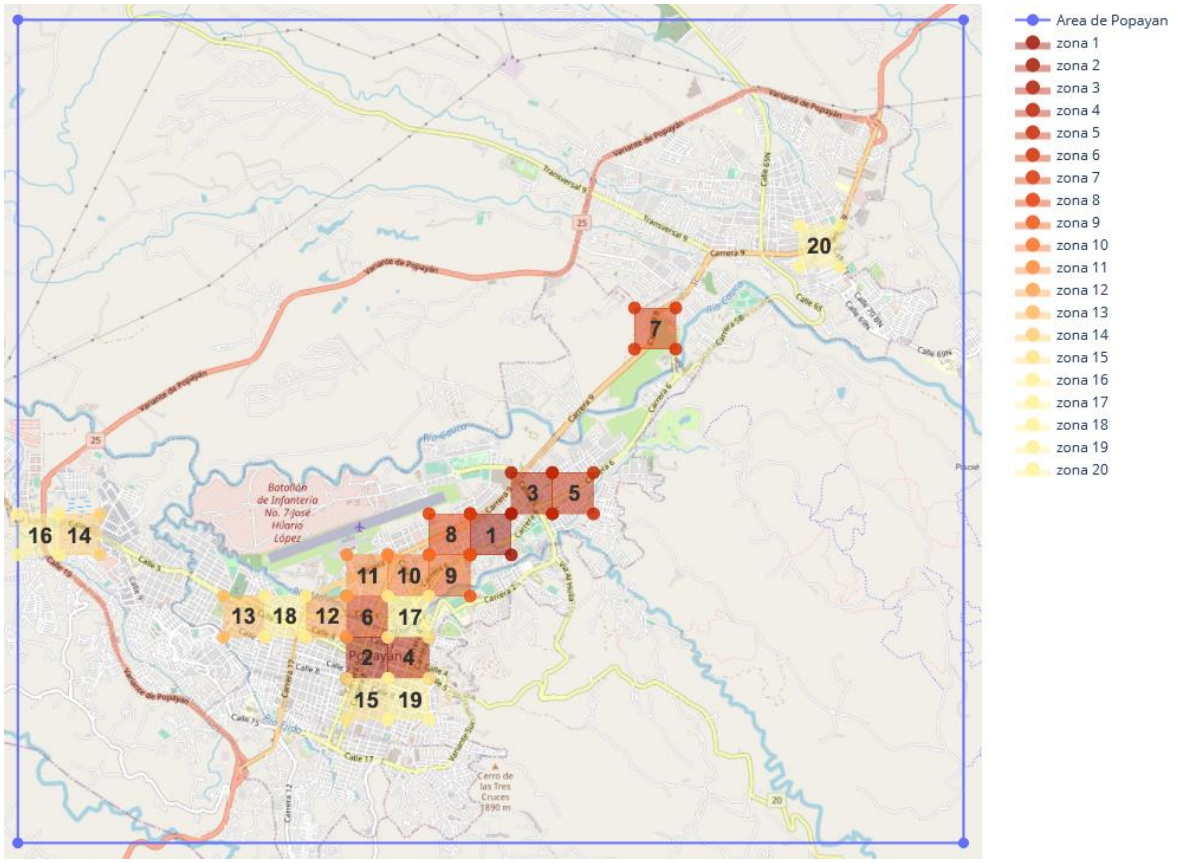

Figura 45. Mapa de zonas con mayor número de accidentes según datos de secretaría de tránsito Popayán

En la **Figura 45** son señaladas las 20 zonas con mayor densidad de AT, ocurridas principalmente en los últimos dos años. Estas zonas fueron tomadas como referencia para evaluar la hipótesis inicialmente planteada a partir de los resultados del presente proyecto. Para esto, fue elaborada la **Tabla 28** con el propósito de buscar una correlación entre las zonas de mayor ocurrencia de *near-crashes,* ya identificadas, y las zonas donde efectivamente han ocurrido un gran número de AT.

La **Tabla 28** parte de una "zonas de referencia", las cuales corresponden a las 20 zonas con mayor densidad de AT. Estas son identificadas con el mismo número asignado tanto en la **Tabla 28** como en la **Figura 45**. A cada una de ellas le fue agregada una descripción breve con el fin de facilitar su reconocimiento y ubicación, además del número de AT presentados en la zona. Por otro lado, las "zonas equivalentes" corresponden a una o dos

de las 15 zonas con mayor ocurrencia de *near-crashes* (según los datos de cada dispositivo) que coinciden geográficamente o son muy cercanas a una zona de referencia.

Tabla 28. Relación entre zonas con mayor número de AT y zonas con mayor ocurrencia de *near-crashes*

Zona de referencia	Descripción	No. AT	*Smartphone*		Disp. Híbrido	
			Zona equivalente	No. *near-crashes*	Zona equivalente	No. *near-crashes*
1	Sectores Catay, ciudad Jardín	41	**12**	2	**4**	14
2	Centro (sector occidental)	40	**3**	5	**2**	17
3	Campanario, semáforo de los Hoyos	37	**14**	2	**5**	8
4	Centro (sector oriental)	30	**1**	8	**3**	16
5	Los Hoyos, Yambitará	26	**-**	-	**-**	-
6	Barrio el Modelo (Cra 9° con Cll. 1N)	24	**-**	-	**-**	-
7	Piedra Norte	24	**4 y 15**	6	**-**	-
8	Vía panamericana entre sector el Recuerdo, barrio Alcalá y Santa Clara	23	**-**	-	**10**	6
9	Cra. 6° Sector Bolívar, Hospital San José, El Recuerdo	21	**-**	-	**-**	-
10	Barrio Belalcázar, glorieta La Toscana	21	**11**	2	**12**	6
11	Vía Panamericana sector Terminal de transportes, Colegio Champagnat	21	**-**	-	**-**	-
12	Barrio Cadillal (Cll. 1, Cll. 4)	21	**-**	-	**-**	-
13	Cementerio central, barrios José María Obando y Santa Elena (Cll 5°, Cll. 4°)	21	**9**	3	**-**	-
14	Calle 5° sector María Occidente	21	**-**	-	**-**	-
15	Sector sur occidental del centro	20	**-**	-	**-**	-
16	Salida al Tambo (unión de variante con calle 5°	20	**2 y 13**	8	**14**	6

17	Plaza barrio Bolívar, glorieta La Lotería, centro comercial La Estación (Cra. 7)	18	**10**	3	**1**	18
18	Pandiguando (Cll. 5° y Cll. 4°)	18	**7**	3	**-**	-
19	Sector sur oriental del centro	17	**-**	-	**-**	-
20	Vía Panamericana sector Sena Norte	15	**-**	-	**-**	-

De acuerdo con las relaciones establecidas en la anterior tabla, de las 15 zonas con mayor ocurrencia de *near-crashes*, identificadas a partir de los datos del *Smartphone,* es posible afirmar que*:*

- 12 de estas zonas coincidieron con alguna de las 20 zonas de referencia seleccionadas.
- El top 4 de estas zonas coincidieron de forma continua con alguna zona de referencia. A partir de la quinta (5°) zona, las coincidencias ocurrieron de forma discontinua o intermitente.

Del mismo modo, para las 15 zonas con mayor ocurrencia de *near-crashes*, identificadas a partir de los datos del dispositivo híbrido*,* es posible afirmar que*:*

- 8 de estas zonas coincidieron con alguna de las 20 de referencia seleccionadas.
- El top 5 de estas zonas coincidieron de forma continua con alguna zona de referencia. A partir de la sexta (6°) zona, las coincidencias ocurrieron de forma discontinua o intermitente.

Según lo anterior, la distribución espacial de los *near-crashes* detectados a partir del uso del *Smartphone* y sus modelos de ML tuvo una mayor similitud con la distribución espacial de las AT ocurridas en los últimos años. Lo anterior debido a que la mayoría de las zonas con mayor número de *near-crashes* detectados con este dispositivo, correspondieron con alguna de las 20 zonas donde efectivamente han ocurrido mayor número de AT.

También, es posible observar que el número de AT ocurridas en una zona de referencia no tuvo una relación proporcional con el número de *near-crashes* detectados en la misma zona. Esto aplicó tanto para los resultados conseguidos con el *Smartphone* como para los del dispositivo híbrido. Para ilustrar mejor la anterior idea, se puede tomar como ejemplo las zonas con mayor número de *near-crashes* detectados con los datos y modelos de cada dispositivo (zonas identificadas con el número 1); siendo posible verificar en la **Tabla 25** y **Tabla 26** que estas zonas no coincidieron con la zona de mayor número de AT (zona de referencia 1). En el caso del *Smartphone,* la zona con mayor cantidad de *near-crashes* coincidió con la zona de referencia 4 (cuarta zona con mayor número de AT); y en el caso del dispositivo híbrido coincidió con la zona de referencia 17 (décimo séptima zona con mayor número de AT).

Capítulo 8

8. Conclusiones y trabajos futuros

8.1. Conclusiones

Partiendo de la hipótesis inicialmente planteada, donde se afirma: “Es posible determinar las zonas con alta probabilidad de AT con ayuda de los *near-crashes*, utilizando una aplicación que use modelos de ML para el análisis de datos recolectados mediante pruebas de campo de conducción naturalista”; y considerando las diferentes etapas de desarrollo del presente trabajo, fue posible concluir lo siguiente:

El sistema SDIRC fue diseñado y desarrollado teniendo en cuenta alternativas idóneas para los parámetros más importantes identificados con base en la revisión de la literatura realizada. Estas fueron: las variables estrechamente relacionadas con la cinemática y definición de un *near-crash*; los dispositivos que pudieran medir y recolectar todas las variables necesarias; y los algoritmos de ML con mejor desempeño, mayor cantidad de implementaciones dentro de la literatura revisada y usados primordialmente con propósitos similares al objetivo del presente trabajo. Además, este sistema estuvo conformado por un módulo de recolección de datos y un módulo de análisis inteligente de información.

Durante la elaboración del módulo de recolección de datos, fue posible comprobar que el desarrollo del dispositivo hibrido resulto ser más complejo que el desarrollo de la aplicación móvil del *Smartphone*, esto debido, principalmente, a los errores presentados en el hardware del dispositivo y a la dificultad que implica garantizar una integración exitosa de todos los componentes. El error más significativo estuvo relacionado con el dispositivo ELM 327, ya que resultó imposible emparejar este dispositivo con algunos vehículos modernos, debido al manejo de nuevos protocolos en el puerto CAN o protocolos privativos por parte de la empresa fabricante de los vehículos.

Igualmente, mediante el análisis del desempeño y rendimiento de los modelos de ML desarrollados fue posible evidenciar que los resultados son muy similares a los obtenidos por otros artículos relacionados [26], [34]. Además, se demostró que el uso de una ventana de tiempo grande para manejar datos con series temporales aumentó el rendimiento de los algoritmos.

A partir del módulo RD desarrollado para el prototipo SDIRC, fue posible la recolección de datos de conducción naturalista mediante la ejecución de las pruebas de campo de validación diseñadas para identificar zonas con alta probabilidad de AT en la ciudad de Popayán. Posteriormente, el módulo AII permitió analizar todos los datos recolectados dando a conocer las ubicaciones de cada uno de los *near-crashes* detectados.

En este sentido, el análisis de los resultados obtenidos (una vez el sistema SDIRC fue implementado) fue realizado mediante una evaluación comparativa de dichos resultados con datos de accidentalidad suministrados (casi al finalizar el proyecto) por parte de la Secretaría de Tránsito de Popayán. Esta evaluación utilizó un sistema de cuadriculas para definir y comparar zonas con una alta ocurrencia de *near-crashes* y zonas con mayor densidad de AT.

Los resultados de este análisis demuestran que existe una relación entre las zonas con una alta ocurrencia de *near-crashes* y las zonas donde las AT suelen producirse más frecuentemente. Esto fue posible evidenciarlo al observar que, independientemente del dispositivo con el que fueron recolectados los datos, la mayoría de las zonas con un alto número de *near-crashes* ocurridos concordaron con alguna de las zonas con mayor densidad de AT.

Por lo tanto, si consideramos el número de *near-crashes* detectados a partir de los datos recolectados y la ejecución del modelo de ML, el dispositivo híbrido desarrollado resulta más conveniente que el *Smartphone* en la detección de estos eventos (con una diferencia de 187 *near-crash*). Sin embargo, esto debería ser validado reentrenando los algoritmos de ML con muchos más datos y usando diferentes filtros y/o técnicas pre-procesamiento. Por otro lado, si consideramos el número coincidencias entre las “zonas equivalentes” y “zonas de referencia” definidas, el *Smartphone* obtuvo un mejor desempeño.

Sin embargo, no fue posible apreciar una relación proporcional directa entre la cantidad de *near-crashes* detectados y la cantidad de AT ocurridas en una misma zona. La causa de esto posiblemente esté relacionada al bajo número de recorridos realizados. Siguiendo esta idea, realizar pruebas de campo con muchas más rutas a las utilizadas en este trabajo, empleando más conductores y realizando repeticiones de los recorridos en diferentes horarios, podría ayudar a descubrir tendencias y relaciones entre las cantidades de *near-crashes* y AT para una determinada zona.

A pesar de lo anterior, el enfoque propuesto y evaluado a partir del sistema SDIRC para identificar zonas con alta probabilidad de AT, presenta una ventaja clara en comparación con las técnicas tradicionales de recolección de datos estadísticos sobre accidentalidad vial. El uso de *near-crashes* y la técnica de ND permiten obtener datos relacionados con la seguridad vial en tiempos más cortos y sin necesidad de que un incidente tenga que ocurrir. Esto permite realizar diagnósticos y análisis de forma más ágil y sus resultados podrían verse reflejados en la implementación de planes de prevención de AT, modificaciones a las políticas de movilidad o mejoras en la infraestructura vial.

Por último, gracias a la aplicación de la metodología SCRUM, la plataforma ICRDS web desarrollada para el prototipo SDIRC fue implementada cumpliendo los requerimientos con prioridad más alta. Esto permitió demostrar que el sistema de manejo de datos de accidentalidad funcionó correctamente y su integración con los algoritmos de ML fue exitosa. Así mismo, las interfaces y dispositivos de recolección de datos fueron desarrollados tomando como prioridad las necesidades del usuario.

8.2. Trabajos futuros

En base al desarrollo del prototipo y los resultados presentados en el trabajo de grado, son realizadas las siguientes recomendaciones y propuestas para los trabajos futuros relacionados con la investigación:

- Agregar nuevas funcionalidades en el sistema SDIRC, como nuevas formas de visualizar los datos, creación de mapas con el sistema de cuadricula desde la interfaz del usuario, funcionalidades de limpieza de datos automáticas, manipulación de los hiper-parámetros del algoritmo ML, gestión de roles y servicios de autenticación y autorización.

- Escalar el módulo de recolección de datos (RD) con funcionalidades como un algoritmo de reorientación, aplicar arquitecturas como "*publish–subscribe pattern"* para él envió constante de datos a la "nube" y en tiempo real hacia la plataforma ICRDS web. De esta manera el prototipo podría adoptar un enfoque basado en tecnologías IoT y ampliar sus posibles aplicaciones.

- Escalar el módulo de análisis inteligente de información (AII) añadiendo más funcionalidades como el procesamiento de los datos a través de herramientas en la "nube", ejecución de algoritmos en la "nube" y la implementación de procesos automatizados de tipo ETL (*extract, transform and load*) o ELT (*extract, load and transform*) para la gestión y transformación de datos.

- Aplicar un nuevo filtro u optimizar el filtro Kalman utilizado en el SDIRC, ya que el uso de este filtro incrementó considerablemente el tiempo de pre-procesamiento de los datos.

- Mejorar el modelo de ML a partir de los nuevos datos capturados en las pruebas de campo de conducción naturalista (ND) o reentrenarlo usando datos de maniobras ejecutadas por conductores con diferentes estilos de conducción.

- Abordar nuevas técnicas de ML para la clasificación de eventos de *near-crash*, estas técnicas pueden usar algoritmos no supervisados o redes neuronales como los trabajos de [39], [75].

- Ejecutar pruebas de campo de validación realizando más recorridos en nuevas rutas o en las diseñadas para este trabajo, empleando un mayor número de conductores, con el fin de recolectar más datos de ND y realizar una evaluación más completa del sistema.

- Evaluar la usabilidad del sistema en entidades como la secretaria de tránsito o en entidades relacionadas con el área de transporte y seguridad vial. Así Como su inclusión en el plan de mejora de seguridad vial de la OMS, proclamado como "Década de Acción para la Seguridad Vial 2021-2030".

Bibliografía

[1] WHO, “Accidentes de tránsito,” 2018. https://www.who.int/es/news-room/fact-sheets/detail/road-traffic-injuries (accessed Oct. 14, 2020).

[2] WHO, “Death on the roads,” 2018. https://extranet.who.int/roadsafety/death-on-the-roads/#deaths (accessed Oct. 14, 2020).

[3] Grupo centro de referencia nacional sobre violencia, “Reporte Forensis 2018. Datos para la Vida,” 2018. Accessed: Jun. 13, 2022. [Online]. Available: https://www.medicinalegal.gov.co/documents/20143/386932/Forensis+2018.pdf

[4] Agencia Nacional de Seguridad Víal, “Observatorio Nacional de Seguridad Vial,” 2020. https://ansv.gov.co/es/observatorio (accessed Oct. 21, 2020).

[5] Congreso de Colombia, *Ley 1702 Creación de la Agencia Nacional*. 2013.

[6] Grupo de seguridad vial, “Plan nacional de seguridad vial Colombia 2011-2021,” 2015. Accessed: Jun. 13, 2022. [Online]. Available: https://ansv.gov.co/sites/default/files/Documentos/Agencia/mipg/1-5-5-docs-e-informes/1 - 5 - 5 - 29 Plan Nacional de Seguridad Vial/Plan Nacional de Seguridad Vial.pdf

[7] J. M. Muñoz, “Estudio sobre patrones de accidentes en la ciudad de València,” Universitat Politècnica de València, España, 2018. Accessed: Jun. 13, 2022. [Online]. Available: https://m.riunet.upv.es/bitstream/handle/10251/111763/Montesinos - Estudio sobre patrones de accidentes en la ciudad de València.pdf?sequence=1&isAllowed=y

[8] Google Trends, “Data science - Explore - Google Trends,” *Google*, 2019. https://trends.google.com/trends/explore?cat=47&date=today 5-y&q=data science (accessed Oct. 21, 2020).

[9] C. Gutierrez-Osorio and C. Pedraza, “Modern data sources and techniques for analysis and forecast of road accidents: A review,” *Journal of Traffic and Transportation Engineering (English Edition)*, vol. 7, no. 4, pp. 432–446, 2020, doi: 10.1016/j.jtte.2020.05.002.

[10] A. Mehdizadeh, M. Cai, Q. Hu, M. A. A. Yazdi, N. Mohabbati-Kalejahi, A. Vinel, S. E. Rigdon, K. C. Davis, and F. M. Megahed, “A review of data analytic applications in road traffic safety. Part 1: Descriptive and predictive modeling,” *Sensors (Switzerland)*, vol. 20, no. 4, pp. 1–24, 2020, doi: 10.3390/s20041107.

[11] H. A. H. Naji, Q. Xue, N. Lyu, C. Wu, and K. Zheng, “Evaluating the driving risk of near-crash events using a mixed-ordered logit model,” *Sustainability (Switzerland)*, vol. 10, pp. 1–20, 2018, doi: 10.3390/su10082868.

[12] F. Guo, S. G. Klauer, J. M. Hankey, and T. A. Dingus, “Near crashes as crash surrogate for naturalistic Driving Studies,” *Transportation Research Record*, pp. 66–74, 2010, doi: 10.3141/2147-09.

[13] P. M. Institute, *PMBOK GUIDE*, 6th ed. Project Management Institute, 2017.

[14] K. Schwaber and J. Sutherland, *The Scrum Guide: The Definitive The Rules of the*

Game. 2017. [Online]. Available: http://www.scrumguides.org/docs/scrumguide/v1/Scrum-Guide-US.pdf

[15] IBM, "Conceptos básicos de ayuda de CRISP-DM," *Ibm*, 2019. https://www.ibm.com/support/knowledgecenter/es/SS3RA7_sub/modeler_crispdm_ddita/clementine/crisp_help/crisp_overview.html (accessed Oct. 21, 2020).

[16] C. Pete, C. Julian, K. Randy, K. Thomas, R. Thomas, S. Colin, and R. Wirth, "Crisp-Dm 1.0," *CRISP-DM Consortium*, p. 76, 2000, [Online]. Available: https://www.kde.cs.uni-kassel.de/wp-content/uploads/lehre/ws2012-13/kdd/files/CRISPWP-0800.pdf

[17] K. F. Wu and P. Jovanis, "Screening Naturalistic Driving Study Data for Safety-Critical Events:," *https://doi.org/10.3141/2386-16*, no. 2386, pp. 137–146, Jan. 2013, doi: 10.3141/2386-16.

[18] H. Singh and A. Kathuria, "Analyzing driver behavior under naturalistic driving conditions: A review," *Accident Analysis & Prevention*, vol. 150, p. 105908, Feb. 2021, doi: 10.1016/J.AAP.2020.105908.

[19] I. van Schagen and F. Sagberg, "The Potential Benefits of Naturalistic Driving for Road Safety Research: Theoretical and Empirical Considerations and Challenges for the Future," 2012, doi: 10.1016/j.sbspro.2012.06.1047.

[20] A. Ziakopoulos, D. Tselentis, A. Kontaxi, and G. Yannis, "A critical overview of driver recording tools," *Journal of Safety Research*, vol. 72, pp. 203–212, 2019, doi: 10.1016/j.jsr.2019.12.021.

[21] T. A. Dingus, S. G. Klauer, V. L. Neale, A. Petersen, S. E. Lee, J. D. Sudweeks, M. A. Perez, J. Hankey, D. J. Ramsey, S. Gupta, C. Bucher, Z. R. Doerzaph, J. Jermeland, and R. R. Knipling, "The 100-Car Naturalistic Driving Study Phase II-Results of the 100-Car Field Experiment," Virginia, 2006. Accessed: Mar. 28, 2022. [Online]. Available: https://www.nhtsa.gov/sites/nhtsa.gov/files/100carmain.pdf

[22] SHRP2 TRB, "SHRP 2 | Strategic Highway Research Program 2 (SHRP 2)," 2015. http://www.trb.org/StrategicHighwayResearchProgram2SHRP2/Blank2.aspx (accessed Nov. 02, 2020).

[23] M. A. Perez, J. D. Sudweeks, E. Sears, J. Antin, S. Lee, J. M. Hankey, and T. A. Dingus, "Performance of basic kinematic thresholds in the identification of crash and near-crash events within naturalistic driving data," *Accident Analysis and Prevention*, vol. 103, pp. 10–19, 2017, doi: 10.1016/j.aap.2017.03.005.

[24] E. I. Vlahogianni and E. N. Barmpounakis, "Driving analytics using smartphones: Algorithms, comparisons and challenges," *Transportation Research Part C: Emerging Technologies*, vol. 79, pp. 196–206, 2017, doi: 10.1016/j.trc.2017.03.014.

[25] J. Wahlström, I. Skog, and P. Händel, "Detection of Dangerous Cornering in GNSS-Data-Driven Insurance Telematics," *IEEE Transactions on Intelligent Transportation Systems*, vol. 16, no. 6, pp. 3073–3083, Dec. 2015, doi: 10.1109/TITS.2015.2431293.

[26] M. Wu, S. Zhang, and Y. Dong, "A novel model-based driving behavior recognition system using motion sensors," *Sensors (Switzerland)*, vol. 16, 2016, doi: 10.3390/s16101746.

[27] S. Daptardar, V. Lakshminarayanan, S. Reddy, S. Nair, S. Sahoo, and P. Sinha,

"Hidden Markov Model based driving event detection and driver profiling from mobile inertial sensor data," *2015 IEEE SENSORS - Proceedings*, 2015, doi: 10.1109/ICSENS.2015.7370312.

[28] E. Wolfgang, *Introduction to Artificial Intelligence*, 2th ed. Switzerland, 2017. doi: 10.1007/978-3-319-58487-4.

[29] L. Igual and S. Seguí, *Introduction to Data Science: A Python Approach to Concepts, Techniques and Applications*. Switzerland, 2017. doi: 10.1007/978-3-319-50017-1.

[30] A. C. Müller and S. Guido, *Introduction to Machine Learning with Python*, 1st ed. United States of America, 2016. Accessed: Mar. 28, 2022. [Online]. Available: https://www.oreilly.com/library/view/introduction-to-machine/9781449369880/

[31] J. VanderPlas, *Python Data Science Handbook*, 1st ed. United States of America, 2016. Accessed: Mar. 28, 2022. [Online]. Available: https://www.oreilly.com/library/view/python-data-science/9781491912126/

[32] M. Kubat, *An Introduction to Machine Learning*, 2nd ed. Switzerland, 2017. doi: 10.1007/978-3-319-63913-0.

[33] A. P. Bradley, "The use of the area under the ROC curve in the evaluation of machine learning algorithms," *Pattern Recognition*, vol. 30, no. 7, pp. 1145–1159, Jul. 1997, doi: 10.1016/S0031-3203(96)00142-2.

[34] J. Ferreira, E. Carvalho, B. V. Ferreira, C. De Souza, Y. Suhara, A. Pentland, and G. Pessin, "Driver behavior profiling: An investigation with different smartphone sensors and machine learning," *PLoS ONE*, vol. 12, pp. 1–16, 2017, doi: 10.1371/journal.pone.0174959.

[35] D. Moher, A. Liberati, J. Tetzlaff, D. G. Altman, and The PRISMA Group, "Preferred reporting items for systematic reviews and meta-analyses: The PRISMA statement," *PLoS Medicine*, vol. 6, no. 7, 2009, doi: 10.1371/journal.pmed.1000097.

[36] O. A. Osman, M. Hajij, P. R. Bakhit, and S. Ishak, "Prediction of Near-Crashes from Observed Vehicle Kinematics using Machine Learning," *Transportation Research Record*, 2019, doi: 10.1177/0361198119862629.

[37] J. Wang, Y. Zheng, X. Li, C. Yu, K. Kodaka, and K. Li, "Driving risk assessment using near-crash database through data mining of tree-based model," *Accident Analysis and Prevention*, vol. 84, pp. 54–64, 2015, doi: 10.1016/j.aap.2015.07.007.

[38] J. Bao, P. Liu, and S. V. Ukkusuri, "A spatiotemporal deep learning approach for citywide short-term crash risk prediction with multi-source data," *Accident Analysis and Prevention*, vol. 122, pp. 239–254, 2019, doi: 10.1016/j.aap.2018.10.015.

[39] S. Liu, K. Koch, B. Gahr, and F. Wortmann, "Brake Maneuver Prediction - An Inference Leveraging RNN Focus on Sensor Confidence," *2019 IEEE Intelligent Transportation Systems Conference, ITSC 2019*, pp. 3249–3255, 2019, doi: 10.1109/ITSC.2019.8917405.

[40] F. Pedregosa, "Scikit-learn: Machine Learning in Python," *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011, Accessed: May 26, 2021. [Online]. Available: http://scikit-learn.sourceforge.net.

[41] E. Frank, M. A. Hall, and I. H. Witten, *The WEKA workbench*. 2017. doi:

10.1016/b978-0-12-804291-5.00024-6.

[42] Deisenroth, *Mathematics for ML*. 2020. [Online]. Available: http://www.maa.org/external_archive/QL/pgs75_89.pdf

[43] C. C. Chang and C. J. Lin, "LIBSVM: A Library for support vector machines," *ACM Transactions on Intelligent Systems and Technology*, vol. 2, pp. 1–39, 2011, doi: 10.1145/1961189.1961199.

[44] ISO, "Road vehicles — Controller area network (CAN) — Part 1: Data link layer and physical signalling." 2015. Accessed: May 26, 2021. [Online]. Available: https://www.iso.org/obp/ui/#iso:std:iso:11898:-1:ed-2:v1:en

[45] ISO, "Road vehicles — Communication between vehicle and external equipment for emissions-related diagnostics — Part 1: General information and use case definition." 2010. Accessed: May 26, 2021. [Online]. Available: https://www.iso.org/obp/ui/#iso:std:iso:15031:-1:ed-2:v1:en

[46] P. Wu and H. Zhao, "Some Analysis and Research of the AdaBoost Algorithm," *Communications in Computer and Information Science*, vol. 134, no. PART 1, pp. 3–5, 2011, doi: 10.1007/978-3-642-18129-0_1.

[47] United States Department of Transportation, "ARC-IT Architecture Reference for Cooperative and Intelligent Transportation," *Architecture Reference for Cooperative and Intelligent Transportation*, 2019. https://local.iteris.com/arc-it/ (accessed Jun. 24, 2021).

[48] AustriaTech, "FRAME ARCHITECTURE," *FRAME Forum*, 2000. https://frame-online.eu/ (accessed Jun. 24, 2021).

[49] M. Amarasinghe, S. Kottegoda, A. L. Arachchi, S. Muramudalige, H. M. N. Dilum Bandara, and A. Azeez, "Cloud-based driver monitoring and vehicle diagnostic with OBD2 telematics," *IEEE International Conference on Electro Information Technology*, vol. 2015-June, pp. 505–510, 2015, doi: 10.1109/EIT.2015.7293433.

[50] S. A. Nugroho, E. Ariyanto, and A. Rakhmatsyah, "Utilization of Onboard Diagnostic II (OBD-II) on Four Wheel Vehicles for Car Data Recorder Prototype," in *2018 6th International Conference on Information and Communication Technology, ICoICT 2018*, 2018, pp. 7–11. doi: 10.1109/ICoICT.2018.8528741.

[51] Z. Chen, J. Yu, Y. Zhu, Y. Chen, and M. Li, "D3: Abnormal driving behaviors detection and identification using smartphone sensors," *2015 12th Annual IEEE International Conference on Sensing, Communication, and Networking, SECON 2015*, pp. 524–532, 2015, doi: 10.1109/SAHCN.2015.7338354.

[52] Lora Alliance org, "A technical overview of LoRa ® and LoRaWAN ™ What is it?," San Ramon, CA, USA, 2015. Accessed: Apr. 04, 2022. [Online]. Available: https://lora-alliance.org/wp-content/uploads/2020/11/what-is-lorawan.pdf

[53] Pallets, "Flask," 2010. https://flask.palletsprojects.com/en/2.1.x/ (accessed Apr. 05, 2022).

[54] Google Cloud Platform Service, "Firebase," 2022. https://firebase.google.com/ (accessed Apr. 12, 2022).

[55] T. James, "Firebase expands to become a unified app platform," 2016.

https://firebase.blog/posts/2016/05/firebase-expands-to-become-unified-app-platform (accessed Apr. 16, 2022).

[56] "rfc4180." https://datatracker.ietf.org/doc/html/rfc4180 (accessed Oct. 30, 2021).

[57] W. McKinney, "Data Structures for Statistical Computing in Python," *Proceedings of the 9th Python in Science Conference*, pp. 56–61, 2010, doi: 10.25080/MAJORA-92BF1922-00A.

[58] "SQLite Home Page." https://www.sqlite.org/index.html (accessed Oct. 30, 2021).

[59] Google Cloud Platform Service, "Estructura tu base de datos | Firebase Documentation," 2021. https://firebase.google.com/docs/database/web/structure-data?hl=es (accessed Apr. 12, 2022).

[60] C. R. Harris, K. J. Millman, S. J. van der Walt, and Numpy Project Group, "Array programming with NumPy," *Nature 2020 585:7825*, vol. 585, no. 7825, pp. 357–362, Sep. 2020, doi: 10.1038/s41586-020-2649-2.

[61] J. D. Hunter, "Matplotlib: A 2D graphics environment," *Computing in Science and Engineering*, vol. 9, no. 3, pp. 90–95, 2007, doi: 10.1109/MCSE.2007.55.

[62] Plotly, "Collaborative data science." Plotly Technologies Inc., Montreal, QC, 2015. [Online]. Available: https://plot.ly

[63] P. Domingos, "A few useful things to know about machine learning," *Communications of the ACM*, vol. 55, no. 10, pp. 78–87, 2012, doi: 10.1145/2347736.2347755.

[64] InvenSense, "MPU-9250 Product Specification," 2016. https://invensense.tdk.com/wp-content/uploads/2015/02/PS-MPU-9250A-01-v1.1.pdf (accessed Jun. 13, 2022).

[65] W. Torge and J. Müller, *Geodesy*, 4th ed. De Gruyter, 2012. doi: doi:10.1515/9783110250008.

[66] I. D. Dinov, *Data science and predictive analytics: Biomedical and health applications using R*. Switzerland: Springer International Publishing, 2018. doi: 10.1007/978-3-319-72347-1.

[67] X. Xiong, M. Wang, Y. Cai, L. Chen, H. Farah, and M. Hagenzieker, "A forward collision avoidance algorithm based on driver braking behavior," *Accident Analysis and Prevention*, vol. 129, no. January, pp. 30–43, 2019, doi: 10.1016/j.aap.2019.05.004.

[68] 501c3 NumFOCUS Foundation, "Project Jupyter | Home," 2022. https://jupyter.org/ (accessed Apr. 19, 2022).

[69] Pykalman, "pykalman — pykalman 0.9.2 documentation." https://pykalman.github.io/ (accessed Apr. 21, 2022).

[70] J. Hunt, *A Beginners Guide to Python 3 Programming*. Switzerland, 2020. doi: 10.1007/978-3-030-20290-3.

[71] Scikit-learn Developers, "Tuning the hyper-parameters of an estimator," *2022*. https://scikit-learn.org/stable/modules/grid_search.html (accessed Apr. 25, 2022).

[72] Joblib Developers, "Joblib: running Python functions as pipeline jobs — joblib 1.2.0.

Documentation," 2021. https://joblib.readthedocs.io/en/latest/index.html (accessed Apr. 27, 2022).

[73] Steer Davies Gleave, "Plan de Movilidad para el municipio de Popayán - Diagnostico Parte I," Colombia, 2015.

[74] Steer Davies Gleave, "Plan de Movilidad para el municipio de Popayán - Diagnostico Parte II," Colombia, 2015.

[75] A. M. Amiri, N. Nadimi, and A. Yousefian, "Comparing the efficiency of different computation intelligence techniques in predicting accident frequency," *IATSS Research*, 2020, doi: 10.1016/j.iatssr.2020.03.003.

Anexo A. Artículo de revisión

El anexo A presenta el artículo científico desarrollado durante la elaboración del presente trabajo de grado, el artículo fue presentado en la revista "*Journal of Traffic and Transportation Engineering (English Edition)".*

Disponible en el siguiente enlace:

https://drive.google.com/file/d/1iNiZBTmzZpsMPo07jGFhph97AboK3T2x/view?usp=sharing

Anexo B. Revisión sistemática de la literatura mediante la metodología PRISMA

El anexo B presenta la revisión sistemática de la literatura del presente documento de trabajo de grado, La revisión fue realizada siguiendo las pautas y lineamientos de la metodología PRISMA.

Disponible en el siguiente enlace:

https://drive.google.com/file/d/1J4hJukWtxlDZXluBknO81cQ1kAm8W2u0/view?usp=sharing

Anexo C. Revisión de arquitecturas

El anexo C presenta la revisión y análisis simple de las arquitecturas establecidas en otros estudios o investigaciones, estas arquitecturas presentan sistemas similares al propuesto en este documento.

Disponible en el siguiente enlace:

https://drive.google.com/file/d/1qbwvFtv1CUyEUUmnImHa0xHChVz0HLlf/view?usp=sharing

Anexo D. Desarrollo del prototipo del sistema mediante SCRUM

El anexo D presenta los documentos de inicio y planeación de la metodología SCRUM, en estos se describen cada una de las épicas, historias de usuario y tareas realizadas para el desarrollo prototipo del sistema.

Disponibles en el siguiente enlace:

https://drive.google.com/drive/folders/177KgUYNVKZkXz7o-bpp74hosU8qBKrN9?usp=sharing

Anexo E. Desarrollo del prototipo del SDIRC

El anexo E presenta el documento que describe el diseño y desarrollo del prototipo del sistema de detección inteligente de riesgos de colisión (SDIRC).

Disponible en el siguiente enlace:

https://drive.google.com/file/d/1EN0HgtOoGZ9h1KdWYKNheDpcIQirWKUG/view?usp=sharing

Anexo F. Aplicación móvil para recolección de datos

El anexo F presenta el código de la aplicación móvil para la recolección de datos, este se encuentra alojado en un repositorio de GitHub.

Disponible en el siguiente enlace: https://github.com/jjuan97/VehicleDataCollectionS

Anexo G. Aplicación para la recolección de datos en el dispositivo híbrido

El anexo G presenta el código de la aplicación para la recolección de datos en el dispositivo híbrido, este se encuentra alojado en un repositorio de GitHub.

Disponible en el siguiente enlace: https://github.com/jjuan97/VehicleDataCollectionRpi

Anexo H. Plataforma ICRDS web

El anexo H presenta los códigos usados en el desarrollo de la plataforma ICRDS web, estos se encuentran alojados en un repositorio de GitHub.

Disponible en el siguiente enlace: https://github.com/santiagoYps/ICRDSWeb

Anexo I. Data sets de maniobras de conducción

El anexo I presenta los "*data sets*", en formato CSV, de las maniobras de conducción usados como base para el desarrollo del modelo de ML.

Disponible en el siguiente enlace:

https://drive.google.com/drive/folders/1Te8cFEf3ThkUODlwjmD6rO3EZMWYgsJm?usp=sharing

Anexo J. Rendimiento de los algoritmos

El anexo J presenta el documento en formato Excel del rendimiento de los algoritmos al ejecutar la optimización de los hiper-parámetros y la validación cruzada.

Disponible en el siguiente enlace:

https://docs.google.com/spreadsheets/d/17VEtViejqvwQv8uTD54wiLxd8xOaUbV4/edit?usp=sharing&ouid=103071332065879887829&rtpof=true&sd=true

Anexo K. Revisión y diseño de las pruebas de campo de ND

El anexo K presenta el documento de revisión de las pruebas de conducción naturalista de los documentos del estado del arte. Además de la descripción de las cinco rutas diseñadas para las pruebas de campo.

Disponible en el siguiente enlace:

https://drive.google.com/file/d/1DN4qeoU9lSYC3ZnIJk4gEhBFpIIrBNls/view?usp=sharing

Anexo L. Data sets de ND

El anexo L presenta los “*data sets*”, en formato CSV, del desarrollo de las pruebas de conducción naturalista en la ciudad de Popayán. Además, es incluido, en formato JSON, los lugares donde el algoritmo clasificó la ocurrencia de un *near-crash*.

Disponible en el siguiente enlace:

https://drive.google.com/drive/folders/1_q_mdpp9Nb06cjpEIGP8RCWTPV3EWbIY?usp=sharing