**Pontificia Universidad Católica del Ecuador Facultad De Ingeniería**

**TEMA:**

# Estudio comparativo de la precisión de algoritmos de aprendizaje automático, regresión logística, máquinas de soporte y clasificador bayesiano, basado en la implementación de modelo predictivos en función de la mortalidad en accidentes en Ecuador

**AUTOR:**

Stalin Sebastian Salgado Escobar

**TUTOR:**

Msc. Edison Vicente Mora Londoño

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

Quito, junio 2023

## DEDICATORIA

Este trabajo va dedicado a mis padres Gloria y Genaro, quienes han velado por mí, por mi bienestar y educación entregando siempre mucho amor y paciencia además de su lucha constante y perseverante que siempre han sido un gran ejemplo por seguir.

A mi hermano Christian quien ha sido siempre antes que mi hermano un amigo que siempre que he necesitado siempre ha estado ahí para mí.

## AGRADECIMIENTO

Agradezco a Dios por darme la vida y salud y permitirme culminar un objetivo en mi carrera profesional, a mis padres y hermano por su comprensión en las labores de mi educación superior.

Al Ms. Edison Mora, por su conocimiento y apoyo incondicional en el desarrollo de este proyecto.

A la Carrera de Ingeniería y a los respectivos ingenieros por el conocimiento compartido durante estos años que he permanecido en la Pontificia Universidad Católica del Ecuador.

## RESUMEN

El estudio actual se enfoca en un estudio comparativo de tres modelos predictivos que permita establecer el más preciso en el escenario planteado, construidos mediante el uso de técnicas de minería de datos, basado en el estudio de la mortalidad en accidentes de tráfico en Ecuador.

El país se enfrenta una tasa de mortalidad en accidentes de tráfico muy alta, transformándose en uno de los mayores problemas dentro del país y reducir es uno de los objetivos prioritarios para la agencia nacional de tránsito del país, dicho estudio se justifica en función de la necesidad de desarrollar herramientas más efectivas para reducir la tasa de mortalidad en accidentes de tránsito, los accidentes de tráfico son una causa importante de mortalidad, y es necesario adoptar nuevas estrategias para abordar este problema de manera más efectiva. El uso de técnicas de minería de datos y modelos predictivos puede ser una forma efectiva de analizar y predecir la mortalidad en accidentes de tráfico.

Varios de los ítems identificados en los accidentes de tráfico en el Ecuador se basan en detalles como infraestructura vial deficiente, falta de cultura vial, falta de educación vial, pésima señalización en algunos escenarios. Estos factores aumentan el riesgo de accidentes de tránsito, y por consiguiente el riesgo de pérdidas humanas.

El objetivo principal de este estudio es desarrollar un modelo predictivo que permita predecir la mortalidad en accidentes de tránsito en Ecuador mediante el uso de técnicas de minería de datos y modelos predictivos, donde por medio de recopilar los datos históricos, selección de variables relevantes, implementación de técnicas de aprendizaje automático supervisado y evaluar la capacidad predictiva del modelo propuesto podamos entregar una

herramienta que permita tomar decisiones más informadas y efectivas en la prevención de accidentes de tráfico y la reducción de la mortalidad.

Los datos históricos de accidentes de tráfico en Ecuador se recopilarán de registro de accidentes publicado por agencia nacional de tránsito (ANT), entre los años 2017 y 2022.

# INDICE DE CONTENIDO

# INDICE DE FIGURAS

# INDICE DE TABLAS

# 1. Capítulo I. Introducción

## 1.1 Generalidades

Según el informe de la Organización Mundial de la Salud (OMS) de 2015 sobre el estado de la seguridad vial en el mundo, las muertes en carretera en los países de ingresos bajos y medianos son el doble que en los países de ingresos altos (1,9 millones para 2020). A pesar de concentrar solo el 54% de los vehículos del mundo, el 90% de las muertes por accidentes de tránsito ocurren en países de bajos y medianos ingresos. En la región de las Américas, la tasa de fatalidad por accidentes de tránsito en 2013 fue de 15,9 por cada 100 000 habitantes, aún por debajo del nivel mundial. Sin embargo, esta tasa varía según la subregión. Por ejemplo, la región andina tuvo la tasa más alta de este tipo de incidentes: 23,4 por cada 100.000 habitantes. En el caso de Ecuador, entre 1998 y 2015 hubo 373.265 accidentes de tránsito en los que fallecieron alrededor de 29.000 personas y resultaron heridas aproximadamente 244.000, algunas de las cuales quedaron incapacitadas permanentemente. Cabe mencionar que hubo una tendencia al alza durante este período, ya que el valor más alto se registró en 2014. En este sentido, con el fin de garantizar la seguridad vial de los participantes de la vía desde un enfoque preventivo a través de la educación y la concientización, Ecuador reformará la Ley Orgánica de Transporte Terrestre, Tránsito y Seguridad Vial en el año 2014. (Campos-Villalta, 2019)

Durante la Semana de la Seguridad Vial, la Organización Mundial de la Salud (OMS) presentó su segundo plan global de acción a 10 años con el objetivo de reducir las muertes y lesiones graves relacionadas con el tránsito en un 50 % para 2030. En su resolución 74/299, la Asamblea General de las Naciones Unidas proclamó una Década de Acción para la Seguridad Vial 2021-2030, que tiene en cuenta cerca de 1,3 millones de muertes prevenibles y alrededor de

50 millones de lesiones anales causadas por accidentes de tránsito. Son la primera causa de muerte entre niños y adolescentes en el mundo. (Salud, 2021)

El plan global propuesto por la Organización Mundial de la Salud considera que estas cifras son inaceptables a nivel mundial y llama a los gobiernos y partes interesadas a tomar un nuevo camino, utilizando un enfoque de sistemas integrados de seguridad que posicione directamente a la seguridad vial como motor del desarrollo sostenible. Si las tendencias actuales continúan, se espera que los accidentes de tráfico maten a 13 millones de personas más y lesionen a 500 millones durante la próxima década, particularmente en países de ingresos bajos y medianos. (Salud, 2021)

Por lo tanto, el objetivo principal de la Segunda Década de Acción para la Seguridad Vial 2021-2030 es reducir las lesiones y muertes relacionadas con el tránsito en un 50% durante este período. En este sentido, el mundo planea negarse a seguir operando como de costumbre para poner la seguridad en el centro del trabajo y garantizar que viajar seguro sea automáticamente un derecho humano. (Salud, 2021)

El Plan Global establece medidas para lograr los objetivos y colabora en el desarrollo de planes de acción y el establecimiento de objetivos nacionales y locales para la Década de Acción. Incluye medidas concretas, cómo se implementarán y con quién, desde los gobiernos hasta el sector privado, la sociedad civil, las entidades de financiación y las agencias de la ONU. (Salud, 2021)

A su vez, plantea la necesidad de asegurar que las perspectivas de género sean consideradas en la planificación del transporte. Si bien las mujeres tienen muchas menos probabilidades que los hombres de morir en un accidente automovilístico, tienen un 47 por

ciento más de probabilidades de sufrir lesiones graves en un accidente automovilístico y cinco veces más de sufrir latigazos. (Salud, 2021)

El exceso de velocidad, la conducción somnolienta, la falta de uso de cinturones de seguridad, cascos o sistemas de sujeción para niños, y la conducción bajo los efectos del alcohol son los principales factores que contribuyen a las lesiones y muertes causadas por el tránsito. De hecho, en Uruguay en marzo de este año, la Organización Panamericana de la Salud y la Fundación Gonzalo Rodríguez realizaron un estudio sobre el impacto del alcohol en la siniestralidad con base en publicaciones nacionales e internacionales y concluyeron que existen regulaciones más restrictivas al consumo de alcohol países con mayores tasas de mortalidad de menores de edad. (Salud, 2021)

Por lo tanto, existe la necesidad de una encuesta para desarrollar modelos predictivos para estudiar la mortalidad en accidentes de tránsito en el Ecuador. Tal modelo podría contribuir significativamente a identificar los factores que contribuyen a las muertes por accidentes de tránsito en el país, ayudando así a desarrollar políticas y medidas de seguridad vial más efectivas y salvando vidas.

## 1.2 Planteamiento del Problema

Según el INEC, y la Agencia Nacional de Tránsito de Ecuador, en el año 2020 hubo 44.242 accidentes de tránsito en el país, un aumento del 1,3 % con respecto al año anterior. Estos accidentes se saldaron con un total de 2.780 víctimas mortales, lo que supone 16,8 muertos por cada 100.000 habitantes. Las cifras son preocupantes y apuntan a la necesidad de desarrollar medidas preventivas para reducir el número de accidentes y muertes en el país. (INEC, 2022)

La investigación "Marco de predicción de la gravedad de los accidentes basado en el aprendizaje profundo". ha demostrado que la minería de datos y los modelos predictivos pueden ser herramientas poderosas que le permiten analizar grandes cantidades de datos y encontrar patrones y relaciones que no son visibles a simple vista, lo que le permite identificar los riesgos asociados con las enfermedades viales y crear medidas de prevención efectivas. estos eventos de desastre. (Rahim, 2021).

El desarrollo de un modelo de predicción de mortalidad de tránsito en Ecuador permitirá tomar decisiones de salud pública y seguridad vial, identificar los riesgos que provocan las fatalidades de tránsito y tomar medidas preventivas para reducir el número de fatalidades, accidentes y enfermedades viales en el país. Sin embargo, cabe decir que este proyecto puede contribuir a la investigación en el campo de la minería de datos y la modelización predictiva, ya que permitirá estudiar los métodos y métodos de análisis y predicción de muertes en carretera.

## 1.3 Objetivos

### *1.3.1 Objetivos General*

Desarrollar un estudio comparativo que permita identificar el modelo más preciso usando regresión logística, máquinas de soporte vectorial y clasificador bayesiano basado en el análisis de la mortalidad en accidentes de tráfico en Ecuador.

### *1.3.2 Objetivos Específicos*

- Recopilar datos históricos de accidentes de tráfico en Ecuador
- Evaluar y seleccionar las variables más relevantes que influyen en la mortalidad en accidentes de tráfico en Ecuador mediante técnicas de minería de datos

- Diseñar e implementar modelos de aprendizaje automático con regresión logística, máquinas de soporte vectorial y clasificador bayesianos que permitan predecir la mortalidad en accidentes de tráfico en Ecuador

- Analizar los resultados obtenidos y mostrar cual es el modelo más indicado para la implementación de un estudio como este.

## 1.4 Alcance

Para este proyecto se va a emplear datos públicos entregados por la agencia nacional de tránsito, los mismos están disponibles entre los años 2017 al 2022, donde podemos encontrar el histórico de accidentes en el Ecuador. En función de estos datos lo que se pretende es aplicar la metodología CRISP-DM que permita realizar todas las fases necesarias, tales como:

- Entendimiento del Negocio
- Comprensión de la data
- Preparación de la data
- Modelado
- Evaluación
- Despliegue

Para realizar los ítems anteriormente listados, se usará Python como lenguaje de programación, y sus principales librerías Pandas, Sklearn, Scipy, Numpy, Seaborn Matplotlib , entre otras.

Partiendo de la implementación antes indicada se generará tres modelos con la misma data ya procesada para realizar una comparativa que permita obtener conclusiones sobre el

comportamiento en función de los accidentes de tránsito y adicionalmente determinar qué modelo o algoritmo es el más preciso.

# 2. Capítulo II. Revisión literaria

## 2.1 Minería de Datos

Los seres humanos han estado recopilando y analizando datos durante miles de años y, en muchos sentidos, estos métodos han cambiado a medida que han madurado las computadoras y los datos, así como las herramientas para administrar y analizar datos, como también el desarrollo de tecnología de base de datos y consultas de lenguaje natural orientadas al usuario, como SQL Query Language, que ayuda a los usuarios comerciales a analizar sus datos con esta extensión. (SAP, 2023)

El descubrimiento de conocimiento en bases de datos (KDD, por sus siglas en inglés), es el proceso de encontrar patrones importantes, patrones y relaciones ocultas dentro de grandes conjuntos de datos utilizando métodos estadísticos y técnicas de aprendizaje automático. (Han, 2012)

El objetivo principal de la minería de datos es extraer información valiosa y procesable de datos grandes y complejos que, de otro modo, serían difíciles de analizar. Esta disciplina es ampliamente utilizada en diversas industrias, como la banca, el comercio minorista, la atención médica, la seguridad y el marketing en Internet. (Han, 2012)

La minería de datos utiliza una variedad de métodos, que incluyen clasificación, desconvolución, agrupación y agrupación, búsqueda de patrones y relaciones en los datos. Una vez que se definen estas relaciones, se pueden utilizar para tomar decisiones informadas y mejorar los procesos comerciales. (Han, 2012)

### *2.1.1 Técnicas de minería de datos*

Con el objetivo de descubrir el conocimiento dentro de la información almacenada en grandes bases de datos se utilizan técnicas de minería de datos para este estudio nos vamos a centrar en aprendizaje supervisado

El aprendizaje supervisado, se refiere a un conjunto de técnicas de minería de datos en las que se proporciona un conjunto de datos etiquetados (datos con respuestas conocidas) a un algoritmo para que aprenda a predecir la respuesta correcta en nuevos datos. El objetivo principal del aprendizaje supervisado es crear un modelo que pueda hacer predicciones precisas sobre datos futuros no etiquetados. (Han, 2012)

### *2.1.2 Técnicas supervisadas*

**2.1.2.1 Clasificación.** Se utiliza para predecir la clase de un objeto a partir de un conjunto de datos etiquetados. Los algoritmos de clasificación más comunes son el Árbol de Decisiones, el Naïve Bayes y las Máquinas de Soporte Vectorial (SVM).

**2.1.2.2 Regresión.** Se utiliza para predecir un valor numérico continúo basado en los valores de otras variables. Los algoritmos de regresión más comunes son la Regresión Lineal y la Regresión Logística.

#### 2.1.2.3 Algoritmos de Clasificación

***2.1.2.3.1 Clasificador Bayesiano.*** Es un algoritmo de aprendizaje automático que utiliza la teoría de probabilidad de Bayes para predecir la clase de un objeto basado en sus características. El clasificador bayesiano se basa en la probabilidad condicional de que una instancia pertenezca a una clase dada, dadas sus características (Alpaydin, 2020).

El clasificador bayesiano se divide en dos categorías principales: Naïve Bayes y Red Bayesiana. El Naïve Bayes es un método simple pero efectivo que asume la independencia, lo que significa que la presencia o ausencia de algo no está relacionada con la presencia o ausencia de otras cosas. Por otro lado, las redes bayesianas modelan las interacciones entre estructuras, lo que permite modelar los datos de forma compleja. (Alpaydin, 2020)

El clasificador bayesiano se utiliza comúnmente en la clasificación de textos, detección de spam, diagnóstico médico y otras aplicaciones donde se necesitan predicciones precisas basadas en datos. Una de las principales ventajas del clasificador bayesiano es su capacidad para manejar datos con alta dimensionalidad y multicolinealidad. (Alpaydin, 2020)

***2.1.2.3.2 Máquinas de Soporte Vectorial.*** Son un tipo de algoritmo de aprendizaje supervisado utilizado para resolver problemas de clasificación y regresión. SVM es una técnica de aprendizaje basada en el concepto de maximización de la separación entre dos clases de datos. En otras palabras, SVM intenta encontrar el hiperplano que mejor separa los datos de diferentes clases. (Bishop, 2006)

En el caso de la clasificación, SVM busca encontrar el hiperplano que mejor separa los datos de diferentes clases en el espacio de características. En el caso de la regresión, SVM busca encontrar una función que pueda predecir el valor de una variable de salida a partir de las variables de entrada. (Bishop, 2006)

SVM es útil en la clasificación de datos no lineales mediante el uso de una función de kernel para transformar los datos de entrada en un espacio dimensional superior. SVM también puede manejar datos con dimensiones muy altas y reducir la posibilidad de sobreajuste. (Bishop, 2006)

Sin embargo, SVM también tiene algunas limitaciones. La selección adecuada de un kernel puede ser difícil y SVM puede ser sensible a los datos atípicos y a los valores atípicos en el conjunto de datos. (Bishop, 2006)

En general, SVM es una técnica poderosa y ampliamente utilizada en el campo de la ciencia de datos para la clasificación y regresión de datos complejos. (Bishop, 2006)

#### 2.1.2.4 Algoritmo de Regresión

***2.1.2.4.1 Regresión Logística.*** Es una técnica de análisis estadístico utilizada para modelar la relación entre una variable de resultado binaria y múltiples variables predictoras. La regresión logística es una extensión de la regresión lineal, que se utiliza para modelar la relación entre una variable de resultado continua y una o varias variables predictoras. (Hosmer Jr, 2013)

En la regresión logística, la variable de resultado es una variable binaria, es decir, toma valores de 0 o 1. La función logística transforma la variable dependiente en una probabilidad, que representa la probabilidad de que la variable de resultado tome el valor de 1 dado los valores de las variables predictoras. (Hosmer Jr, 2013)

La regresión logística se utiliza comúnmente en la investigación médica, biológica y social para predecir la probabilidad de que una persona tenga una determinada enfermedad o comportamiento, en función de varios factores de riesgo. (Hosmer Jr, 2013)

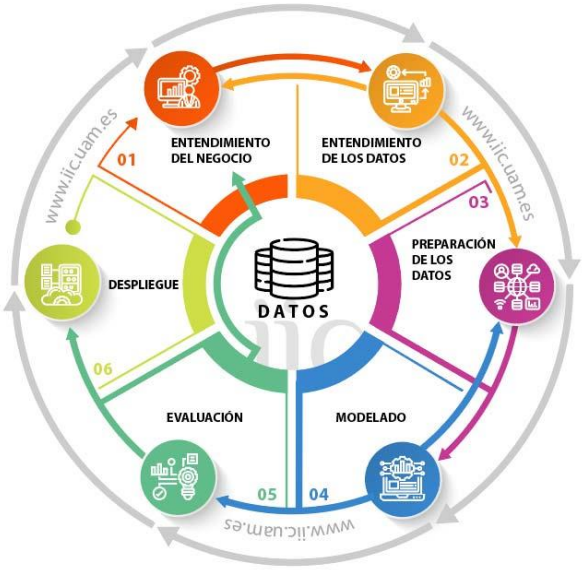
La regresión logística se puede realizar utilizando software estadístico especializado, como Python R o SPSS. Los coeficientes de la regresión logística pueden interpretarse para determinar la dirección y la fuerza de la relación entre las variables predictoras y la variable de resultado. (Hosmer Jr, 2013)

## 2.2 Metodología de Minería de Datos

### *2.2.1 CRISP-DM*

El método CRISP-DM se define como un conjunto de pasos o pasos a los que muchos expertos del sector privado se refieren para resolver problemas de minería de datos. Estas medidas han sido desarrolladas a partir de su experiencia y de los métodos de medición más populares. Dado que este método se utiliza principalmente en empresas privadas, su crecimiento y popularidad en el campo de la minería de datos se muestra en la Figura 1. (Galán Cortina, 2016)

**Figura 1 Proceso CRISP-DM**




*Fuente: Wikipedia: https://www.iic.uam.es/innovacion/metodologia-crisp-dm-ciencia-de-datos/Archivo: CRISP-DM.jpg (Creative Commons)*

CRISP-DM incluye una plantilla e instrucciones paso a paso. Otra ventaja es que no está organizado, sino que puedes ir de un paso a otro para poder ver cualquier parte seleccionada y volver al paso en el que estábamos antes, lo que nos permite trabajar de forma similar al método SCRUM para mejorar el proyecto, en la Figura 1 podemos ver todos los componentes de CRISP-DM, donde se puede buscar la secuencia entre ellos. (Galán Cortina, 2016)

### *2.2.2 Metodologías y Técnicas*

CRISP-DM es un proceso de 6 fases que dispone en diferentes fases, por lo tanto, no se considera un proceso rígido

**2.2.2.1 Comprensión del Negocio.** En esta fase, se identifica el problema empresarial a resolver y se establecen los planes del proyecto. Además, se recopila información y se hace un análisis preliminar para comprender los hechos y las posibles relaciones entre las variables. (Wirth, 2000)

**2.2.2.2 Comprensión de la Data.** En esta etapa, los datos se analizan en detalle y se seleccionan los relevantes para el análisis. Las técnicas de análisis estadístico se utilizan para analizar datos e identificar patrones y relaciones. (Wirth, 2000)

**2.2.2.3 Preparación de la Data.** En esta fase, se preparan los datos para el análisis mediante la limpieza, la transformación y la selección de variables. Esta fase también puede incluir la creación de nuevas variables y la combinación de múltiples conjuntos de datos. (Wirth, 2000)

**2.2.2.4 Modelado.** En esta fase, se construyen modelos predictivos utilizando técnicas de modelado estadístico o de aprendizaje automático. Se seleccionan los modelos más apropiados para el problema en cuestión y se ajustan sus parámetros para optimizar su rendimiento. (Wirth, 2000)

**2.2.2.5 Evaluación.** En esta fase, se evalúan los modelos construidos en términos de su capacidad para predecir el comportamiento futuro de los datos. Se utilizan métricas de evaluación para comparar los modelos y seleccionar el mejor modelo para su uso en la toma de decisiones. (Wirth, 2000)

**2.2.2.6 Despliegue.** En esta fase, se implementa el modelo seleccionado en la práctica y se monitoriza su rendimiento para asegurarse de que sigue siendo efectivo. (Wirth, 2000)

La metodología CRISP-DM es ampliamente utilizada en la industria y en la academia para guiar el proceso de minería de datos y asegurar que se sigan las mejores prácticas. Además, la metodología es flexible y se puede adaptar a diferentes situaciones y problemas de negocio. (Wirth, 2000)

# 3. Capítulo III. Marco metodológico

## 3.1 Herramientas

### *3.1.1 Lenguaje de Programación*

**3.1.1.1 Python.** Es uno de los lenguajes de programación de ciencia de datos más populares debido a su sintaxis clara y concisa, las muchas herramientas y bibliotecas de análisis de datos disponibles y su fácil integración con la tecnología, algunas en ciencia de datos. Las bibliotecas de Python comúnmente utilizadas para la ciencia de datos incluyen Pandas para ciencia de datos, NumPy para computación numérica, Matplotlib y Seaborn para visualización de datos y Scikit-Learn para aprendizaje automático. (VanderPlas, 2016)

Además, Python se usa ampliamente en el análisis de big data porque se puede integrar fácilmente con herramientas de big data como Apache Hadoop y Spark. Python también se usa para desarrollar aplicaciones web para visualización de datos y análisis de datos en tiempo real. (VanderPlas, 2016)

Python es una herramienta de ciencia de datos poderosa y flexible debido a su capacidad para manejar grandes conjuntos de datos, la capacidad de integrarse con otras tecnologías de procesamiento de datos y los tipos de muchas bibliotecas y herramientas para el análisis de datos. (VanderPlas, 2016)

### *3.1.2 Herramientas para uso de Python*

**3.1.2.1 Jupyter Notebook.** Es una herramienta web de código abierto que le permite crear y compartir documentos interactivos con código, elementos visuales y texto. Jupyter Notebook es una herramienta de ciencia de datos popular debido a su capacidad para crear y ejecutar código en tiempo real y su capacidad para combinar texto e imágenes en un solo documento. (Kluyver, 2016)

Jupyter Notebook admite muchos lenguajes de programación, incluidos Python, R y Julia, lo que lo convierte en una herramienta versátil para diferentes proyectos y necesidades. Además, Jupyter Notebook se puede integrar fácilmente con otras herramientas de procesamiento y bibliotecas como Pandas, NumPy y Matplotlib. (Kluyver, 2016)

Jupyter Notebook también se usa para enseñar y aprender sobre ciencia de datos, ya que permite a los estudiantes interactuar con el código y los datos en tiempo real y ver los resultados de una manera clara y fácil de entender. (Kluyver, 2016)

Jupyter Notebook es una herramienta esencial para la ciencia de datos debido a su capacidad para crear y ejecutar código en tiempo real, combinar documentos y visualizaciones y adaptarse a diferentes proyectos y necesidades. (Kluyver, 2016)

**3.1.2.2 Anaconda.** Es una distribución de código abierto que le permite ejecutar paquetes y crear estaciones de trabajo para el procesamiento de datos. Anaconda incluye muchas herramientas y bibliotecas de ciencia de datos populares, como Python, R y Jupyter Notebook, así como paquetes especializados como NumPy, Pandas, Matplotlib y SciPy. (Anaconda, 2021)

Anaconda se usa ampliamente en la comunidad de ciencia de datos debido a su facilidad de instalación y uso, así como a la capacidad de manejar dependencias. La plataforma Anaconda también es compatible con muchos sistemas operativos, lo que la convierte en una herramienta flexible para diferentes proyectos y necesidades. (Anaconda, 2021)

Además, Anaconda proporciona una interfaz gráfica de usuario (GUI) llamada Anaconda Navigator que permite a los usuarios encontrar y administrar fácilmente los paquetes y procesos instalados. (Anaconda, 2021)

### *3.1.3 Librerías*

**3.1.3.1 Scikit-learn.** Es una biblioteca de aprendizaje automático de código abierto para Python, proporciona una amplia variedad de herramientas y algoritmos para el aprendizaje supervisado y no supervisado, la selección de funciones, la validación de modelos y la evaluación del rendimiento. Scikit-Learn también proporciona herramientas de preprocesamiento de datos, como normalización y reducción de tamaño. (Pedregosa, 2011)

Scikit-learn se usa ampliamente en la comunidad de ciencia de datos debido a su fácil integración con otras bibliotecas de Python como NumPy y Pandas. Además, su documentación detallada y numerosos ejemplos de uso facilitan el aprendizaje y el uso tanto para principiantes como para usuarios avanzados. (Pedregosa, 2011)

Scikit-Learn es una biblioteca de aprendizaje automático esencial para la ciencia de datos, que ofrece una variedad de herramientas y soluciones para construir y probar modelos de aprendizaje automático. (Pedregosa, 2011)

**3.1.3.2 sklearn.model_selection:** proporciona una función útil para segmentar datos en conjuntos de entrenamiento y prueba en el aprendizaje automático. La función train_test_split se utiliza para dividir la recopilación de datos en pruebas aleatorias y subpruebas. La función asume un conjunto de datos de entrada, una variable de selección y un tamaño de caso de prueba. (Pedregosa, 2011)

La función train_test_split es muy útil para probar si el modelo se ajusta bien a los datos y para evaluar el rendimiento del modelo. Al dividir los datos en un conjunto de entrenamiento y uno de prueba, podemos entrenar un modelo en el conjunto de entrenamiento y comprobar su rendimiento en el conjunto de prueba. (Pedregosa, 2011)

La sintaxis básica de la función es:

```
from sklearn.model_selection import train_test_split

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2)
```

Donde X es la matriz de características y ‘y’ es la variable objetivo.

La función train_test_split es una de las funciones más utilizadas en la biblioteca de aprendizaje automático de Python y es una herramienta esencial para cualquier científico de datos o desarrollador de aprendizaje automático. (Pedregosa, 2011)

La librería scikit-learn en Python ofrece una variedad de algoritmos de aprendizaje automático, incluyendo el clasificador de Naive Bayes. Este clasificador se encuentra en el módulo sklearn.naive_bayes como la clase GaussianNB. (developers, 2021)

La implementación de este grupo se basa en la teoría de Bayes, que asume que los procesos del evento son independientes entre sí y que todos afectan el resultado final de manera independiente. Esta implementación funciona mejor con datos discretos, ya que maneja la representación de los datos de forma transparente y puede minimizar la probabilidad de clúster. (developers, 2021)

La librería scikit-learn de Python proporciona varias funciones de evaluación de modelos de aprendizaje automático que pueden ayudar a los científicos de datos a entender mejor el rendimiento de sus modelos. Algunas de estas funciones incluyen accuracy_score, precision_score, recall_score, classification_report, roc_curve, auc, y confusion_matrix. (developers, 2021)

La función accuracy_score calcula la precisión del modelo al comparar las etiquetas verdaderas con las predicciones del modelo. La precisión mide la fracción de etiquetas verdaderas que fueron clasificadas correctamente. La función precision_score calcula la proporción de etiquetas positivas verdaderas sobre todas las etiquetas clasificadas positivamente. La función recall_score mide la proporción de etiquetas positivas verdaderas sobre todas las etiquetas positivas verdaderas y falsas negativas. (developers, 2021)

La función classification_report muestra un informe que contiene el valor de precisión, recall y f1-score para cada clase. La curva ROC se utiliza para evaluar el rendimiento del clasificador a diferentes niveles de umbral de clasificación. La curva se traza al representar la tasa de verdaderos positivos frente a la tasa de falsos positivos. La función auc devuelve el área bajo la curva ROC. (developers, 2021)

Finalmente, la función confusion_matrix se utiliza para evaluar el rendimiento de un modelo clasificador en términos de la matriz de confusión, que es una tabla que muestra la cantidad de verdaderos positivos, falsos positivos, verdaderos negativos y falsos negativos. (developers, 2021)

**3.1.3.3 GaussianNB.** Tiene varios parámetros ajustables, como el suavizado y la prioridad de clase. Además, también incluye métodos para ajustar el modelo a los datos de entrenamiento y para realizar predicciones en nuevos datos. (developers, 2021)

**3.1.3.4 El clasificador Naive Bayes.** Se ha utilizado con éxito en problemas de clasificación de texto, detección de spam y filtrado de correo electrónico, entre otros. (developers, 2021)

La librería sklearn.utils en Python proporciona varias herramientas útiles para la manipulación de datos y preprocesamiento en ciencia de datos. Una de estas herramientas es la función resample que permite realizar re-muestreo de datos para equilibrar clases desequilibradas. (Pedregosa, 2011)

Cuando se trabaja con un conjunto de datos desigual donde un grupo tiene más ejemplos que el otro, puede ser difícil obtener buenos resultados en los modelos de aprendizaje automático. El remuestreo es una forma común de resolver este problema, ya que puede comparar grupos asignando más muestras al grupo minoritario o eliminando muestras del grupo mayoritario. (Pedregosa, 2011)

La función resample puede realizar dos tipos de re-muestreo: sub-muestreo y sobre-muestreo. El sub-muestreo reduce el número de ejemplos de la clase mayoritaria, mientras que el sobre-muestreo aumenta el número de ejemplos de la clase minoritaria. La función resample también proporciona opciones para controlar el equilibrio de clase deseado, como el número de ejemplos de la clase minoritaria y la estrategia de re-muestreo. (Pedregosa, 2011)

El uso de resample en conjunto con otras herramientas de preprocesamiento de datos y técnicas de modelado de aprendizaje automático puede ayudar a mejorar el rendimiento del modelo en conjuntos de datos desequilibrados. (Pedregosa, 2011)

La librería scikit-learn incluye la implementación de máquinas de soporte vectorial (SVM) a través del módulo sklearn.svm. Las SVM son un tipo de modelo de aprendizaje supervisado que se utiliza tanto para la clasificación como para la regresión. Su objetivo principal es encontrar un hiperplano en un espacio de alta dimensión que maximice la separación entre las clases en un conjunto de datos de entrenamiento. (Pedregosa, 2011)

**3.1.3.5 SVM.** Es SVC (Support Vector Classification) esta implementación puede manejar tanto problemas de clasificación binaria como multiclase, y puede utilizar diferentes kernels para modelar relaciones no lineales en los datos. Los kernels disponibles incluyen lineal, polinómico, radial basis function (RBF) y sigmoidal. Además, SVC también admite la clasificación multietiqueta y la clasificación probabilística. (Pedregosa, 2011)

La librería SVC también ofrece diferentes parámetros que pueden ser ajustados para mejorar el rendimiento del modelo, como C (parámetro de regularización), gamma (parámetro de kernel RBF) y coef0 (parámetro de kernel polinómico y sigmoidal). Estos parámetros pueden ser ajustados utilizando técnicas de búsqueda de hiperparámetros, como Grid Search o Random Search. (Pedregosa, 2011)

La implementación de SVM en la librería scikit-learn es una herramienta poderosa y flexible para resolver problemas de clasificación y regresión en ciencia de datos. Su capacidad para manejar datos no lineales y multiclase, junto con la facilidad de ajuste de los parámetros del modelo, lo hacen ideal para una amplia gama de aplicaciones en diferentes campos. (Pedregosa, 2011)

La librería scikit-learn proporciona una variedad de algoritmos de aprendizaje automático, incluyendo modelos de regresión lineal y logística. La regresión logística es una técnica de clasificación que se utiliza para predecir la probabilidad de un evento binario. La regresión logística utiliza la función sigmoide para transformar la salida del modelo en valores de probabilidad, y los coeficientes del modelo se ajustan para maximizar la verosimilitud de los datos. (Pedregosa, 2011)

**3.1.3.6 LogisticRegression.** Implementa la regresión logística y proporciona una variedad de opciones de regularización, incluyendo L1, L2 y elastic net. La regularización se utiliza para evitar el sobreajuste y mejorar la generalización del modelo. Además, la clase LogisticRegression permite la especificación de pesos de clase para abordar problemas de desequilibrio de clase. (Pedregosa, 2011)

La librería scikit-learn también proporciona diversas métricas de evaluación de modelos, incluyendo la precisión, el recall, la f1-score y la matriz de confusión. Estas métricas se pueden utilizar para evaluar el rendimiento del modelo y ajustar los parámetros del modelo para mejorar su rendimiento. También se puede utilizar la curva ROC para evaluar el rendimiento del modelo y seleccionar el umbral de clasificación óptimo. (Pedregosa, 2011)

**3.1.3.7 Pandas.** Es una biblioteca de Python utilizada en ciencia de datos para el análisis y manipulación de datos estructurados. Permite la carga y almacenamiento de datos en diferentes formatos, como CSV, Excel, SQL, entre otros, y su posterior procesamiento a través de funciones y métodos especializados para la manipulación de dataframes y series de tiempo. (McKinney, 2010)

Entre las características de pandas se incluyen la capacidad de filtrar y seleccionar datos, fusionar y unir conjuntos de datos, manipulación de índices y etiquetas, cálculo de estadísticas y agregaciones, entre otros. (McKinney, 2010)

Pandas es una herramienta esencial para la limpieza, transformación y análisis de datos en Python y es ampliamente utilizada en la industria y la academia. (McKinney, 2010)

**3.1.3.8 Matplotlib.** Es una biblioteca de Python utilizada en ciencia de datos para la creación de gráficos y visualizaciones. Permite la generación de una gran variedad de gráficos,

desde simples gráficos de línea y de barras hasta gráficos 3D y mapas de calor. Matplotlib es altamente personalizable y cuenta con una gran cantidad de opciones de configuración para los gráficos. (Hunter, 2007)

Entre las características de Matplotlib se incluyen la capacidad de visualizar datos en tiempo real, la integración con otras bibliotecas de Python como NumPy y Pandas, la creación de gráficos interactivos y la exportación de gráficos en diferentes formatos. (Hunter, 2007)

Matplotlib es una herramienta esencial para la visualización de datos en Python y es ampliamente utilizada en la industria y la academia. (Hunter, 2007)

**3.1.3.9 Scipy.** Es una biblioteca de Python utilizada en ciencia de datos para el procesamiento y análisis de datos científicos. Proporciona una amplia variedad de herramientas para el cálculo numérico, la optimización, el álgebra lineal, el procesamiento de señales, el análisis estadístico y más. (Virtanen, 2020)

Scipy es una de las bibliotecas más utilizadas en Python para el análisis científico y es altamente valorada por su capacidad para realizar operaciones matemáticas complejas y eficientes en grandes conjuntos de datos. (Virtanen, 2020)

Entre las características de Scipy se incluyen la capacidad de realizar interpolación de datos, integración numérica, transformada de Fourier, ajuste de curvas, resolución de ecuaciones diferenciales y más. (Virtanen, 2020)

Scipy también es compatible con otras bibliotecas de Python utilizadas en ciencia de datos, como Numpy y Matplotlib. (Virtanen, 2020)

La librería scipy.stats es una biblioteca de funciones estadísticas para Python. Contiene una amplia variedad de distribuciones de probabilidad, pruebas estadísticas y herramientas de ajuste de datos. En particular, dos funciones útiles son chi2_contingency. (Virtanen, 2020)

**3.1.3.9 Numpy.** Es una biblioteca de Python utilizada en ciencia de datos para el procesamiento de datos científicos y numéricos. Proporciona una amplia variedad de herramientas para la creación y manipulación de matrices y arrays, así como funciones matemáticas para el cálculo numérico y algebraico. (Harris, 2020)

La principal ventaja de Numpy es que permite realizar cálculos matemáticos y estadísticos eficientes en grandes conjuntos de datos, ya que utiliza una estructura de datos que permite una rápida indexación y cálculos vectorizados. Además, Numpy es compatible con otras bibliotecas de Python utilizadas en ciencia de datos, como Matplotlib y Pandas. (Harris, 2020)

Entre las características de Numpy se incluyen la capacidad de realizar operaciones matemáticas básicas y avanzadas, crear arrays multidimensionales, manipular arrays y matrices, realizar operaciones de álgebra lineal y más. (Harris, 2020)

**3.1.3.11 Seaborn.** Es una librería de visualización de datos en Python basada en Matplotlib, que proporciona una interfaz de alto nivel para la creación de gráficos estadísticos atractivos e informativos. Seaborn facilita la exploración y comprensión de los datos, permitiendo la creación de gráficos tales como histogramas, diagramas de dispersión, diagramas de violín, mapas de calor y más. Además, también ofrece la capacidad de personalizar fácilmente la apariencia de los gráficos y agregar elementos como etiquetas y títulos. (Waskom, 2021)

Una de las ventajas de Seaborn es su capacidad para visualizar relaciones complejas entre múltiples variables. Por ejemplo, la función pairplot permite mostrar la relación entre múltiples

pares de variables en una sola trama. Además, Seaborn también cuenta con herramientas para la visualización de datos categóricos, como gráficos de barras y gráficos de pastel. (Waskom, 2021)

La librería también es altamente personalizable y se integra bien con otras librerías de Python para análisis de datos, como Pandas y NumPy. Además, Seaborn también incluye conjuntos de datos de ejemplo para facilitar la experimentación con diferentes gráficos y configuraciones. (Waskom, 2021)

## 3.2 Métodos

### *3.2.1 Metodología*

El método utilizado para crear este programa es CRISP-DM, que crea un conjunto de pasos que permite una estructura específica, y también les da la capacidad de repetirse, dando la posibilidad de volver a la fase anterior, entendida como cíclica. proceso, hasta encontrar en este caso el ejemplo deseado.

**3.2.1.1 Comprensión del Negocio.** De acuerdo con este método, comenzará con un entendimiento comercial, es decir, cómo trabaja la Agencia Nacional de Tránsito con los datos, cómo los recopila, si usan un método que les permita manejar muchos datos. y, o si ha separado este control para ver la complejidad y riesgo que pueden tener los datos con los que trabajaremos.

**3.2.1.2 Comprensión de la Data.** A continuación, empezaríamos con la comprensión de los datos, para ellos debemos ya tener la información en algún formato ya sea por api rest, archivos csv o algún archivo plano, partiendo de esto usaremos un jupiter notebook para poder cargar los datos y poder visualizar la data en un escenario macro y verificar ítem como:

- Números de Registro

- Numero de columnas
    - Tipos de dato de cada una de las columnas
- Verificar si existen campos nulos
    - Si los existiera se debe hacer una analizar para tratar los datos
- Análisis de Target o clase objetivo
    - En este caso nos vamos a enfocar en el campo “Condicio_1”
- Análisis de Variables
    - En este paso se analizará la distribución de cada una de las variables en función del target o clases objetivo, de manera de poder identificar patrones identificando también la calidad de la data en función de encontrar outlier entre otros escenarios.
- Matriz de Correlación
    - Esta matriz permitirá evaluar la relación o dependencias entre las variables dentro del dataset obtenido, si hubiera una relación representativa, y de ser el caso elegir entre una de las dos variables.

**3.2.1.3 Preparación de la Data.** En esta tercera fase lo que realizamos es transformar las características nominales a enteras, para luego dividir la data en 70% la data de aprendizaje y el 30% la data de evaluación

**3.2.1.4 Modelado.** En esta siguiente fase lo que haremos es implementar cada uno de los modelos propuestos para este estudio tales como:

- Máquinas de Soporte Vectorial
- Regresión Logística Multinomial

- Clasificar Bayesiano Multinomial

**3.2.1.5 Evaluación.** En esta fase lo que haremos es evaluar los tres modelos en varias formas tales como:

- Curva ROC
- Accuracy Score
- Matriz de Confusión

**3.2.1.5 Despliegue.** En esta última fase lo que haremos es a partir de modelo generado más preciso crearlo para que pueda trabajar con datos totalmente nuevos, que para este caso serán del año 2023, a continuación, cargamos el modelo creado, y preparamos la data tal cual como lo hicimos para la data de aprendizaje, con el objetivo el modelo se comporte de manera correcta.

# 4. Capítulo IV. Resultados

## 4.1 Análisis del estado actual de los accidentes de tránsito en Ecuador

### *4.1.1 Comprensión del Negocio*

La agencia nacional de tránsito es la institución encargada de gestionar de forma centralizada de todos los registros de los accidentes de tránsito del país, sin embargo, existen instituciones regionales o cantonales que registran los datos de forma local, a continuación, mostraremos el listado correspondiente. (Transito, 2022)

- (PNE) Policía Nacional del Ecuador
    - El ámbito de operación es en la red vial estatal con excepción de aquellas circunscripciones de competencia de la Comisión de Tránsito del Ecuador y de las zonas urbanas, de jurisdicción de los Gobiernos Autónomos Descentralizados (Transito, 2022) (Transito, 2022)
- (CTE) Comisión de Tránsito del Ecuador
    - El ámbito de operación es en la red vial estatal con excepción de las zonas urbanas de competencia de los Gobiernos Autónomos Descentralizados y aquellas circunscripciones de competencia de la Policía Nacional (Transito, 2022)
- (DMQ) Agencia Metropolitana de Tránsito de Quito
    - El ámbito de operación dentro del Distrito Metropolitano de Quito
- (ATM) Agencia de Tránsito y Movilidad de Guayaquil
    - El ámbito de operación dentro del cantón Guayaquil
- (MAM) Dirección de Tránsito, Trasporte Terrestre y Seguridad Vial de Ambato
    - El ámbito de operación dentro del cantón Ambato

- (MBA) Autoridad de Tránsito Municipal de Babahoyo - TRANSVIAL EP
    - El ámbito de operación dentro del cantón Babahoyo
- (MCU) Empresa Pública de Movilidad de Cuenca - EMOV EP
    - El ámbito de operación dentro del cantón Cuenca
- (MES) Empresa Municipal de Tránsito de Esmeraldas - ESVIAL EP
    - El ámbito de operación dentro del cantón Esmeraldas
- (MLO) Unidad de Control Operativo de Tránsito del Municipio de Loja – UCOT
    - El ámbito de operación dentro del cantón Loja
- (MMA) Agencia Municipal De Transito Manta
    - El ámbito de operación dentro del cantón Manta
- (MMC) Movilidad Machala EP
    - El ámbito de operación es dentro del cantón Machala
- (MPO) Empresa Pública Municipal de Tránsito de Portoviejo - PORTOVIAL EP
    - El ámbito de operación es dentro del cantón Portoviejo
- (MRI) Dirección de Gestión de Movilidad, Tránsito y Transporte de la ciudad de Riobamba
    - El ámbito de operación es dentro del cantón Riobamba
- (MSD) Empresa Pública Municipal de Transporte Santo Domingo - EPMT-SD
    - El ámbito de operación es dentro del cantón Santo Domingo
- (MEP) Empresa Pública de Movilidad del Norte - MOVIDELNOR EP
    - El ámbito de operación es dentro de los cantones: San Pedro de Huaca, Montufar, Bolívar, Mira, Espejo, San Lorenzo, Eloy Alfaro, Ríoverde,

Ibarra, Urcuquí, Pimampiro, Antonio Ante, Cotacachi, Otavalo, Pedro Moncayo, que conforman la Mancomunidad.

Los empleados de organizaciones especiales en el campo de la gestión del tráfico en cada área, respectivamente, recopilan información cuando ocurre un accidente de tráfico, que es una declaración que registra toda la información necesaria sobre el evento, la recopilación se realiza. cada vez que ocurre un evento, es seguido por una recopilación, procesamiento y liberación mensuales. (Transito, 2022)

Los datos centralizados de siniestralidad a nivel nacional, será actualizada a partir del día 16 de cada mes, luego de la recopilación y procesamiento de los datos obtenidos de las diferentes fuentes. (Transito, 2022)

En relación con este proceso, existen dos casos en los cuales se agrupan a los entes de control, en función de la forma en la que proveen la información recolectada, a la Agencia Nacional de Tránsito. (Transito, 2022)

Los entes de control que no disponen de un sistema propio para almacenamiento de datos de siniestralidad ingresan y validan la información en el Sistema Nacional de Estadísticas de Tránsito SINET, que es el sistema de la ANT en el cual se consolida la información de siniestralidad a nivel nacional. (Transito, 2022)

Los entes de control que disponen de un sistema propio para almacenamiento de datos de siniestralidad ingresan la información directamente en sus sistemas propios, y la misma es consumida por la ANT, a través de una interconexión o enlace con el Sistema Nacional de Estadísticas de Tránsito SINET. (Transito, 2022)

Una vez que la información de cada parte es ingresada y validada en el Sistema Nacional de Estadísticas de Tránsito SINET, por parte de todos los entes de control, la Unidad de Estadísticas de la Dirección de Estudios y Proyectos de la ANT, descarga las bases de datos para su respectivo procesamiento. (Transito, 2022)

### *4.1.2 Problemática para resolver*

Según el INEC, y la Agencia Nacional de Tránsito de Ecuador, en el año 2020 se registraron 44.242 accidentes de tráfico en el país, lo que representa un aumento del 1,3% respecto al año anterior. Estos accidentes dejaron un saldo de 2.780 personas fallecidas, lo que significa una tasa de mortalidad de 16,8 por cada 100.000 habitantes. Estas cifras son preocupantes y evidencian la necesidad de desarrollar medidas preventivas para reducir la cantidad de accidentes y muertes en el país. (INEC, 2022)

### *4.1.3 Objetivos del negocio*

Los objetivos planteados en base a la problemática planteada se pretenden:

- Identificar una escala entre las variables más importantes que intervienen dentro de un accidente de transito
- Implementar el modelo más precisión que pueda predecir esta la mortalidad en los accidentes de tránsito

### *4.1.4 Criterios de éxito*

Los registros recabados por la agencia nacional de tránsito en función de todos los accidentes de tránsito registrados serán utilizados para realizar una limpieza de estos, en función de eso analizar la data e identificar los patrones de comportamiento que existan en cada una de las variables como en el target de estudio planteado.

Partiendo de este análisis se implementarán tres modelos de aprendizaje predictivo y poder identificar el modelo más eficiente basado en métricas como la curva roc, la matriz de confusión el accuracy_score, sin embargo sin dejar de lado que con la implementación de la técnica de chi-cuadrado podremos identificar las variables que más influyen para que exista un accidente, dejando este conocimiento para que los organismo que corresponda puedan generar campañas o mejoras para evitar el número de accidentes de tránsito y aún más la mortalidad que existe

## 4.2 Aplicación de las técnicas de Minería de Datos

La implementación de la minería de datos como hemos planteado anteriormente se usó la metodología CRISP-DM

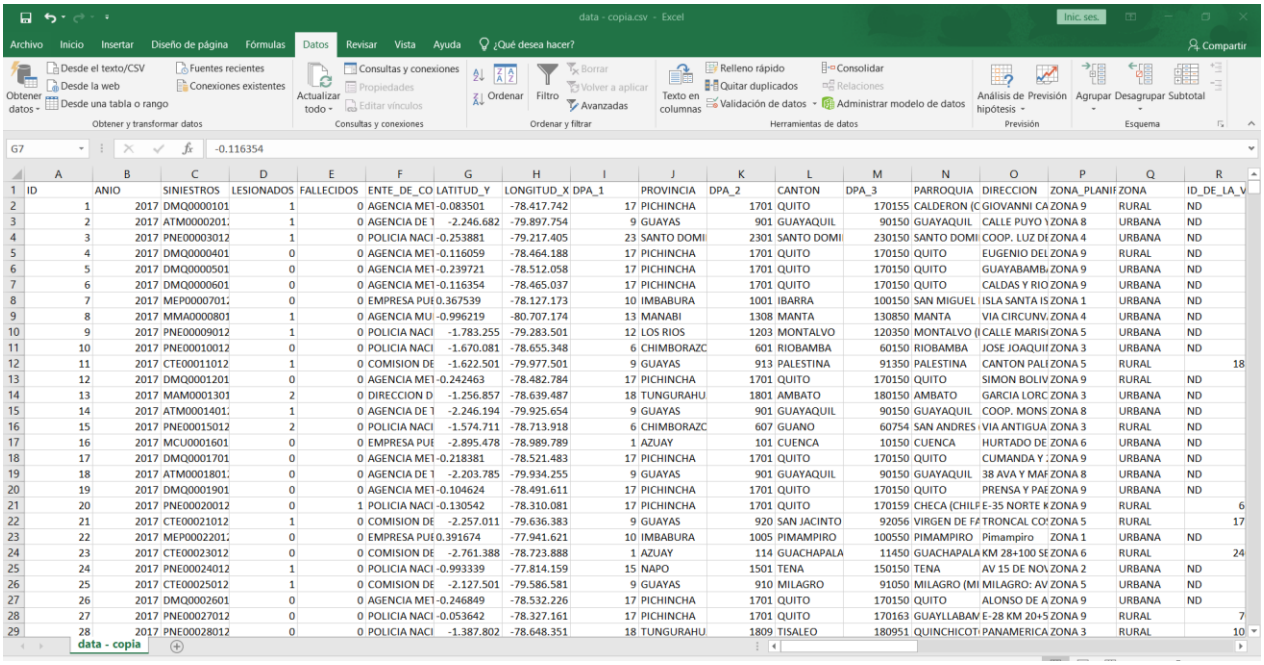
### *4.2.1 Comprensión del Negocio*

Bueno hemos podido identificar que la Agencia Nacional de Transito tiene distribuido el registro de los accidentes de tránsito en varias entidades, como vimos en la sección (4.1.1), en el cual cada uno de ellos son los que registran localmente los mismo y mediante el sistema de gestión de accidentes envían los registros a la agencia nacional de tránsito para de esta manera tener centralizada y administrada por la Agencia antes mencionada.

### *4.2.2 Comprensión de la Data*

Para obtener la data se descargó un archivo .csv de la página de la Agencia nacional de tránsito, mismos que tenían la data inicialmente hasta marzo del año actual, que posee 142582 registro y 56 columnas, como se muestra en la Figura 2.

**Figura 2 Datos de Estudio**



| ID | ANIO | SINIESTROS | LESIONADOS | FALLECIDOS | ENTE_DE_CO | LATITUD_Y | LONGITUD_X | DPA_1 | PROVINCIA | DPA_2 | CANTON | DPA_3 | PARROQUIA | DIRECCION | ZONA_PLANIF | ZONA | ID_DE_LA_V |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 2017 | DMQ0000101 | 1 | 0 | AGENCIA MET | -0.083501 | -78.417.742 | 17 | PICHINCHA | 1701 | QUITO | 170155 | CALDERON (C | GIOVANNI CA | ZONA 9 | RURAL | ND |
| 2 | 2017 | ATM0000201 | 1 | 0 | AGENCIA DE T | -2.246.682 | -79.897.754 | 9 | GUAYAS | 901 | GUAYAQUIL | 90150 | GUAYAQUIL | CALLE PUYO Y | ZONA 8 | URBANA | ND |
| 3 | 2017 | PNE00003012 | 1 | 0 | POLICIA NACI | -0.253881 | -79.217.405 | 23 | SANTO DOMI | 2301 | SANTO DOMI | 230150 | SANTO DOMI | COOP. LUZ DE | ZONA 4 | URBANA | ND |
| 4 | 2017 | DMQ0000401 | 0 | 0 | AGENCIA MET | -0.116059 | -78.464.188 | 17 | PICHINCHA | 1701 | QUITO | 170150 | QUITO | EUGENIO DEL | ZONA 9 | RURAL | ND |
| 5 | 2017 | DMQ0000501 | 0 | 0 | AGENCIA MET | -0.239721 | -78.512.058 | 17 | PICHINCHA | 1701 | QUITO | 170150 | QUITO | GUAYABAMBA | ZONA 9 | URBANA | ND |
| 6 | 2017 | DMQ0000601 | 0 | 0 | AGENCIA MET | -0.116354 | -78.465.037 | 17 | PICHINCHA | 1701 | QUITO | 170150 | QUITO | CALDAS Y RIO | ZONA 9 | URBANA | ND |
| 7 | 2017 | MEP0000701 | 0 | 0 | EMPRESA PUE | 0.367539 | -78.127.173 | 10 | IMBABURA | 1001 | IBARRA | 100150 | SAN MIGUEL | ISLA SANTA IS | ZONA 1 | URBANA | ND |
| 8 | 2017 | MMA0000801 | 1 | 0 | AGENCIA MU | -0.996219 | -80.707.174 | 13 | MANABI | 1308 | MANTA | 130850 | MANTA | VIA CIRCUNV | ZONA 4 | URBANA | ND |
| 9 | 2017 | PNE00009012 | 1 | 0 | POLICIA NACI | -1.783.255 | -79.283.501 | 12 | LOS RIOS | 1203 | MONTALVO | 120350 | MONTALVO (I | CALLE MARIS | ZONA 5 | URBANA | ND |
| 10 | 2017 | PNE00010012 | 0 | 0 | POLICIA NACI | -1.670.081 | -78.655.348 | 6 | CHIMBORAZO | 601 | RIOBAMBA | 60150 | RIOBAMBA | JOSE JOAQUI | ZONA 3 | URBANA | ND |
| 11 | 2017 | CTE00011012 | 1 | 0 | COMISION DE | -1.622.501 | -79.977.501 | 9 | GUAYAS | 913 | PALESTINA | 91350 | PALESTINA | CANTON PALI | ZONA 5 | RURAL | 18 |
| 12 | 2017 | DMQ0001201 | 0 | 0 | AGENCIA MET | -0.242463 | -78.482.784 | 17 | PICHINCHA | 1701 | QUITO | 170150 | QUITO | SIMON BOLIV | ZONA 9 | RURAL | ND |
| 13 | 2017 | MAM0001301 | 2 | 0 | DIRECCION D | -1.256.857 | -78.639.487 | 18 | TUNGURAHU | 1801 | AMBATO | 180150 | AMBATO | GARCIA LORC | ZONA 3 | URBANA | ND |
| 14 | 2017 | ATM0001401 | 1 | 0 | AGENCIA DE T | -2.246.194 | -79.925.654 | 9 | GUAYAS | 901 | GUAYAQUIL | 90150 | GUAYAQUIL | COOP. MONS | ZONA 8 | URBANA | ND |
| 15 | 2017 | PNE00015012 | 2 | 0 | POLICIA NACI | -1.574.711 | -78.713.918 | 6 | CHIMBORAZO | 607 | GUANO | 60754 | SAN ANDRES | VIA ANTIGUA | ZONA 3 | RURAL | ND |
| 16 | 2017 | MCU0001601 | 0 | 0 | EMPRESA PUE | -2.895.478 | -78.989.789 | 1 | AZUAY | 101 | CUENCA | 10150 | CUENCA | HURTADO DE | ZONA 6 | URBANA | ND |
| 17 | 2017 | DMQ0001701 | 0 | 0 | AGENCIA MET | -0.218381 | -78.521.483 | 17 | PICHINCHA | 1701 | QUITO | 170150 | QUITO | CUMANDA Y | ZONA 9 | URBANA | ND |
| 18 | 2017 | ATM0001801 | 0 | 0 | AGENCIA DE T | -2.203.785 | -79.934.255 | 9 | GUAYAS | 901 | GUAYAQUIL | 90150 | GUAYAQUIL | 38 AVA Y MAF | ZONA 8 | URBANA | ND |
| 19 | 2017 | DMQ0001901 | 0 | 0 | AGENCIA MET | -0.104624 | -78.491.611 | 17 | PICHINCHA | 1701 | QUITO | 170150 | QUITO | PRENSA Y PAE | ZONA 9 | URBANA | ND |
| 20 | 2017 | PNE00020012 | 0 | 1 | POLICIA NACI | -0.130542 | -78.310.081 | 17 | PICHINCHA | 1701 | QUITO | 170159 | CHECA (CHILF | E-35 NORTE K | ZONA 9 | RURAL | 6 |
| 21 | 2017 | CTE00021012 | 1 | 0 | COMISION DE | -2.257.011 | -79.636.383 | 9 | GUAYAS | 920 | SAN JACINTO | 92056 | VIRGEN DE FA | TRONCAL CO | ZONA 5 | RURAL | 17 |
| 22 | 2017 | MEP0002201 | 0 | 0 | EMPRESA PUE | 0.391674 | -77.941.621 | 10 | IMBABURA | 1005 | PIMAMPIRO | 100550 | PIMAMPIRO | Pimampiro | ZONA 1 | URBANA | ND |
| 23 | 2017 | CTE00023012 | 0 | 0 | COMISION DE | -2.761.388 | -78.723.888 | 1 | AZUAY | 114 | GUACHAPALA | 11450 | GUACHAPALA | KM 28+100 SE | ZONA 6 | RURAL | 24 |
| 24 | 2017 | PNE00024012 | 1 | 0 | POLICIA NACI | -0.993339 | -77.814.159 | 15 | NAPO | 1501 | TENA | 150150 | TENA | AV 15 DE NO | ZONA 2 | URBANA | ND |
| 25 | 2017 | CTE00025012 | 1 | 0 | COMISION DE | -2.127.501 | -79.586.581 | 9 | GUAYAS | 910 | MILAGRO | 91050 | MILAGRO (MI | MILAGRO: AV | ZONA 5 | URBANA | ND |
| 26 | 2017 | DMQ0002601 | 0 | 0 | AGENCIA MET | -0.246849 | -78.532.226 | 17 | PICHINCHA | 1701 | QUITO | 170150 | QUITO | ALONSO DE A | ZONA 9 | URBANA | ND |
| 27 | 2017 | PNE00027012 | 0 | 0 | POLICIA NACI | -0.053642 | -78.327.161 | 17 | PICHINCHA | 1701 | QUITO | 170163 | GUAYLLABAM | E-28 KM 20+5 | ZONA 9 | RURAL | 7 |
| 28 | 2017 | PNE00028012 | 0 | 0 | POLICIA NACI | -1.387.802 | -78.648.351 | 18 | TUNGURAHU | 1809 | TISALEO | 180951 | QUINCHICOT | PANAMERICA | ZONA 3 | RURAL | 10 |

**4.2.2.1 Descripción de los Datos.** En función del archivo data.csv se obtuvo 56 columnas las mismas a continuación se detallan cada una de ellas en la Tabla 1.

**Tabla 1 Estructura Dataset**

| CARACTERISTICAS | DESCRIPCIÓN | TIPO DE DATO |
|---|---|---|
| **ID** | Es el identificador único de cada instancia | Int64 |
| **ANIO** | Es el año del accidente | Int64 |
| **SINIESTROS** | Es código de único del accidente | Object |
| **LESIONADOS** | Es el número de Lesionados en el accidente | Int64 |
| **FALLECIDOS** | Es el número de Fallecidos en el accidente | Int64 |

| | | |
|---|---|---|
| **ENTE_DE_CONTROL** | Es el ente de control que registro el accidente | Object |
| **LATITUD_Y** | Es la coordenada de la geolocalización del accidente en eje X | Float |
| **LONGITUD_X** | Es la coordenada de la geolocalización del accidente en eje Y | Float |
| **DPA_1** | Es el código de la provincia donde ocurrió el accidente | Int64 |
| **PROVINCIA** | Es el nombre de la provincia donde ocurrió el accidente | Object |
| **DPA_2** | Es el código del cantón donde ocurrió el accidente | Int64 |
| **CANTON** | Es el nombre del cantón donde ocurrió el accidente | Object |
| **DPA_3** | Es el código de la parroquia donde ocurrió el accidente | Int64 |
| **PARROQUIA** | Es el nombre de la parroquia donde ocurrió el accidente | Object |
| **DIRECCION** | Es la dirección de accidente | Object |

| | | |
|---|---|---|
| **ZONA_PLANIFICACION** | Es la distribución por zonas en donde ocurrió el accidente "Zona (1-9)" | Object |
| **ZONA** | Es el tipo de zona "Rural", "Urbana" | Object |
| **ID_DE_LA_VIA** | Es el identificador único la vía | Object |
| **NOMBRE_DE_LA_VIA** | Es nombre de la vía | Object |
| **UBICACION_DE_LA_VIA** | Es la ubicación de la vía | Object |
| **JERARQUIA_DE_LA_VIA** | Es el código de jerarquía de la vía | Object |
| **FECHA** | Es la fecha del accidente | Object |
| **HORA** | Es la hora minutos y segundos aproximada del accidente | Object |
| **PERIODO_1** | Es la hora agrupada aproximada del accidente | Object |
| **PERIODO_2** | Es el código de la hora agrupada aproximada del accidente | Int64 |
| **DIA_1** | Es el día de la semana del accidente (Lunes, Martes, Miércoles, Jueves, Viernes, Sábado, Domingo) | Object |

| | | |
|---|---|---|
| **DIA_2** | Es en código del día de la semana del accidente (1-7) | Int64 |
| **MES_1** | Es el nombre del mes cuando ocurrió el accidente | Object |
| **MES_2** | Es el código del mes del nombre del mes cuando ocurrió el accidente | Int64 |
| **FERIADO** | Es un campo si o no que indica si el accidente fue en feriado | Object |
| **CODIGO_CAUSA** | Es el código de la causa probable del accidente | Object |
| **CAUSA_PROBABLE** | Es el nombre de la causa probable del accidente | Object |
| **TIPO_DE_SINIESTRO** | Es el tipo de siniestro registrado por cada accidente | Object |
| **TIPO_DE_VEHICULO_1** | Es el tipo de vehículo "Motocicleta" , "Automóvil" ,"Furgoneta", "Bus", "Camioneta", "Camión", “VEHÍCULO DEPORTIVO UTILITARIO" | Object |
| **SERVICIO_1** | Es el nombre del tipo de servicio "PUBLICO","PARTICULAR" | Object |

| | | |
|---|---|---|
| **AUTOMOVIL** | Es el número de automóviles en el accidente | Int64 |
| **BICICLETA** | Es el número de bicicletas en el accidente | Int64 |
| **BUS** | Es el número de bus en el accidente | Int64 |
| **CAMION** | Es el número de camión en el accidente | Int64 |
| **CAMIONETA** | Es el número de camionetas en el accidente | Int64 |
| **EMERGENCIAS** | Es el número de vehículos de emergencias en el accidente | Int64 |
| **ESPECIAL** | Es el número de vehículos especiales en el accidente | Int64 |
| **FURGONETA** | Es el número de furgonetas en el accidente | Int64 |
| **MOTOCICLETA** | Es el número de motocicletas en el accidente | Int64 |
| **NO_IDENTIFICADO** | Es el número de no identificados en el accidente | Int64 |
| **SCOOTER_ELECTRICO** | Es el número de scotters en el accidente | Int64 |

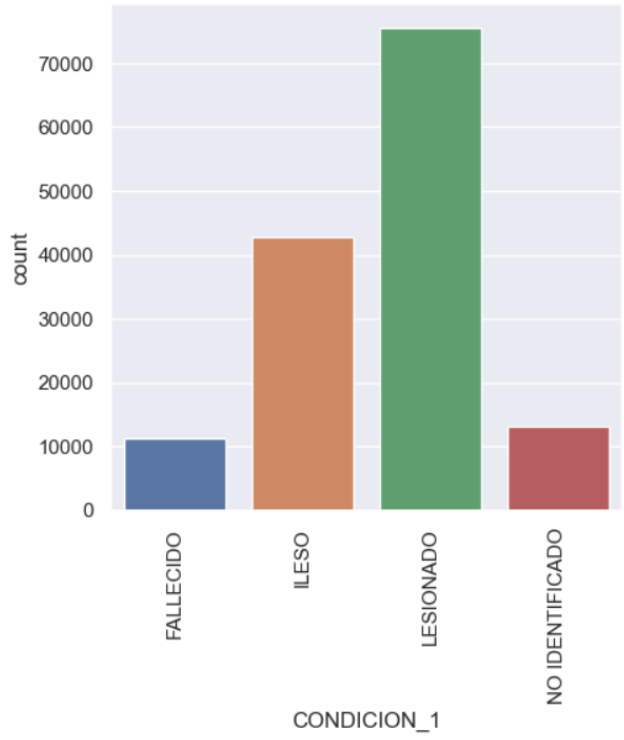
| | | |
|---|---|---|
| **TRICIMOTO** | Es el número de tricimotos en el accidente | Int64 |
| **VEHICULO_DEPORTIVO_ UTILITARIO** | Es el número de vehículos deportivos en el accidente | Int64 |
| **SUMA_DE_VEHICULOS** | Es número de vehículos en el accidente | Int64 |
| **TIPO_ID_1** | Es el tipo de documento de identificación "Cedula","Pasaporte","Licencia" | Object |
| **EDAD_1** | Es la edad de participante en el accidente | Object |
| **SEXO_1** | Es el sexo del participante en el accidente "Masculino”,"Femenino","No identificado" | Object |
| **CONDICION_1** | Es la condición del participante "Lesionado”, "Ileso", "Fallecido","No identificado" | Object |
| **PARTICIPANTE_1** | Es el tipo de participante "Conductor”, "Pasajero”, Peatón" | Object |

| | | |
|---|---|---|
| **CASCO_1** | Es el identificado si usa o no casco | Object |
| **CINTURON_1** | Es el identificador si usa o no cinturón | Object |

**4.2.2.2 Análisis del Target.** Se identifica el campo CONDICION_1 como el campo target la cual realizando un análisis preliminar se identifica que es un variable nominal que tiene inicialmente cuatro tipos 'LESIONADO','NO IDENTIFICADO','ILESO', y 'FALLECIDO'.

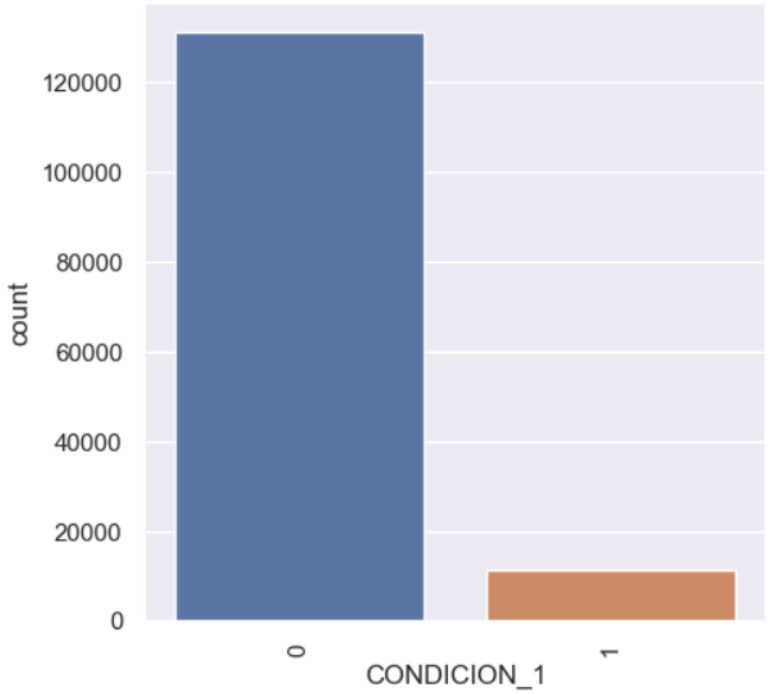
A continuación, en la Figura 3 analizamos como está la distribución del target en donde se identifica:

**Figura 3 Análisis del Target**



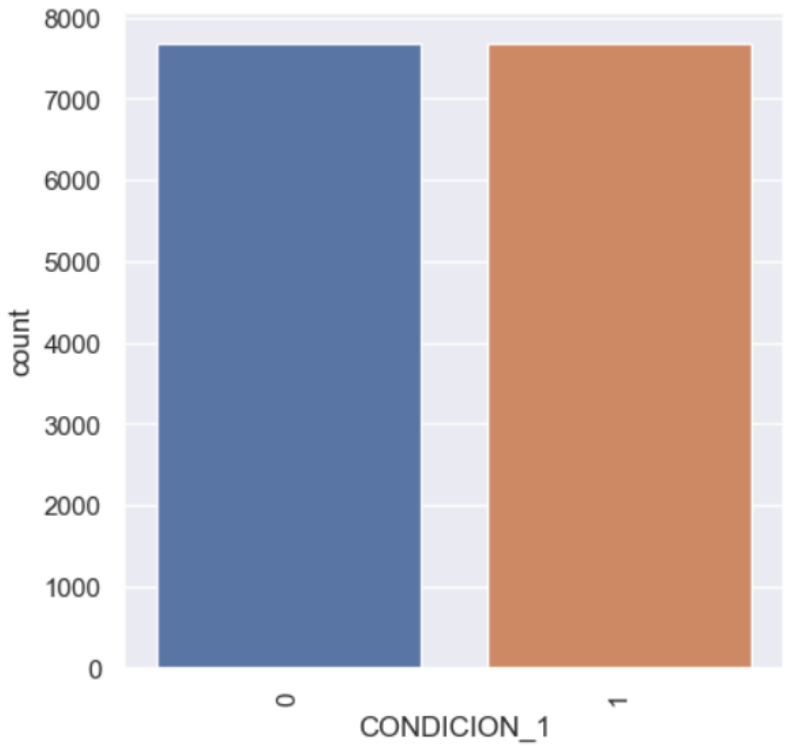
Para el análisis de este estudio se requiere analizar el ítem fallecido por esta razón restructuraremos la variable CONDICION_1, realizando una transformación a una variable dicotómica donde se diferenciará entre no fallecido como “0”, donde entraran los escenarios de ileso, lesionado y no identificado y como fallecido como “1” donde entrara lógicamente el escenario de fallecido, como se muestra en la Figura 4.

**Figura 4 Distribución del Target**




Donde podemos visualizar es que nuestro dataset esta descompasado y dado que nuestro estudio se basa específicamente en la mortalidad, vamos a balancear nuestro dataset para no tener un sesgo en el mismo, como se muestra la Figura 5.

**Figura 5 Análisis del Target Limpio**




Para obtener esta grafica lo que se realizó varios filtros, el primero fue trabajar solo con datos desde el 2017 al 2022, no tomas en cuenta la condición no identificada dado que nos puede generar sesgos en los modelos que posteriormente construimos, quedando con un dataset de 15360 registros y por ahora mantenemos las 56 columnas.

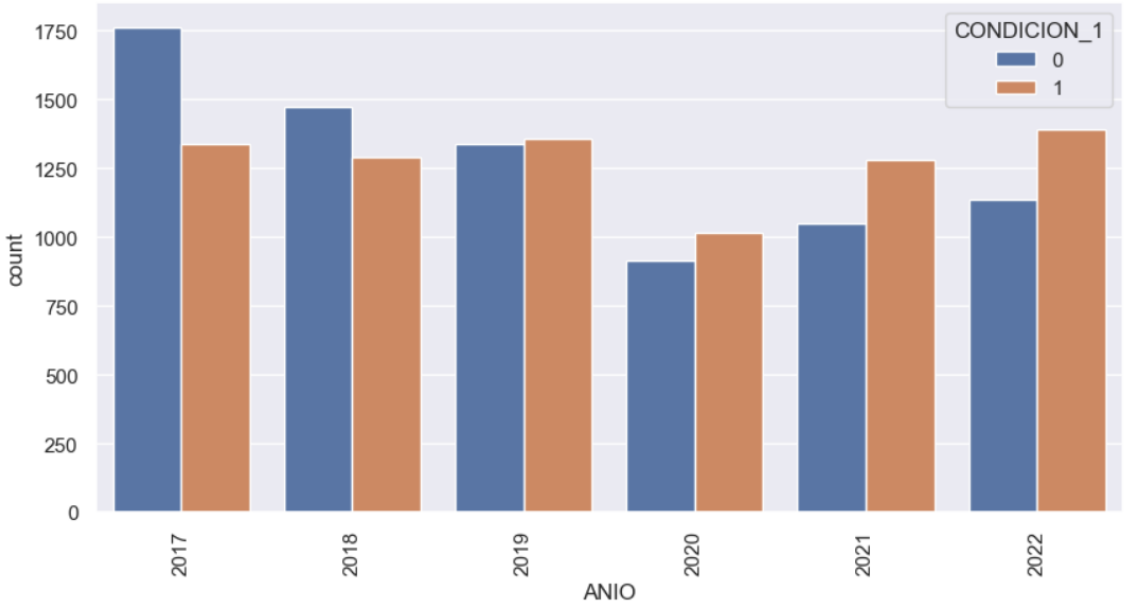
**4.2.2.3 Análisis de Variables.** El análisis de las variables la realizamos con el objetivo de identificar patrones o comportamientos preliminar de cada una de las variables numéricas del dataset.

***4.2.2.3.1 Año.*** Se realizo un gráfico de distribución de la variable año en función de la variable de estudio en este caso la condición.

La Figura 6 muestra el análisis de accidentes por año desde 2017 hasta 2022, dividiendo el número de accidentes en dos categorías con o sin fallecidos: "sí"=>1  y "no" =>0. Se ha

observado que habido un promedio de 1200 muertes por accidentes de tránsito en los años anteriores, excluyendo los años afectados por la pandemia de COVID-19, es decir, 2020 y 2021, en los cuales el número de muertes se redujo a 1000 aproximadamente.

**Figura 6 Variable Año**




Esta disminución en el número de fallecidos durante los años de la pandemia puede atribuirse a varias razones. Las restricciones de movilidad y el confinamiento impuestos durante esos años redujeron significativamente la actividad vial y, por lo tanto, disminuyeron las oportunidades de accidentes. Además, las campañas de concienciación sobre la seguridad vial y las medidas de control más estrictas podrían haber contribuido a esta reducción en el número de fallecidos.

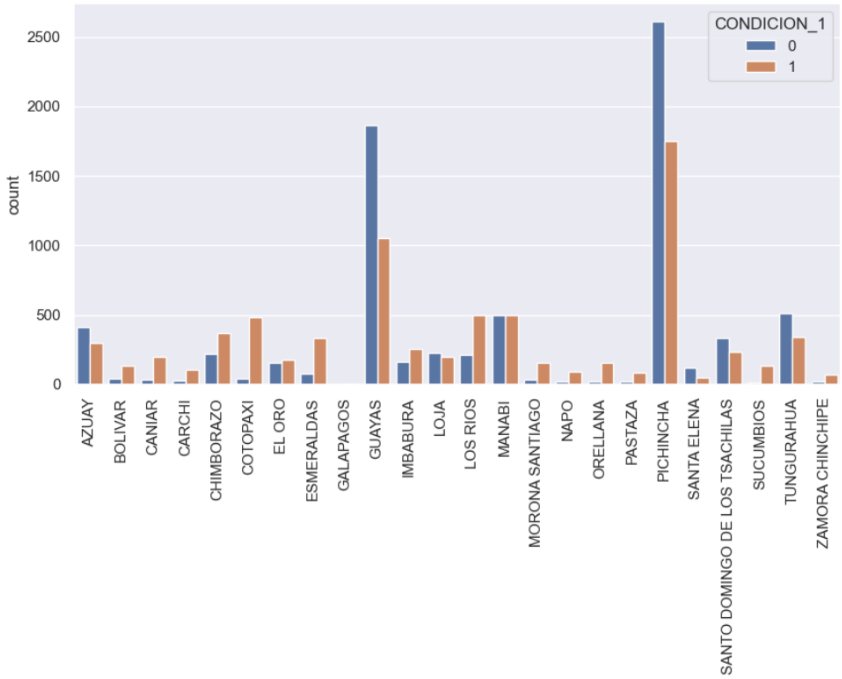
Sin embargo, es preocupante observar que el último año, 2022, ha experimentado un aumento en el número de fallecidos por accidentes de tránsito, superando ligeramente el

promedio histórico de 1200 muertes. Este aumento puede indicar la necesidad aun mayor de atención y acciones preventivas para abordar esta problemática.

***4.2.2.3.2 Provincia.*** Se realizo un gráfico de distribución de la variable provincia en función de la variable de estudio en este caso la condición.

La Figura 7 proporciona un análisis de accidentes de tránsito por provincia, dividiendo los accidentes en dos categorías: con o sin fallecidos "sí"=>1  y "no" =>0. Se observa que hay dos provincias destacadas en términos de accidentes de tránsito: Pichincha y Guayas. Entre estas dos provincias, Pichincha se destaca como la provincia con la mayor cantidad de accidentes, así como el mayor número de muertes registradas desde 2017 hasta 2022.

**Figura 7 Variable Provincia**




La información proporcionada demuestra la importancia de invertir más recursos en la provincia de Pichincha para solucionar el problema de los accidentes de tránsito. El análisis

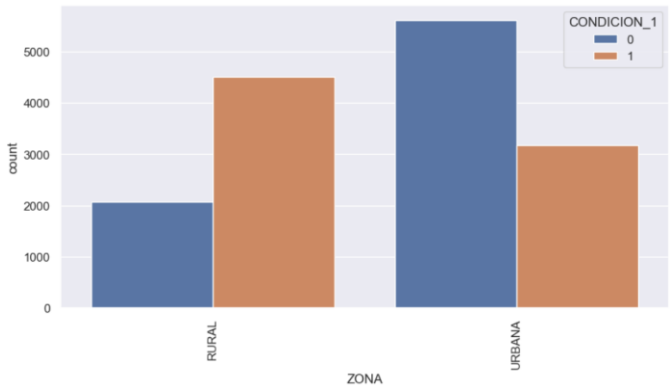
exhaustivo de las causas fundamentales de estos accidentes puede proporcionar una base sólida para la implementación de medidas preventivas y estrategias de seguridad vial más efectivas.

En la provincia de Cotopaxi, es importante destacar que el número de fallecidos supera el número de accidentes reportados. Esta discrepancia es alarmante y indica que los accidentes de tránsito en Cotopaxi son particularmente mortales y tienen graves consecuencias para las personas involucradas. Este patrón requiere investigación más profunda para comprender las causas específicas y tomar medidas preventivas adecuadas.

***4.2.2.3.3 Zona.*** Se realizo un gráfico de distribución de la variable zona en función de la variable de estudio en este caso la condición.

En la Figura 8 muestra el análisis de las zonas "Rural" y "Urbana" en relación con el número de accidentes divididos en dos categorías con o sin fallecidos: "sí"=>1  y "no" =>0, representando si existió muertos en cada accidente. Al examinar los datos, se observa una diferencia significativa en los resultados entre las dos zonas. En la zona urbana, se evidencia un mayor número de accidentes sin muertos, mientras que en la zona rural se observa un mayor número de accidentes con muertos.

**Figura 8 Variable Zona**



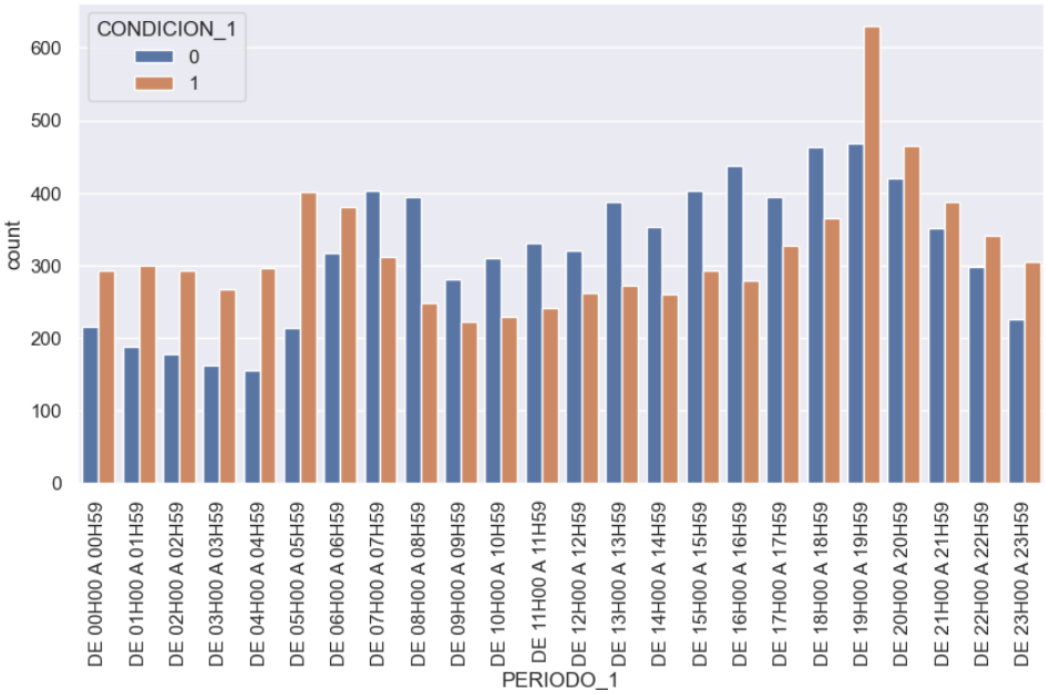
Este descubrimiento es crucial para comprender y abordar los problemas de seguridad vial en diversos entornos. La densidad de población y el flujo constante de tráfico pueden estar relacionados con la mayor cantidad de accidentes sin muertos en las áreas urbanas. Es probable que ocurran más colisiones y percances de menor gravedad en estas áreas, lo que aumenta la probabilidad de accidentes sin víctimas fatales.

Sin embargo, la región rural se encuentra en una situación más preocupante debido al aumento en el número de accidentes mortales. La infraestructura vial deficiente, las condiciones de la carretera, la iluminación insuficiente y la distancia entre los servicios de emergencia y las áreas rurales son algunos de los factores que pueden afectar esta disparidad. Estos factores pueden aumentar la gravedad de los accidentes y retrasar la respuesta a las emergencias.

***4.2.2.3.4 Periodo.*** Se realizo un gráfico de distribución de la variable zona en función de la variable de estudio en este caso la condición.

La Figura 9 muestra un rango de horas separadas por una hora y el número de accidentes divididos en dos categorías con o sin fallecidos: "sí"=>1  y "no" =>0. Al analizar los datos, se puede observar que hay un patrón claro en los accidentes registrados. El rango de tiempo con mayor cantidad de accidentes en la categoría "sí" es de 19:00 a 20:00. Este hallazgo es de gran importancia y merece una atención especial, ya que indica que hay un mayor riesgo de accidentes durante esa hora.

**Figura 9 Variable Periodo**

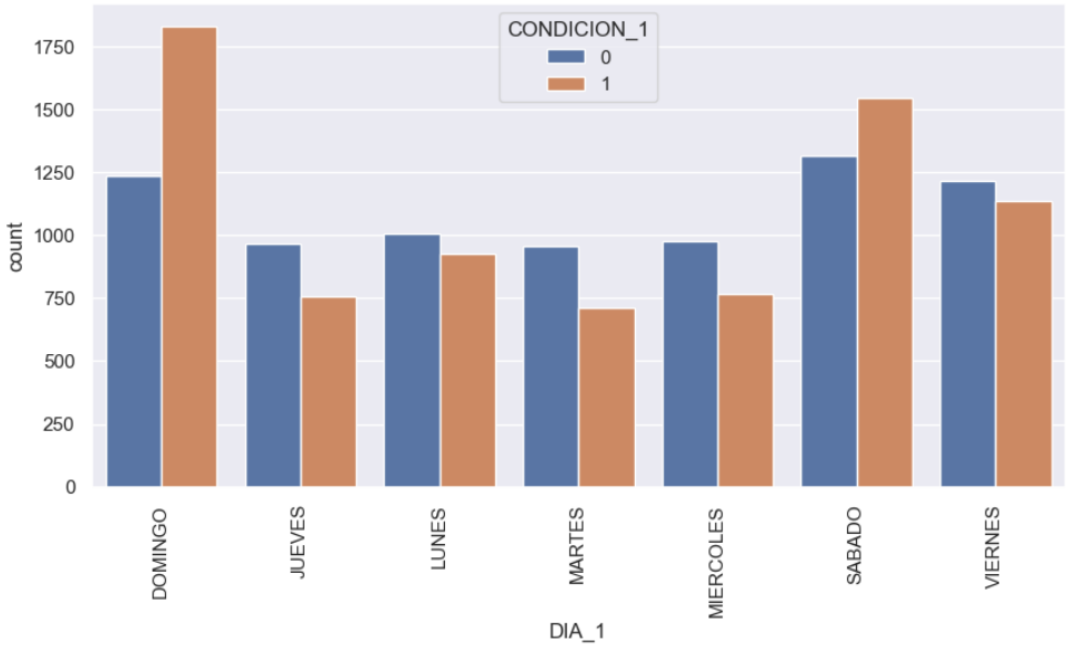



La hora pico de 19:00 a 20:00 es un período crítico para la seguridad vial, y es necesario investigar las posibles causas de este aumento en los accidentes. Podría haber varios factores que contribuyan a esta tendencia, como el tráfico congestionado durante la hora punta, la fatiga de los conductores después de un largo día de trabajo o la reducción de la visibilidad debido a la puesta de sol.

***4.2.2.3.5 Día.*** Se realizo un gráfico de distribución de la variable día en función de la variable de estudio en este caso la condición.

La Figura 10 proporciona un análisis de accidentes de tránsito por día de la semana, dividiendo los accidentes en dos categorías: "sí" y "no" en términos de si hubo fallecidos o no. Se observa que los fines de semana, en general, son los días con más accidentes de tránsito, y el domingo en particular muestra un número alto de fatalidades.

**Figura 10 Variable Día**




Este resultado indica que, dado que los fines de semana parecen ser los momentos de mayor riesgo en las vías, es esencial prestar especial atención a los comportamientos de conducción. El aumento del tráfico, las normas de conducción relajadas y el consumo de alcohol pueden contribuir a esta situación.

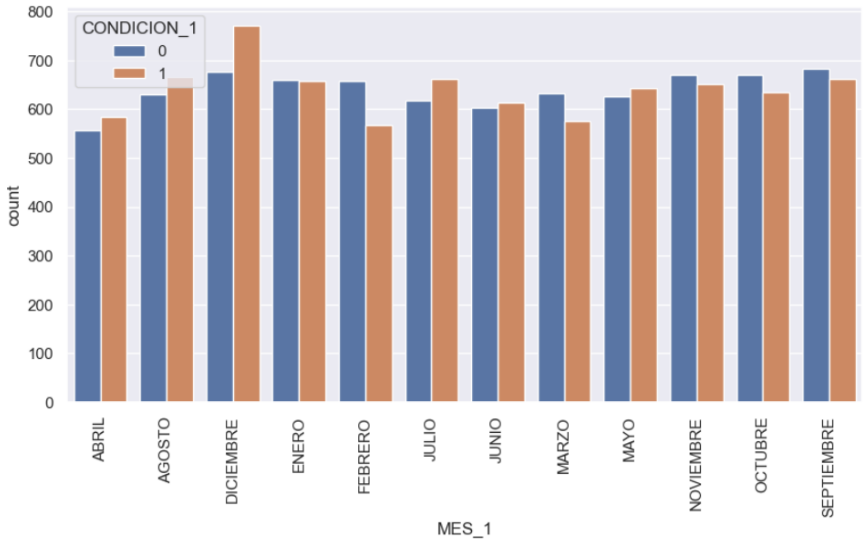
Es crucial señalar que el resto de los siete días de la semana tienen un promedio de 1000 accidentes por día. Es importante recordar que, aunque esta cifra puede parecer significativa, se trata de un promedio y que cada día puede haber variaciones en la cantidad de accidentes y su gravedad.

***4.2.2.3.6 Mes.*** Se realizo un gráfico de distribución de la variable mes en función de la variable de estudio en este caso la condición.

La Figura 11, el análisis de accidentes de tránsito por mes del año, dividiendo los accidentes en dos categorías: "sí" y "no" en términos de si hubo fallecidos o no. Se observa que,

en general, existe un comportamiento bastante distribuido entre todos los meses, pero destaca el mes de diciembre como el que registra tanto la mayor cantidad de accidentes como el mayor número de fallecidos.

**Figura 11 Variable Mes**

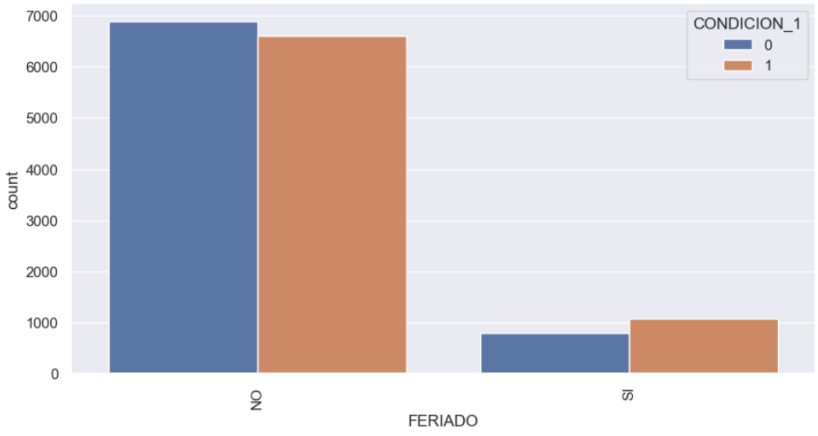



Es preocupante que diciembre sea el mes con mayor número de accidentes y muertes en comparación con otros meses, y requiere una atención especial. El aumento del tráfico debido a las festividades, las condiciones climáticas desfavorables, el aumento de la velocidad y el consumo de alcohol durante las celebraciones son algunas de las causas de esta tendencia.

***4.2.2.3.7 Feriado.*** Se realizo un gráfico de distribución de la variable mes en función de la variable de estudio en este caso la condición.

Como se puede visualizar en la Figura 12 no existe una proporción de accidentes representativa en un feriado versus cuando no es feriado.

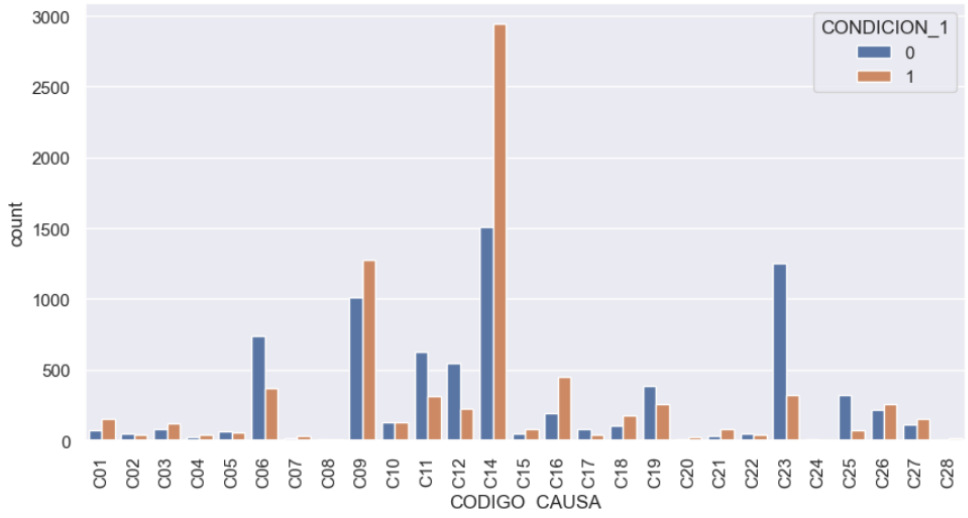
**Figura 12 Variable Feriado**




***4.2.2.3.8 Causa Probable.*** Se realizo un gráfico de distribución de la variable causa probable en función de la variable de estudio en este caso la condición.

La Figura 13 muestra el análisis de accidentes de tránsito por código de causa, dividiendo los accidentes en dos categorías: "sí" y "no" en términos de si hubo fallecidos o no. Se destaca que la causa "C14", que corresponde a "conducir desatento a las condiciones de tránsito (celular, pantallas de video, comida, maquillaje o cualquier otro elemento distractor)", es la causa con mayor mortalidad, con una diferencia abismal en comparación con otras causas.

**Figura 13 Variable Causa Probable**



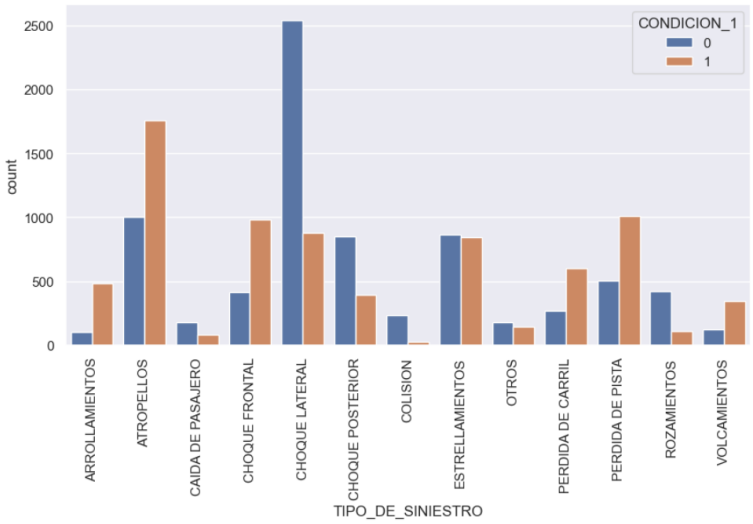
Adicionalmente también se destaca el código “C09” que corresponda “conducir vehículo superando los límites máximos de velocidad” es la segunda con mayor mortalidad, con la diferencia que casi es similar que ocasiona tanto fallecimientos como no.

Esta información destaca la gravedad y el impacto de la conducción desatenta en las condiciones de tránsito. El uso de dispositivos móviles, la distracción con pantallas de video, la alimentación, el maquillaje u otras distracciones pueden reducir significativamente la atención del conductor y aumentar significativamente el riesgo de accidentes y fallecimientos.

***4.2.2.3.9 Tipo de Siniestro.*** Se realizo un gráfico de distribución de la variable tipo de siniestro en función de la variable de estudio en este caso la condición.

La Figura 14 muestra el análisis de accidentes de tránsito por tipo de siniestro, dividiendo los accidentes en dos categorías: "sí" y "no" en términos de si hubo fallecidos o no. Se evidencia que el "choque lateral" es el siniestro que más accidentes causa, mientras que los "atropellos" son el siniestro con el mayor índice de accidentes y mayor mortalidad.

**Figura 14 Tipo de Siniestro**



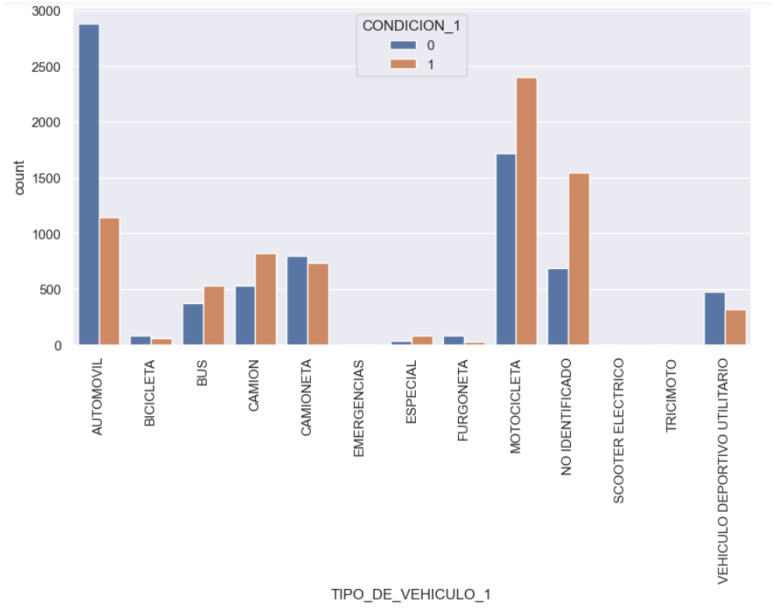
El hecho de que el siniestro más frecuente sea el "choque lateral" demuestra la relevancia de aumentar la conciencia situacional de los conductores y fomentar conductas defensivas. Los choques laterales pueden ocurrir debido a la falta de atención, los errores de juicio, las maniobras inadecuadas o el incumplimiento de las normas de tránsito.

Por otro lado, los "atropellos" son especialmente preocupantes debido a su alta tasa de mortalidad. Este tipo de accidente ocurre cuando un vehículo impacta directamente an un peatón u otro usuario vulnerable de la vía.

***4.2.2.3.10 Tipo de Vehículo.*** Se realizo un gráfico de distribución de la variable tipo de siniestro en función de la variable de estudio en este caso la condición.

La Figura 15 muestra el análisis de accidentes de tránsito por tipo de vehículo, dividiendo los accidentes en dos categorías: "sí" y "no" en términos de si hubo fallecidos o no. Se evidencia que existen más accidentes de tránsito en vehículos en general, pero los fallecimientos son más frecuentes en motocicletas.

**Figura 15 Variable Tipo de Vehículo**

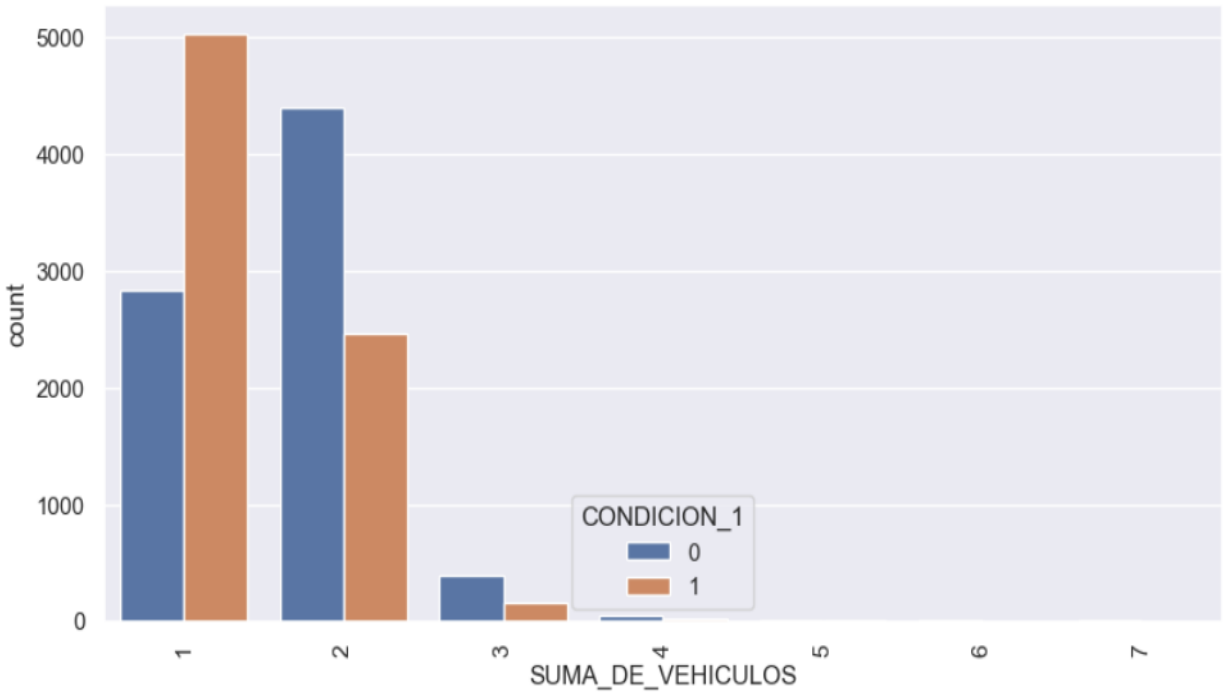


Es preocupante que los fallecimientos sean más comunes en motocicletas. Las motocicletas son vehículos más pequeños y ofrecen menos protección en caso de choque. La falta de obstáculos físicos entre el conductor y el entorno también aumenta el riesgo de lesiones graves o fatales. Es fundamental que los motociclistas reciban capacitación adecuada en técnicas de conducción segura, utilicen equipo de protección personal y comprendan los riesgos asociados con la conducción de motocicletas.

***4.2.2.3.11 Suma de Vehículos.*** Se realizo un gráfico de distribución de la variable suma de vehículos en función de la variable de estudio en este caso la condición.

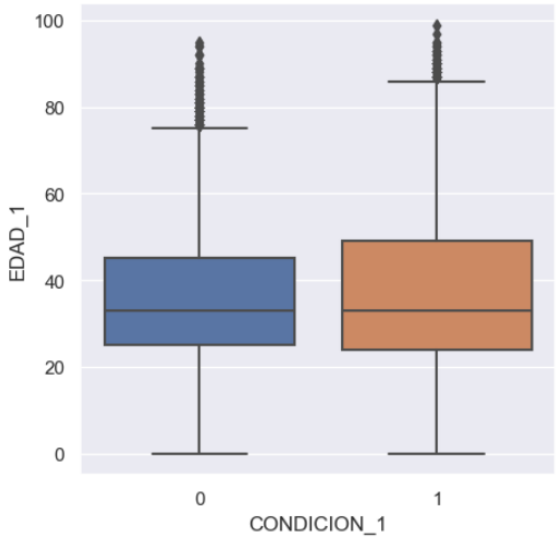
Como se visualiza en la Figura 16 se puede evidenciar que cuando existe por lo menos un vehículo existen más accidentes como también más fallecimientos.

**Figura 16 Suma de Vehículos**



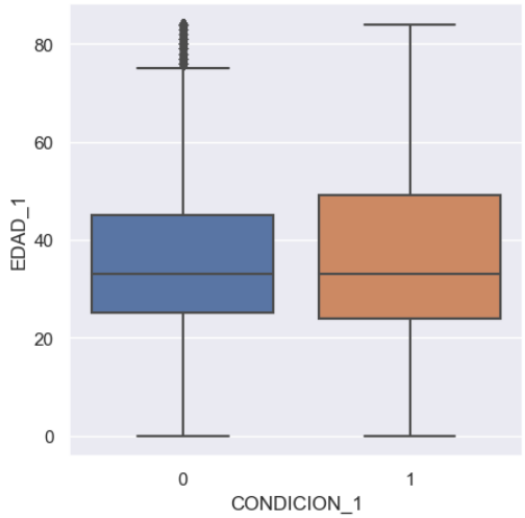
***4.2.2.3.12 Edad.*** Para esta variable se implementó un análisis primero de los outliers dado que se detectó valores fuera de lo común presentando la siguiente Figura 17.

**Figura 17 Variable Edad**




Se pudo detectar que existía valores mayores a 83 años que representaban menos del 1% de la dataset, en este caso imputamos el percentil 99 para evitar que existan outliers en función de este campo, como se muestra en la Figura 18.

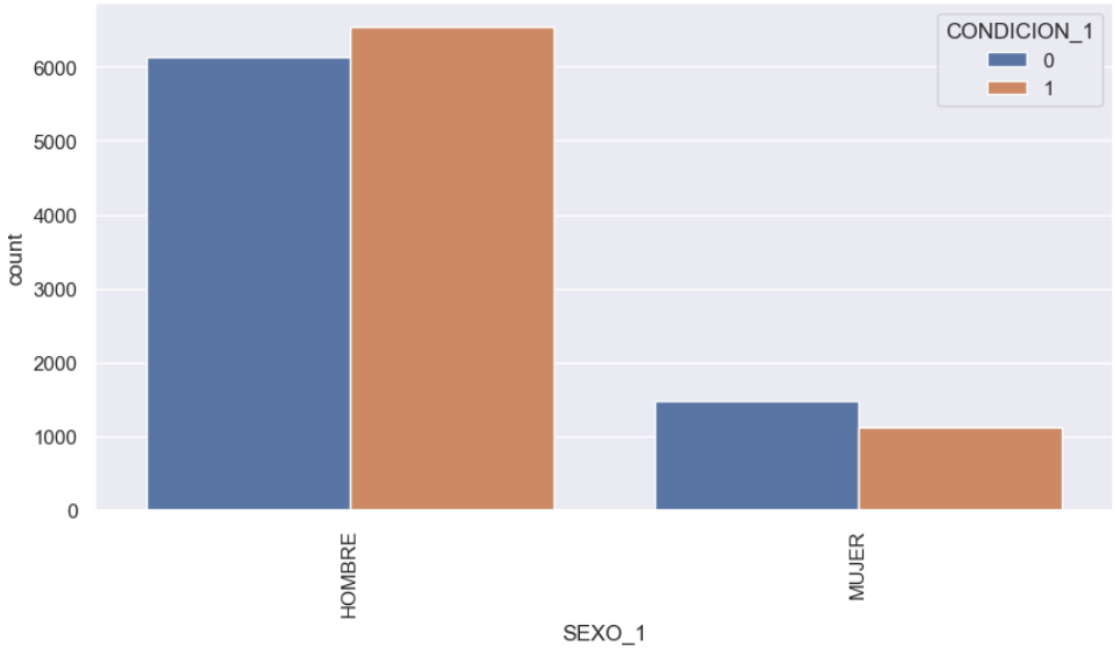
**Figura 18 Variable Edad Sin Outliers**



***4.2.2.3.13 Sexo.*** Se realizo un gráfico de distribución de la variable sexo en función de la variable de estudio en este caso la condición.

Como se pude visualizar en la Figura 19, se evidencia que el sexo masculino es el que tiene mayor número de fallecimiento como también accidentes de tránsito.

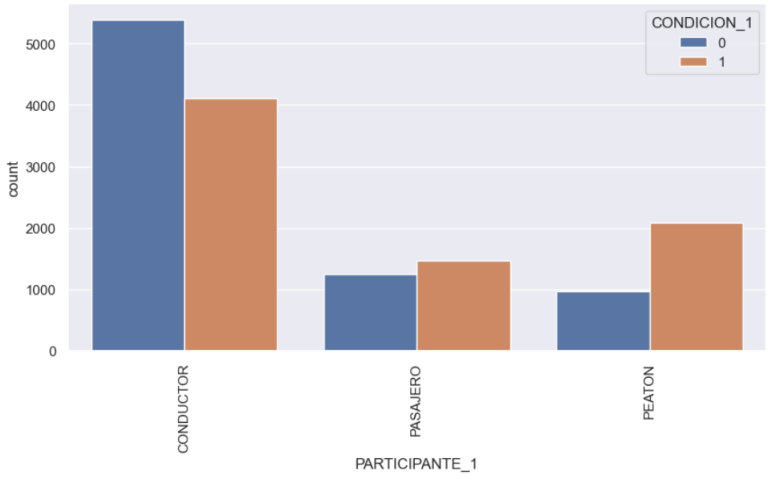
**Figura 19 Variable Sexo**




***4.2.2.3.14 Participante.*** Se realizo un gráfico de distribución de la variable participante en función de la variable de estudio en este caso la condición.

Como se puede visualizar en la Figura 20, se evidencia que los conductores son los que más accidentes de tránsito tiene, sin embargo, podemos ver un volumen parejo entre pasajeros y peatones.

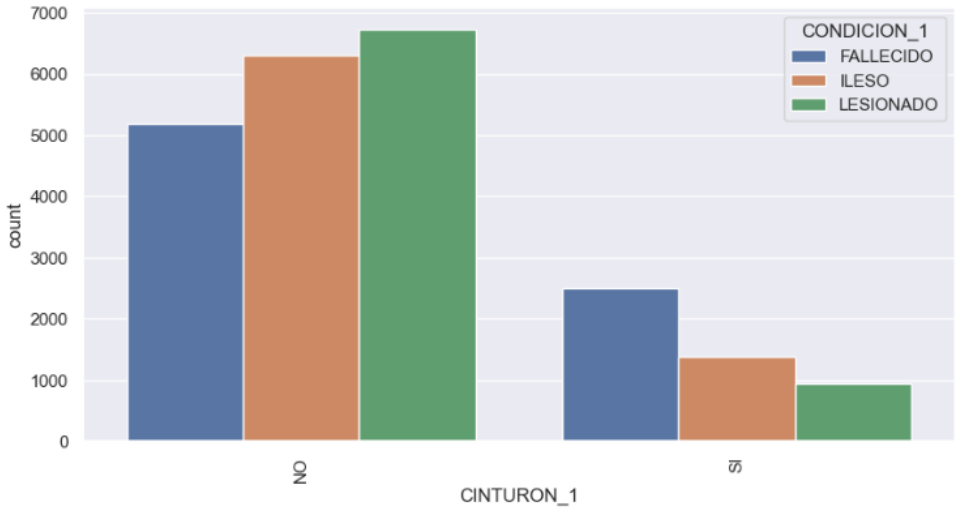
**Figura 20 Variable Participante**




***4.2.2.3.15 Cinturón.*** Se realizo un gráfico de distribución de la variable cinturón en función de la variable de estudio en este caso la condición.

Como se visualiza en la Figura 21, se evidencia la importancia del uso del cinturón de seguridad, este accesorio puede ayudar en 50% que no exista mortalidad en accidentes de transito

**Figura 21 Variable Cinturón**



### *4.2.3 Preparación de la Data*

**4.2.3.1 Limpieza de Variables.** Para la implementación de este proceso hemos identificado algunas columnas que son repetidas dado que son la codificación, propia ya sea propuesto por la agencia nacional de tránsito como también por el INEC.

A continuación, mostraremos en la Figura 22 donde se evidencia la codificación antes descrita.

**Figura 22 Codificación**

| DPA_1 | PROVINCIA | PERIODO_1 | PERIODO_2 | DIA_1 | DIA_2 | MES_1 | MES_2 | CODIGO_CAUSA | CAUSA_PROBABLE |
|---|---|---|---|---|---|---|---|---|---|
| 9 | GUAYAS | DE 23H00 A 23H59 | 23 | JUEVES | 4 | NOVIEMBRE | 11 | C14 | CONDUCIR DESATENTO A LAS CONDICIONES DE TRANSI... |
| 17 | PICHINCHA | DE 00H00 A 00H59 | 0 | JUEVES | 4 | NOVIEMBRE | 11 | C09 | CONDUCIR VEHICULO SUPERANDO LOS LIMITES MAXIMO... |
| 18 | TUNGURAHUA | DE 11H00 A 11H59 | 11 | LUNES | 1 | DICIEMBRE | 12 | C14 | CONDUCIR DESATENTO A LAS CONDICIONES DE TRANSI... |
| 9 | GUAYAS | DE 19H00 A 19H59 | 19 | JUEVES | 4 | NOVIEMBRE | 11 | C23 | NO RESPETAR LAS SENIALES REGLAMENTARIAS DE TRA... |
| 14 | MORONA SANTIAGO | DE 17H00 A 17H59 | 17 | SABADO | 6 | ABRIL | 4 | C06 | CONDUCE BAJO LA INFLUENCIA DE ALCOHOL, SUSTANC... |

Dado este escenario se creó un subset sin repetir variables resultando un dataset de 15267 registros con 31 columnas incluido la clase target., como se muestra en Figura 23.

**Figura 23 Subset**

| | DPA_1 | PROVINCIA | PERIODO_1 | PERIODO_2 | DIA_1 | DIA_2 | MES_1 | MES_2 | CODIGO_CAUSA | CAUSA_PROBABLE |
|---|---|---|---|---|---|---|---|---|---|---|
| **21440** | 9 | GUAYAS | DE 22H00 A 22H59 | 22 | VIERNES | 5 | SEPTIEMBRE | 9 | C14 | CONDUCIR DESATENTO A LAS CONDICIONES DE TRANSI... |
| **23256** | 8 | ESMERALDAS | DE 16H00 A 16H59 | 16 | DOMINGO | 7 | OCTUBRE | 10 | C14 | CONDUCIR DESATENTO A LAS CONDICIONES DE TRANSI... |
| **94854** | 5 | COTOPAXI | DE 19H00 A 19H59 | 19 | SABADO | 6 | DICIEMBRE | 12 | C14 | CONDUCIR DESATENTO A LAS CONDICIONES DE TRANSI... |
| **48726** | 17 | PICHINCHA | DE 01H00 A 01H59 | 1 | SABADO | 6 | OCTUBRE | 10 | C14 | CONDUCIR DESATENTO A LAS CONDICIONES DE TRANSI... |
| **94328** | 17 | PICHINCHA | DE 19H00 A 19H59 | 19 | VIERNES | 5 | DICIEMBRE | 12 | C16 | NO TRANSITAR POR LAS ACERAS O ZONAS DE SEGURID... |

Sin embargo, podemos ver que existen variables como zona, tipo de siniestro que son nominales Figura 24, estas características deben ser transformadas a variables decimales

**Figura 24 Datatypes**

```
PROVINCIA                        object
ZONA                             object
PERIODO_2                         int64
DIA_2                             int64
MES_2                             int64
FERIADO                          object
CODIGO_CAUSA                     object
TIPO_DE_SINIESTRO                object
TIPO_DE_VEHICULO_1               object
SERVICIO_1                       object
SEXO_1                           object
PARTICIPANTE_1                   object
CINTURON_1                       object
CASCO_1                          object
ANIO                              int64
AUTOMOVIL                         int64
BICICLETA                         int64
BUS                               int64
CAMION                            int64
CAMIONETA                         int64
EMERGENCIAS                       int64
ESPECIAL                          int64
FURGONETA                         int64
MOTOCICLETA                       int64
NO_IDENTIFICADO                   int64
SCOOTER_ELECTRICO                 int64
TRICIMOTO                         int64
VEHICULO_DEPORTIVO_UTILITARIO     int64
SUMA_DE_VEHICULOS                 int64
EDAD_1                          float64
CONDICION_1                       int64
dtype: object
```

Para el tratamiento de los datos lo primero que hicimos fue separar la data entre la variable objetivo, de las variables predictoras como se muestra en la Figura 25.

**Figura 25 Variables Predictoras - Objetivo**

```
# Preprocesamiento de datos
X = df_limpio.drop("CONDICION_1", axis=1)  # Variables predictoras
y = df_limpio["CONDICION_1"]  # Variable objetivo
```

A continuación, transformamos la variables predictoras nominales u tipo Object para tener todas las variables aptas para el estudio es decir que sean tipo int64 como se muestra en la Figura 26.

**Figura 26 Codificar Variables Categóricas**

```
# Codificar variables categóricas
categorical_features = ["PROVINCIA", "ZONA","FERIADO","CODIGO_CAUSA", "TIPO_DE_SINIESTRO", "TIPO_DE_VEHICULO_1", "SERVICIO_1", "S
label_encoders = {}
for feature in categorical_features:
    label_encoders[feature] = LabelEncoder()
    X[feature] = label_encoders[feature].fit_transform(X[feature])
```

Como se puede evidenciar estamos utilizando la librería “LabelEncoder” la cual nos permite codificar las variables categóricas.

Luego lo que se realizo fue normalizar las variables enteras de manera que podamos tener de forma equilibrada todas las variables y no exista sesgo en el dataset de estudio, como se muestra en la Figura 27.

**Figura 27 Normalización variables numéricas**

```
# Normalizar variables numéricas
numerical_features = ["ANIO",'AUTOMOVIL',"PERIODO_2","DIA_2","MES_2",'BICICLETA','BUS','CAMION','CAMIONETA','EMERGENCIAS','ESPECI
scaler = StandardScaler()
X[numerical_features] = scaler.fit_transform(X[numerical_features])
```

Como podemos ver utilizamos la librería “StandardScaler” la cual nos permite normalizar las variables numéricas.

Para terminar este proceso visualizamos como quedo el dataset final casi listo para poder generar los modelos correspondientes, como se muestra en la Figura 28

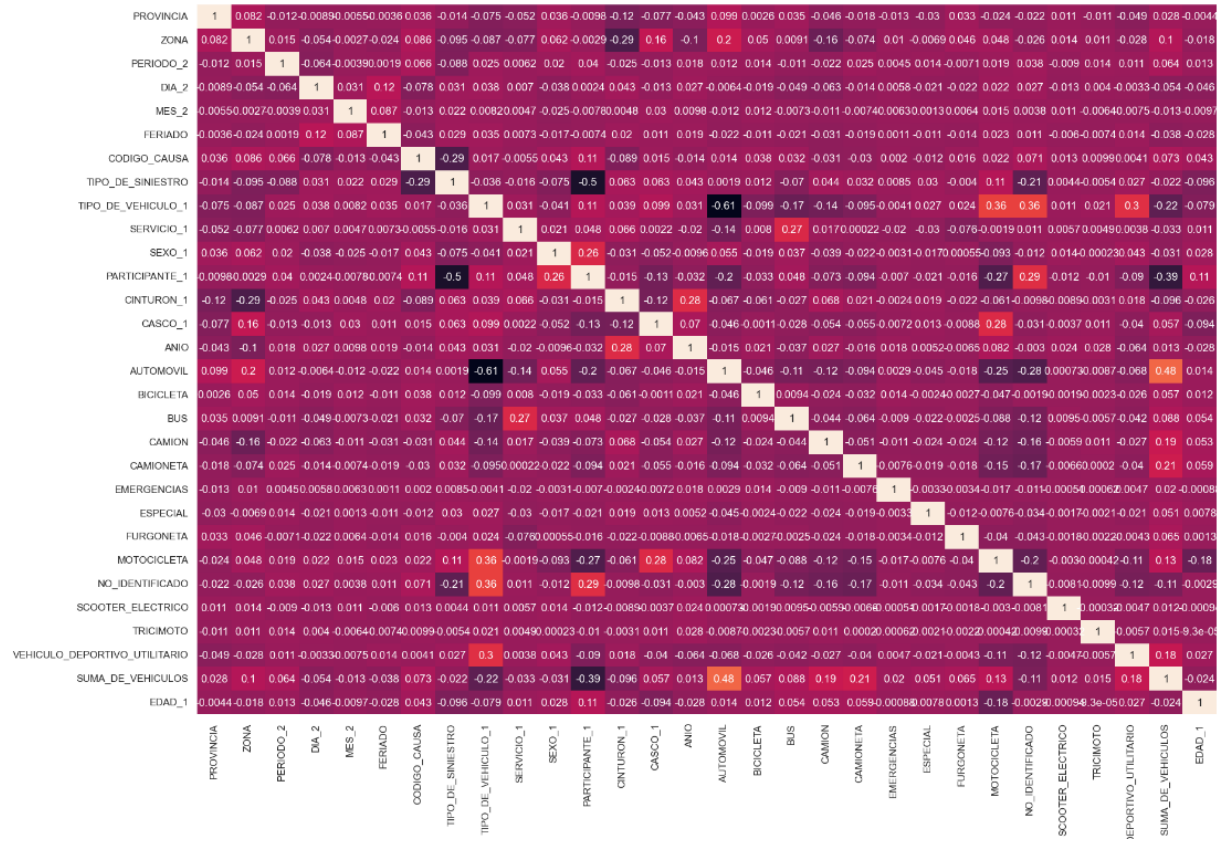
**Figura 28 Dataset Transformado y Normalizado**

| | PROVINCIA | ZONA | PERIODO_2 | DIA_2 | MES_2 | FERIADO | CODIGO_CAUSA | TIPO_DE_SINIESTRO | TIPO_DE_VEHICULO_1 | SERVICIO_1 | ... | EMER |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **102642** | 18 | 1 | 0.667802 | -1.179792 | -0.467092 | 0 | 21 | 4 | 0 | 4 | ... | |
| **82312** | 18 | 1 | 0.371202 | -0.204358 | -1.325502 | 0 | 5 | 7 | 0 | 4 | ... | |
| **39057** | 13 | 0 | -1.111799 | 1.258794 | -0.467092 | 1 | 21 | 7 | 4 | 4 | ... | |
| **54038** | 9 | 1 | -0.666898 | -1.179792 | 1.535864 | 1 | 5 | 6 | 0 | 4 | ... | |
| **11025** | 22 | 1 | 0.667802 | 0.771077 | -0.467092 | 0 | 1 | 4 | 0 | 4 | ... | |
| **...** | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... |
| **139114** | 18 | 1 | 0.074602 | 0.771077 | 1.535864 | 1 | 14 | 1 | 8 | 4 | ... | |
| **139123** | 0 | 1 | 0.519502 | 0.771077 | 1.535864 | 1 | 12 | 4 | 8 | 4 | ... | |
| **139130** | 18 | 1 | 0.816102 | 0.771077 | 1.535864 | 1 | 19 | 9 | 0 | 4 | ... | |
| **139138** | 18 | 1 | 0.964402 | 0.771077 | 1.535864 | 1 | 24 | 1 | 9 | 4 | ... | |
| **139146** | 18 | 0 | 1.261002 | 0.771077 | 1.535864 | 1 | 8 | 7 | 9 | 4 | ... | |

15267 rows × 30 columns

**4.2.3.2 Matriz de Correlación.** La matriz de correlación la implementamos para poder seleccionar las variables que no tengan relación entre sí, la teoría establece que existe más relación tanto positiva como negativa entre más cerca al 1 exista, según la Figura 29 se puede visualizar que no existe una relación fuerte es de decir ninguna relación super el 0.75 o el -0.75 que en este caso sería el parámetro que no defina dicha relación fuerte por esta razón no existe razón porque eliminar alguna de las variables que se encuentran dentro del dataset seleccionado.

**Figura 29 Matriz de Correlación**




### *4.2.4 Modelado*

Para el estudio planteado se van a implementar tres modelos de aprendizaje supervisado, los mismos que se desarrollaron a continuación:

**4.2.4.1 Clasificador Bayesiano.** El primer paso que se realizó para la implementación de este modelo es separar el target objetivo y las variables dependientes, luego dividimos el dataset ya restructurado y limpio, esta división se realizó en una proporción de 80/20 en el 80% será nuestra data de entrenamiento y el 20% restante corresponderá a la data de evaluación o test, como se muestra en la Figura 30.

**Figura 30 División Dataset**

```
# Dividir el dataset en conjuntos de entrenamiento y prueba
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
```

A continuación, usaremos la data de entrenamiento y generaremos el modelo como se visualizar en la Figura 31, sin embargo, hay que tomar en cuenta que dado que nuestra variable objetivo es multinomial se usara el clasificar bayesiano multinomial.

**Figura 31 Clasificador Bayesiano**

```
# Entrenar el clasificador bayesiano
gnb = GaussianNB()
# Entrenar el modelo
cb=gnb.fit(X_train, y_train)
```

**4.2.4.2 Máquinas de Soporte Vectorial.** Usando la misma data de entrenamiento implementamos el modelo como se muestra en la Figura 32

**Figura 32 Máquinas de Soporte Vectorial**

```
#Maquinas de Soporte
# Entrenar la SVM
svm = SVC(kernel='linear', probability=True)
msv=svm.fit(X_train, y_train)
```

**4.2.4.3 Regresión Logística.** Usando la misma data de entrenamiento implementamos el modelo como se muestra en la figura 33.

**Figura 33 Regresión Logística**

```
## Regresion logistica multinomial
# Entrenar el clasificador de regresión logística
lr = LogisticRegression(solver='lbfgs', max_iter=20000)
rl = lr.fit(X_train, y_train)
```

Como detalle importante podemos que al momento de implementar cada uno de los modelos, son ajustados con la función fit.

### *4.2.5 Evaluación de los modelos*

Para la implementación de la evaluación de cada uno de los modelos se han planteado estadísticos como:

- Accuracy_score, Precision_score
- Curva ROC
- Matriz de Confusión

**4.2.5.1 Clasificador Bayesiano.** En función del modelo generado se procede a probar el modelo con la data que corresponde al 20% como se mostrara en la Figura 34.

**Figura 34 Evaluación Clasificador Bayesiano**

```
# Realizar predicciones en el conjunto de prueba
y_pred_cb = cb.predict(X_test)
y_prob_cb = cb.predict_proba(X_test)
```

***4.2.5.1.1 Accuracy_score, Precision_score.*** Dado las predicciones implementadas en el conjunto de datos de pruebas generamos los estadísticos de evaluación, como se muestra en la Figura 35

**Figura 35 Evaluación Métricas Clasificador Bayesiano**

```
accuracy = accuracy_score(y_test, y_pred_cb)
precision = precision_score(y_test, y_pred_cb, average='macro')
recall = recall_score(y_test, y_pred_cb, average='macro')
```

```
# Imprimir las métricas de rendimiento
print('Exactitud:', accuracy)
print('Precisión:', precision)
print('Sensibilidad:', recall)
```

```
Exactitud: 0.6830386378519974
Precisión: 0.7244108929467369
Sensibilidad: 0.6826154083680717
```

#### *4.2.5.1.2 Curva Roc*

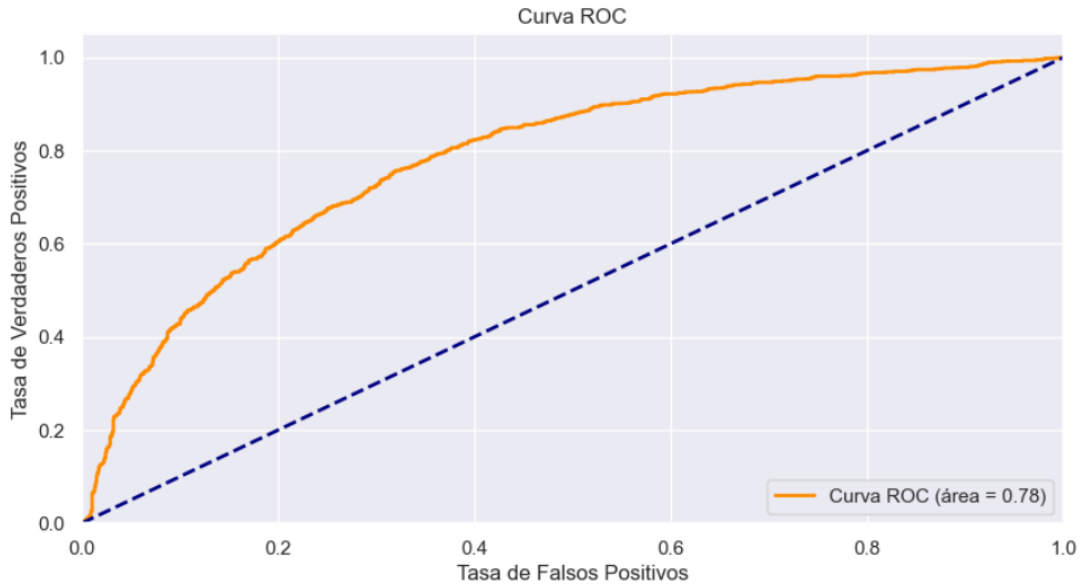
**Figura 36 Evaluación Curva Roc Clasificador Bayesiano V1**

```
#Curva ROC
# Calcular la probabilidad de las predicciones
y_score = cb.predict_proba(X_test)

# Calcular la curva ROC
fpr, tpr, thresholds = roc_curve(y_test, y_score[:,1], pos_label=1)
roc_auc = auc(fpr, tpr)

# Graficar la curva ROC
plt.figure()
plt.plot(fpr, tpr, color='darkorange', lw=2, label='Curva ROC (área = %0.2f)' % roc_auc)
plt.plot([0, 1], [0, 1], color='navy', lw=2, linestyle='--')
plt.xlim([0.0, 1.0])
plt.ylim([0.0, 1.05])
plt.xlabel('Tasa de Falsos Positivos')
plt.ylabel('Tasa de Verdaderos Positivos')
plt.title('Curva ROC')
plt.legend(loc="lower right")
plt.show()
```

**Figura 37 Evaluación Curva Roc Clasificador Bayesiano V2**



#### *4.2.5.1.3 Matriz de Confusión*

**Figura 38 Evaluación Matriz de Confusión Clasificador Bayesiano**

```
confusion_matrix(y_test, y_pred_cb)
```

```
array([[ 712,  812],
       [ 156, 1374]], dtype=int64)
```

```
(712+1374)/(712+1374+812+156)
```

```
0.6830386378519974
```

**4.2.5.2 Máquinas de Soporte Vectorial.** En función del modelo generado se procede a probar el modelo con la data que corresponde al 20% como se mostrara en la Figura 39.

**Figura 39 Evaluación Maquinas de Soporte Vectorial**

```
# Hacer predicciones en el conjunto de prueba
y_pred_msv = msv.predict(X_test)
y_prob_msv = msv.predict_proba(X_test)
```

***4.2.5.2.1 Accuracy_score, Precision_score.*** Dado las predicciones implementadas en el conjunto de datos de pruebas generamos los estadísticos de evaluación, como se muestra en la Figura 40

**Figura 40 Evaluación Métricas Maquinas de Soporte Vectorial**

```
# Calcular la precisión (accuracy) de las predicciones
accuracy = accuracy_score(y_test, y_pred_msv)
precision = precision_score(y_test, y_pred_msv, average='macro')
recall = recall_score(y_test, y_pred_msv, average='macro')
```

```
# Imprimir las métricas de rendimiento
print('Exactitud:', accuracy)
print('Precisión:', precision)
print('Sensibilidad:', recall)
```

```
Exactitud: 0.7321545514079896
Precisión: 0.7331094003604522
Sensibilidad: 0.7320904739848695
```

#### *4.2.5.2.2 Curva Roc*

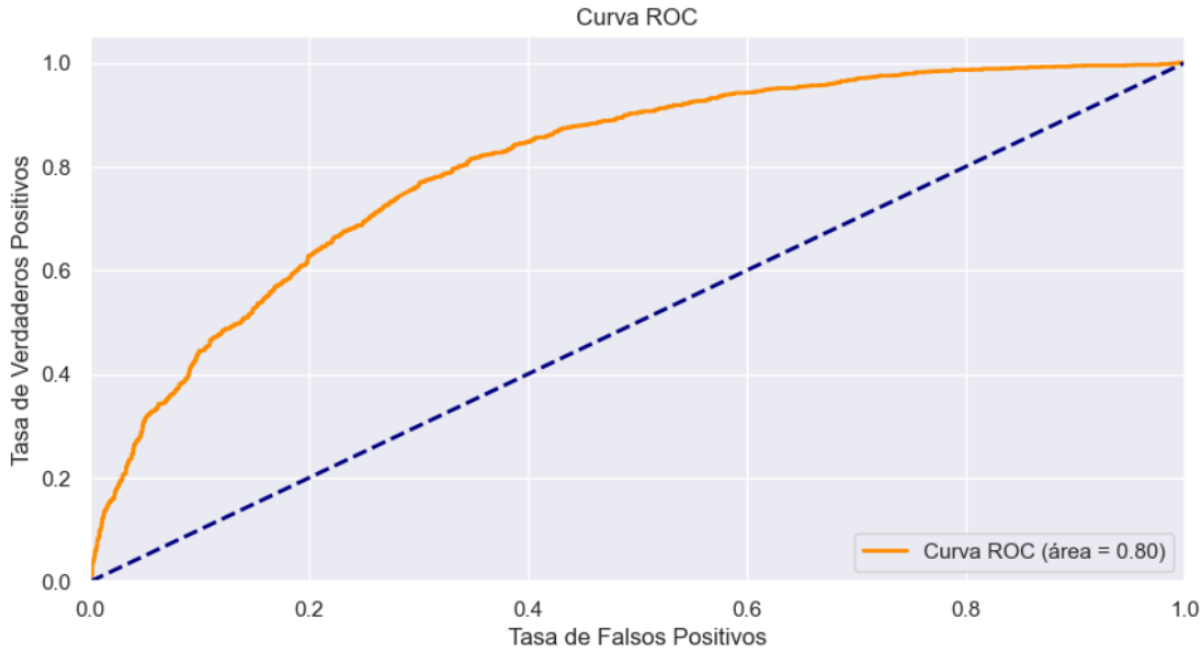
**Figura 41 Evaluación Curva Roc Maquinas de Soporte Vectorial V1**

```
# Calcular probabilidad de predicción para cada clase
y_score = msv.predict_proba(X_test)

# Calcular la curva ROC
fpr, tpr, thresholds = roc_curve(y_test, y_score[:,1], pos_label=1)
roc_auc = auc(fpr, tpr)

# Graficar la curva ROC
plt.figure()
plt.plot(fpr, tpr, color='darkorange', lw=2, label='Curva ROC (área = %0.2f)' % roc_auc)
plt.plot([0, 1], [0, 1], color='navy', lw=2, linestyle='--')
plt.xlim([0.0, 1.0])
plt.ylim([0.0, 1.05])
plt.xlabel('Tasa de Falsos Positivos')
plt.ylabel('Tasa de Verdaderos Positivos')
plt.title('Curva ROC')
plt.legend(loc="lower right")
plt.show()
```

**Figura 42 Evaluación Curva Roc Maquinas de Soporte Vectorial V2**



#### *4.2.5.2.3 Matriz de Confusión*

**Figura 43 Evaluación Matriz de Confusión Maquinas de Soporte Vectorial**

```
confusion_matrix(y_test, y_pred_msv)
```

```
array([[1066,  458],
       [ 360, 1170]], dtype=int64)
```

```
(1066+1170)/(1066+1170+458+360)
```

```
0.7321545514079896
```

**4.2.5.3 Regresión Logística.** En función del modelo generado se procede a probar el modelo con la data que corresponde al 20% como se mostrara en la Figura 43.

**Figura 44 Evaluación Regresión Logística**

```
# Hacer predicciones en el conjunto de prueba
y_pred_rl = rl.predict(X_test)
y_prob_rl = rl.predict_proba(X_test)
```

***4.2.5.3.1 Accuracy_score, Precision_score.*** Dado las predicciones implementadas en el conjunto de datos de pruebas generamos los estadísticos de evaluación, como se muestra en la Figura 44

**Figura 45 Evaluación Métricas Regresión Logística**

```
# Calcular la precisión (accuracy) de las predicciones
accuracy = accuracy_score(y_test, y_pred_rl)
precision = precision_score(y_test, y_pred_rl, average='macro')
recall = recall_score(y_test, y_pred_rl, average='macro')
```

```
# Imprimir las métricas de rendimiento
print('Exactitud:', accuracy)
print('Precisión:', precision)
print('Sensibilidad:', recall)
```

```
Exactitud: 0.7288801571709234
Precisión: 0.7290845256657714
Sensibilidad: 0.7288495188101487
```

#### *4.2.5.3.2 Curva Roc*

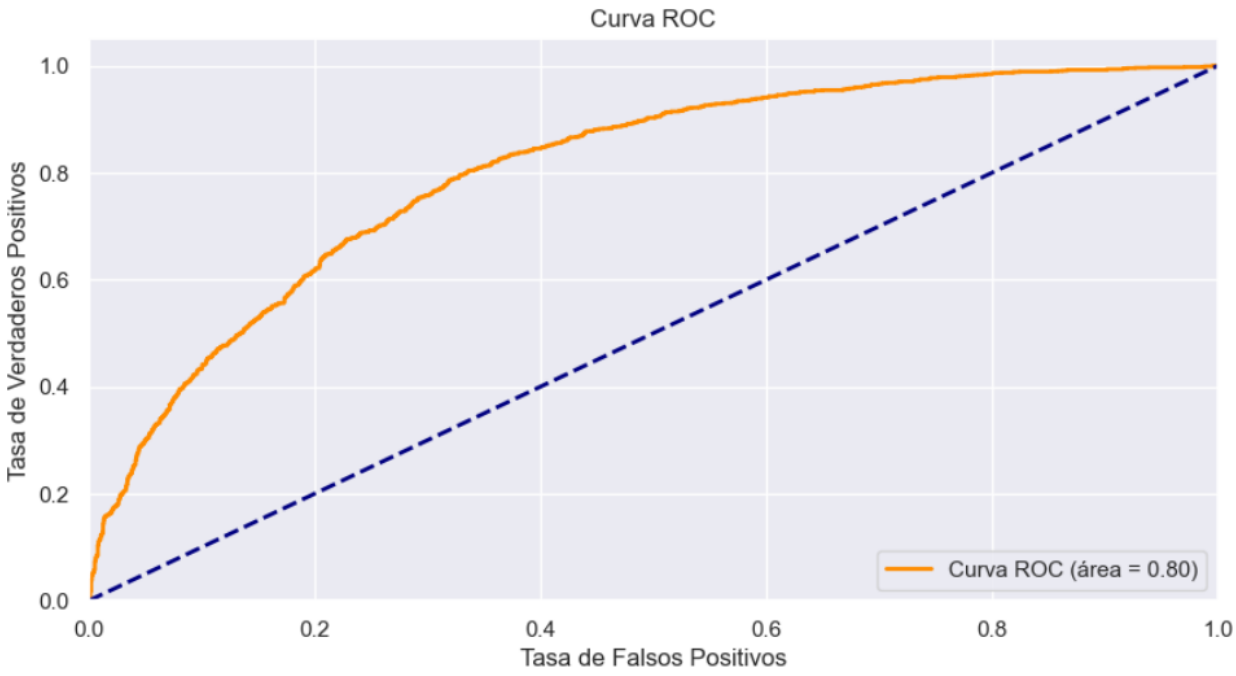
**Figura 46 Evaluación Curva Roc Regresión Logística V1**

```
# Calcular probabilidad de predicción para cada clase
y_score = rl.predict_proba(X_test)

# Calcular la curva ROC
fpr, tpr, thresholds = roc_curve(y_test, y_score[:,1], pos_label=1)
roc_auc = auc(fpr, tpr)

# Graficar la curva ROC
plt.figure()
plt.plot(fpr, tpr, color='darkorange', lw=2, label='Curva ROC (área = %0.2f)' % roc_auc)
plt.plot([0, 1], [0, 1], color='navy', lw=2, linestyle='--')
plt.xlim([0.0, 1.0])
plt.ylim([0.0, 1.05])
plt.xlabel('Tasa de Falsos Positivos')
plt.ylabel('Tasa de Verdaderos Positivos')
plt.title('Curva ROC')
plt.legend(loc="lower right")
plt.show()
```

**Figura 47 Evaluación Curva Roc Regresión Logística V2**



#### *4.2.5.3.3 Matriz de Confusión*

**Figura 48 Evaluación Matriz de Confusión Regresión Logística**

```
confusion_matrix(y_test, y_pred_rl)
```

```
array([[1087,  437],
       [ 391, 1139]], dtype=int64)
```

```
(1087+1139)/(1087+1139+437+391)
```

```
0.7288801571709234
```

### *4.2.6 Despliegue*

Para el despliegue, se creó un dataset basado en datos del 2023 en donde se implementó el modelo de máquinas de soporte vectorial dado que es el modelo más preciso, como se muestra en la Figura 49.

**Figura 49 Dataset para Despliegue**

```
#Dataset de despliegue
filtro_val = (df["ANIO"] > 2022) & (df["EDAD_1"] >= 0)
df_val=df[filtro_val]
```

| | index | ID | ANIO | SINIESTROS | LESIONADOS | FALLECIDOS | ENTE_DE_CONTROL | LATITUD_Y | LONGITUD_X | DPA_1 | ... | TRICIMOTO | VEHICUL |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **0** | 139156 | 139157 | 2023 | MCU00001012023 | 1 | 0 | EMPRESA PUBLICA DE MOVILIDAD, TRANSITO Y TRANS... | -2.900646 | -78.984631 | 1 | ... | 0 | |
| **1** | 139158 | 139159 | 2023 | DMQ00002012023 | 0 | 0 | AGENCIA METROPOLITANA DE TRANSITO DE QUITO - AMT | -0.284215 | -78.516339 | 17 | ... | 0 | |
| **2** | 139161 | 139162 | 2023 | MAM00005012023 | 3 | 0 | DIRECCION DE TRANSITO, TRASPORTE TERRESTRE Y S... | -1.252839 | -78.629924 | 18 | ... | 0 | |
| **3** | 139162 | 139163 | 2023 | DMQ00006012023 | 6 | 0 | AGENCIA METROPOLITANA DE TRANSITO DE QUITO - AMT | -0.057577 | -78.464477 | 17 | ... | 0 | |
| **4** | 139164 | 139165 | 2023 | PNE00001012023 | 0 | 1 | POLICIA NACIONAL DEL ECUADOR | -0.901179 | -80.237069 | 13 | ... | 0 | |
| **...** | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | |
| **2187** | 142577 | 142578 | 2023 | MPO00671022023 | 0 | 0 | EMPRESA PUBLICA MUNICIPAL DE TRANSITO, TRANSPO... | -1.063441 | -80.456982 | 13 | ... | 0 | |
| **2188** | 142578 | 142579 | 2023 | CTE00485022023 | 1 | 0 | COMISION DE TRANSITO DEL ECUADOR - CTE | -1.006694 | -80.686833 | 13 | ... | 0 | |
| **2189** | 142579 | 142580 | 2023 | DMQ00672022023 | 0 | 0 | AGENCIA METROPOLITANA DE TRANSITO DE QUITO - AMT | -0.187918 | -78.491713 | 17 | ... | 0 | |
| | | | | | | | AGENCIA DE TRANSITO Y | | | | | | |

**4.2.6.1 Crear Modelo.** El primer paso del despliegue es guardar el modelo de máquinas de soporte vectorial como se muestra en la Figura 50

**Figura 50 Crear Modelo**

```
# guarda modelo
from joblib import dump, load
dump(msv, 'SVM_Bank.joblib')
```

**4.2.6.2 Cargamos Modelo.** Cargamos el modelo guardado anteriormente como se muestra en la Figura 51

**Figura 51 Cargar Modelo**

```
# cargar modelo
loaded_model = load('SVM_Bank.joblib')
```

**4.2.6.3 Preparación de la Data.** Realizamos el mismo paso que hicimos con la data de aprendizaje como se muestra en la Figura 52

**Figura 52 Preparación Data Despliegue**

```
df_val = df_val.loc[:,["PROVINCIA", "ZONA", "PERIODO_2","DIA_2","MES_2","FERIADO","CODIGO_CAUSA", "TIPO_DE_SINIESTRO", "TIPO_DE_V
df_val
```

```
# Preprocesamiento de datos
X = df_val.drop("CONDICION_1", axis=1)  # Variables predictoras
y = df_val["CONDICION_1"]  # Variable objetivo
```

```
# Codificar variables categóricas
categorical_features = ["PROVINCIA", "ZONA","FERIADO","CODIGO_CAUSA", "TIPO_DE_SINIESTRO", "TIPO_DE_VEHICULO_1", "SERVICIO_1", "S
label_encoders = {}
for feature in categorical_features:
    label_encoders[feature] = LabelEncoder()
    X[feature] = label_encoders[feature].fit_transform(X[feature])
```

```
# Normalizar variables numéricas , "PERIODO_1","DIA_1","MES_1"
numerical_features = ["ANIO",'AUTOMOVIL',"PERIODO_2","DIA_2","MES_2",'BICICLETA','BUS','CAMION','CAMIONETA','EMERGENCIAS','ESPECI
scaler = StandardScaler()
X[numerical_features] = scaler.fit_transform(X[numerical_features])
```

**4.2.6.4 Usando el Modelo.** Mediante el modelo cargado anteriormente y los datos ya preparados usamos el modelo y predecimos con la nuevo dataset, como se muestra en la Figura 53

**Figura 53 Uso del Modelo**

```
# prediccion
y_pred_msv = loaded_model.predict(X)
y_prob_msv = loaded_model.predict_proba(X)
```

```
# transformo a dataframe la probabilidad
df_bin_val = pd.DataFrame(y_prob_msv[:, 1], columns=['Probabilidad'])
```

```
y_pred_msv
```

```
array([1, 0, 1, ..., 0, 0, 0], dtype=int64)
```

**4.2.6.5 Reporte Uso del Modelo.** Como se puede visualizar en la Figura 54 mediante el uso del modelo generado se habrían podido evitar 1041 fallecidos de 2152 que fue el tamaño del dataset que se usó en esta fase.

**Figura 54 Uso de Modelo**

```
from collections import Counter

# Contar la frecuencia de cada elemento en el arreglo
frecuencia = Counter(y_pred_msv)

# Imprimir el resultado
for elemento, cantidad in frecuencia.items():
    print("Elemento:", elemento, "- Cantidad:", cantidad)
```

```
Elemento: 1 - Cantidad: 1041
Elemento: 0 - Cantidad: 1151
```

## 4.3 Validación de Objetivos planteados

- Recopilar datos históricos de accidentes de tráfico en Ecuador
    - Se ha obtenido los datos históricos de accidentes de tránsito en Ecuador basado a los datos administrados por la agencia nacional de tránsito desde los años 2017 hasta el 2022, partiendo de este hecho se realizó un proceso de entendiendo de la

misma como también la limpieza correspondiente, de esta manera se ha cumplido con el objetivo planteado.

- Evaluar y seleccionar las variables más relevantes que influyen en la mortalidad en accidentes de tráfico en Ecuador mediante técnicas de minería de datos.
    - Para el proceso de evaluación de variables se identificó inicialmente los tipos de dato
    - Se elimino las variables que se repetían dado que venían precodificadas por la agencia nacional de transito
    - Se implemento la matriz de correlación para identificar alguna relación fuerte entre las variables

Partiendo de estos puntos se evidencia que se cumplió planteado

- Diseñar e implementar modelos de aprendizaje automático con regresión logística, máquinas de soporte vectorial y clasificador bayesianos que permitan predecir la mortalidad en accidentes de tráfico en Ecuador
    - En el Capítulo IV Modelado, se realizó la implementación de los tres modelos planteados, usando la misma data con la misma distribución de la data 80% de la data de aprendizaje 20% para la data de test o validación del modelo, adicionalmente en el Capítulo VIII (Anexos) se mostrará todo el proceso de que se realizó para la implementación de los modelos propuestos
    - Analizar los resultados obtenidos y mostrar cual es el modelo más indicado para la implementación de un estudio como este.
    - En el menú 4.2.5 Evaluación de Modelos donde se evidencio en todos los estadísticos planteados que el modelo más indicado para realizar un análisis de los

accidentes de tránsito en el ecuador es por medio del uso de máquinas de soporte vectorial dado que es el más preciso como se muestra en la Figura 55.

**Figura 55 Modelo más Preciso SVM**

```
# Calcular la precisión (accuracy) de las predicciones
accuracy = accuracy_score(y_test, y_pred_msv)
precision = precision_score(y_test, y_pred_msv, average='macro')
recall = recall_score(y_test, y_pred_msv, average='macro')
```

```
# Imprimir las métricas de rendimiento
print('Exactitud:', accuracy)
print('Precisión:', precision)
print('Sensibilidad:', recall)
```

```
Exactitud: 0.7321545514079896
Precisión: 0.7331094003604522
Sensibilidad: 0.7320904739848695
```

Es decir, tiene una exactitud casi del 74% dado esto planteamos que las máquinas de soporte vectorial son más adecuadas para el estudio planteado.

# 5. Capitulo V Conclusiones y Recomendaciones

## 5.1 Conclusiones

Podemos llegar a las siguientes conclusiones

- En base a la data que obtuvimos se puede podemos decir que estuvo bastante estructurada y apta para la implementación de los tres modelos propuestos.
- Me pareció muy interesante conocer que en función de un análisis de comportamiento en este caso de la variable PERIODO_2 se pudo determinar que el rango horario que más accidentes tenemos en Ecuador es 19:00 a 20:00.
- El principal causante de accidentes de tránsito es “conducir desatento a las condiciones de tránsito (celular, pantallas de video, comida, maquillaje o cualquier otro elemento distractor)”.
- El día de la semana con más fallecimientos es el domingo, además podemos decir que la mayoría de los fallecimientos está en rango de edad de entre los 22 y 42 años.
- Se pudo notar una característica especial, los accidentes de tránsito en zonas urbanas predominan versus la rural, sin embargo, si nos referimos a los fallecimientos es en la zona rural que lidera este parámetro.
- En mi opinión creo que la metodología cris-dm es la mejor opción para este tipo de estudio dado que me permitió tener estructurado cada uno de los pasos tanto de limpieza análisis y comprensión de la data, sin dejar de lado que me permitió volver a fases anteriores realizar algunos ajustes hasta construir de la mejor manera la data como los modelos.

- Este estudio permitió establecer que para realizar el estudio de accidentes de tránsito el modelo más preciso es la máquina de soporte vectorial, hecho que la matriz de confusión reafirma esta conclusión.

## 5.2 Recomendaciones

- Es necesario realizar campañas de concientización para que los conductores y peatones no usen dispositivos móviles cuando este manejando como al momento de cruzar una calle dado que como hemos visto dentro de este estudio es la primera causa de mortalidad en el Ecuador.
- Es importante que se realice campañas de concientización a los conductores cuando viajes los fines de semana en especial el domingo día donde se evidencia mayor cantidad tanto de accidentes como de fallecimientos.
- Es imprescindible para mi criterio desplegar agentes de tránsito en el rango de 19:00 a 20:00 que permita un número mínimo de accidentes y por ende en número de fallecimientos en este periodo de tiempo.
- Se recomienda usar el algoritmo maquinas de soporte vectorial para escenarios de análisis de accidentes dado el nivel de precisión que ha presentado el presente estudio
- Este estudio puede ser el punto de partida para estudios más específicos, enfocados en el geoposicionamiento de los accidentes, lugares más comunes entre otras

# 6. Capítulo VI Bibliografía

## 6.1 Referencias


Alpaydin, E. (2020). *Introduction to machine learning.* MIT press.

Anaconda. (01 de 01 de 2021). *https://docs.anaconda.com/free/anaconda/getting-started/what-is-distro/*. Obtenido de https://docs.anaconda.com/free/anaconda/getting-started/what-is-distro/: https://docs.anaconda.com/free/anaconda/getting-started/what-is-distro/

Bishop, C. M. (2006). Pattern recognition and machine learning. *New York: springer*, (Vol. 4, No. 4, p. 738).

Campos-Villalta, Y. Y.-B.-G.-A. (2019). Sistema de indicadores de morbilidad y mortalidad por accidentes de tráfico: una revisión sistemática. *Revista de Salud Pública*, 21(6).

developers, S.-l. (01 de 01 de 2021). *https://scikit-learn.org*. Obtenido de https://scikit-learn.org: https://scikit-learn.org/stable/modules/naive_bayes.html

Galán Cortina, V. (2016). Aplicación de la metodología CRISP-DM a un proyecto de minería de datos en el entorno universitario. *Bachelor's thesis*.

Han, J. K. (2012). Data mining concepts and techniques third edition. *University of Illinois at Urbana-Champaign Micheline Kamber Jian Pei Simon Fraser University*.

Harris, C. R. (2020). Array programming with NumPy. *Nature*, 585(7825), 357-362.

Hosmer Jr, D. W. (2013). *Applied logistic regression.* John Wiley & Sons.

Hunter, J. D. (2007). Matplotlib: A 2D graphics environment. *Computing in science & engineering*, 90-95.

INEC. (1 de Julio de 2022). *ecuadorencifras*. Obtenido de ecuadorencifras: https://www.ecuadorencifras.gob.ec/documentos/web-inec/Estadisticas_Economicas/Estadistica%20de%20Transporte/2021/2021_SINIESTROS_PPT.pdf

Kluyver, T. R.-K. (2016). Jupyter Notebooks-a . *Computational Workflows*, Vol. 2016, pp. 87-90.

McKinney, W. (2010). Data structures for statistical computing in python. *In Proceedings of the 9th Python in Science Conference*, (Vol. 445, No. 1, pp. 51-56).

Pedregosa, F. V. (2011). Scikit-learn: Machine learning in Python. *the Journal of machine Learning research*, 12, 2825-2830.

Rahim, M. A. (2021). A deep learning based traffic crash severity prediction framework. *Accident Analysis & Prevention*, 154, 106090.

Salud, O. P. (29 de Septiembre de 2021). *paho.org*. Obtenido de paho.org: https://www.paho.org/es/noticias/29-10-2021-semana-seguridad-vial-oms-presento-plan-mundial-para-reducir-50-muertes

SAP. (1 de 3 de 2023). *sap.com*. Obtenido de sap.com: https://www.sap.com/latinamerica/insights/what-is-data-mining.html

Transito, A. N. (01 de 07 de 2022). *www.ant.gob.ec.* Obtenido de www.ant.gob.ec: https://www.ant.gob.ec/wp-content/uploads/2022/07/Ficha_metodologica_Vf8.3.21.7.22.pdf

VanderPlas, J. (2016). *Python data science handbook: Essential tools for working with data.* O'Reilly Media, Inc.

Virtanen, P. G. (2020). SciPy 1.0: fundamental algorithms for scientific computing in Python. *Nature methods*, 261-272.

Waskom, M. L. (2021). Seaborn: statistical data visualization. *Journal of Open Source Software*, 6(60), 3021.

Wirth, R. &. (2000). CRISP-DM: Towards a standard process model for data mining. *In Proceedings of the 4th international conference on the practical applications of knowledge discovery and data mining*, (Vol. 1, pp. 29-39).

# 7. Capitulo VII Anexos

## 7.1 Código Jupiter Notebook

Estudio comparativo de la precisión de algoritmos de aprendizaje automático, regresión logística, máquinas de soporte y clasificador bayesiano, basado en la implementación de modelo predictivos en función de la mortalidad en accidentes en Ecuador¶

### Objetivos¶

- Aplicar la metodología CRISP - DM sobre un conjunto de datos obtenidos de la Agencia Nacional de Transito
- Realizar un análisis comparativo de la precisión en la implementacion de modelos como: Regresión Lineal , Maquinas de Soporte Vectorial y clasificador Bayesiano

### Implementación de la Metodología CRISP-DM¶

In [1]:

```
#importando librerias
import pandas as pd
from sklearn.model_selection import train_test_split,cross_val_score
from sklearn.naive_bayes import GaussianNB
from sklearn.metrics import accuracy_score, precision_score, recall_score,classification_report,r
oc_curve, auc ,confusion_matrix
from sklearn.utils import resample
from sklearn.preprocessing import LabelEncoder, StandardScaler
from sklearn.svm import SVC
import matplotlib.pyplot as plt
from sklearn.linear_model import LogisticRegression
from scipy.stats import chi2_contingency, f_oneway
import numpy as np
import seaborn as sns
```

### 1. Business Understanding¶

- La información o dataset que vamos a utilizar en este estudio, es de acceso público y se lo puede encontrar en la página de la ant en ítem de https://www.ant.gob.ec/visor-de-siniestralidad-estadisticas/
- La ant administra todo el registro de accidentes del país, el mismo es alimentado por los GADs correspondientes, los mismo son actualizados cada 15 días de cada mes, sin embargo, este estudio se basará en el registro desde el año 2017 al año 2022
- Los objetivos de la organización se enmarcan en su manual, específicamente el mejoramiento de la gestión de la seguridad vial, vehículos más seguros, infraestructura vial más segura, usuarios más seguros, y sistemas de respuesta ante la emergencia.

- El presente estudio además de realizar el análisis comparativo pretende determinar las variables más importantes que ocasionan un accidente de tránsito, más específicamente en la mortalidad.

## 2. Data Understanding¶

### Carga Datos¶

In [2]:
df = pd.read_csv('data.csv', encoding='ISO-8859-1')

### 2.1.- Visualización de Datos¶

In [3]:
#imprimir el dataset
df

Out[3]:

| | ID | ANIO | SINIESTROS | LESIONADOS | FALLECIDOS | ENTE_DE_CONTROL | LATITUD_Y | LONGITUD_X | DPA_1 | PROVINCIA | ... | TRICIMOTO |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 1 | 2017 | DMQ00001012017 | 1 | 0 | AGENCIA METROPOLITANA DE TRANSITO DE QUITO - AMT | -0.083501 | -78.417742 | 17 | PICHINCHA | ... | 0 |
| 1 | 2 | 2017 | ATM00002012017 | 1 | 0 | AGENCIA DE TRANSITO Y MOVILIDAD DE GUAYAQUIL -... | -2.246682 | -79.897754 | 9 | GUAYAS | ... | 0 |
| 2 | 3 | 2017 | PNE00003012017 | 1 | 0 | POLICIA NACIONAL DEL ECUADOR | -0.253881 | -79.217405 | 23 | SANTO DOMINGO DE LOS TSACHILAS | ... | 0 |
| 3 | 4 | 2017 | DMQ00004012017 | 0 | 0 | AGENCIA METROPOLITANA DE TRANSITO DE QUITO - AMT | -0.116059 | -78.464188 | 17 | PICHINCHA | ... | 0 |

### 2.2.- Tipos de dato dentro del Dataset¶

In [4]:
print(df.dtypes)

```
ID                  int64
ANIO                int64
SINIESTROS         object
LESIONADOS          int64
FALLECIDOS          int64
ENTE_DE_CONTROL    object
LATITUD_Y         float64
LONGITUD_X        float64
DPA_1               int64
PROVINCIA          object
DPA_2               int64
CANTON             object
DPA_3               int64
PARROQUIA          object
```

```
DIRECCION                          object
ZONA_PLANIFICACION                 object
ZONA                               object
ID_DE_LA_VIA                       object
NOMBRE_DE_LA_VIA                   object
UBICACION_DE_LA_VIA                object
JERARQUIA_DE_LA_VIA                object
FECHA                              object
HORA                               object
PERIODO_1                          object
PERIODO_2                          int64
DIA_1                              object
DIA_2                              int64
MES_1                              object
MES_2                              int64
FERIADO                            object
CODIGO_CAUSA                       object
CAUSA_PROBABLE                     object
TIPO_DE_SINIESTRO                  object
TIPO_DE_VEHICULO_1                 object
SERVICIO_1                         object
AUTOMOVIL                          int64
BICICLETA                          int64
BUS                                int64
CAMION                             int64
CAMIONETA                          int64
EMERGENCIAS                        int64
ESPECIAL                           int64
FURGONETA                          int64
MOTOCICLETA                        int64
NO_IDENTIFICADO                    int64
SCOOTER_ELECTRICO                  int64
TRICIMOTO                          int64
VEHICULO_DEPORTIVO_UTILITARIO      int64
SUMA_DE_VEHICULOS                  int64
TIPO_ID_1                          object
EDAD_1                             int64
SEXO_1                             object
CONDICION_1                        object
PARTICIPANTE_1                     object
CASCO_1                            object
CINTURON_1                         object
dtype: object
```

**2.3.- Verificar Datos Nulos¶**

In [5]:

```
# verificar datos nulos
df.isna().sum()
```

Out[5]:

```
ID                          0
ANIO                        0
SINIESTROS                  0
LESIONADOS                  0
FALLECIDOS                  0
ENTE_DE_CONTROL             0
LATITUD_Y                   0
LONGITUD_X                  0
DPA_1                       0
PROVINCIA                   0
DPA_2                       0
CANTON                      0
DPA_3                       0
PARROQUIA                   0
DIRECCION                   0
ZONA_PLANIFICACION          0
ZONA                        0
ID_DE_LA_VIA                0
NOMBRE_DE_LA_VIA            0
UBICACION_DE_LA_VIA         0
JERARQUIA_DE_LA_VIA         0
FECHA                       0
HORA                        0
PERIODO_1                   0
PERIODO_2                   0
DIA_1                       0
DIA_2                       0
MES_1                       0
MES_2                       0
FERIADO                     0
CODIGO_CAUSA                0
CAUSA_PROBABLE              0
TIPO_DE_SINIESTRO           0
TIPO_DE_VEHICULO_1          0
SERVICIO_1                  0
AUTOMOVIL                   0
BICICLETA                   0
BUS                         0
CAMION                      0
CAMIONETA                   0
EMERGENCIAS                 0
ESPECIAL                    0
FURGONETA                   0
```

```
MOTOCICLETA                   0
NO_IDENTIFICADO               0
SCOOTER_ELECTRICO             0
TRICIMOTO                     0
VEHICULO_DEPORTIVO_UTILITARIO 0
SUMA_DE_VEHICULOS             0
TIPO_ID_1                     0
EDAD_1                        0
SEXO_1                        0
CONDICION_1                   0
PARTICIPANTE_1                0
CASCO_1                       0
CINTURON_1                    0
dtype: int64
```

In [6]:

```
null_data = df[df.isnull().any(axis=1)]
null_data
# No hay valores faltantes
```

Out[6]:

| ID | ANIO | SINIESTROS | LESIONADOS | FALLECIDOS | ENTE_DE_CONTROL | LATITUD_Y | LONGITUD_X | DPA_1 | PROVINCIA | ... | TRICIMOTO | VEHICULO_DEPOR |
|---|---|---|---|---|---|---|---|---|---|---|---|---|

0 rows × 56 columns

### 2.4.- Número de registro y columnas¶

In [7]:

```
#Verificando el numero de registros
df.shape
```

Out[7]:

```
(142582, 56)
```

In [8]:

```
# overview de los datos
df.sample(5)
```

Out[8]:

| | ID | ANIO | SINIESTROS | LESIONADOS | FALLECIDOS | ENTE_DE_CONTROL | LATITUD_Y | LONGITUD_X | DPA_1 | PROVINCIA | ... | TRICIMOTO | VE |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **23020** | 23021 | 2017 | CTE23021102017 | 1 | 0 | COMISION DE TRANSITO DEL ECUADOR - CTE | -1.896363 | -80.013552 | 9 | GUAYAS | ... | 0 | |
| **24655** | 24656 | 2017 | CTE24656112017 | 0 | 0 | COMISION DE TRANSITO DEL ECUADOR - CTE | -0.143638 | -79.240166 | 23 | SANTO DOMINGO DE LOS TSACHILAS | ... | 0 | |
| **100621** | 100622 | 2021 | CTE04558032021 | 1 | 0 | COMISION DE TRANSITO DEL ECUADOR - CTE | -2.144374 | -79.593768 | 9 | GUAYAS | ... | 0 | |
| **61879** | 61880 | 2019 | PNE07383042019 | 2 | 0 | POLICIA NACIONAL DEL ECUADOR | -0.090091 | -78.403312 | 17 | PICHINCHA | ... | 0 | |
| **60023** | 60024 | 2019 | DMQ05527032019 | 0 | 0 | AGENCIA METROPOLITANA DE TRANSITO DE QUITO - AMT | -0.254554 | -78.530844 | 17 | PICHINCHA | ... | 0 | |

5 rows × 56 columns

## 2.5.- Análisis de Target¶

In [9]:
```
# Obtener todos los valores únicos de la columna 'nombre_de_columna'
valores_unicos = df['CONDICION_1'].unique()
# Imprimir los valores únicos
print(valores_unicos)
```

['LESIONADO' 'NO IDENTIFICADO' 'ILESO' 'FALLECIDO']

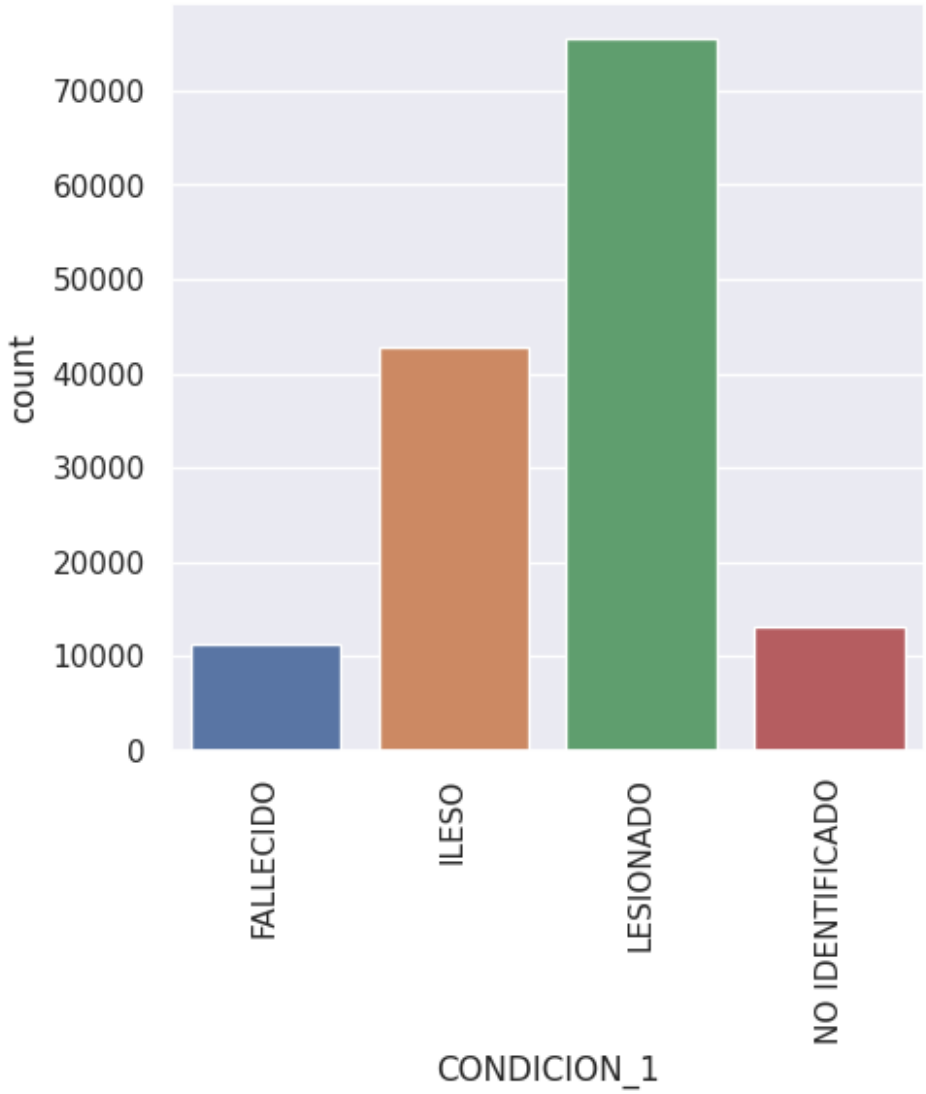
In [10]:
```
#Obtenemos la distribución de la Columna de Diagnostico
df['CONDICION_1'].value_counts()
```

Out[10]:
```
LESIONADO         75492
ILESO             42708
NO IDENTIFICADO    13104
FALLECIDO         11278
Name: CONDICION_1, dtype: int64
```

In [11]:
```
# Defino el tamaño de los graficos
sns.set(rc={'figure.figsize':(5,5)})
target = pd.DataFrame({'count' : df.groupby( ['CONDICION_1'] ).size()}).reset_index()
ax = sns.barplot(x="CONDICION_1", y="count", data=target)
ax.set_xticklabels(ax.get_xticklabels(),rotation=90)
```

Out[11]:
```
[Text(0, 0, 'FALLECIDO'),
 Text(1, 0, 'ILESO'),
```

Text(2, 0, 'LESIONADO'),
Text(3, 0, 'NO IDENTIFICADO')]




In [12]:

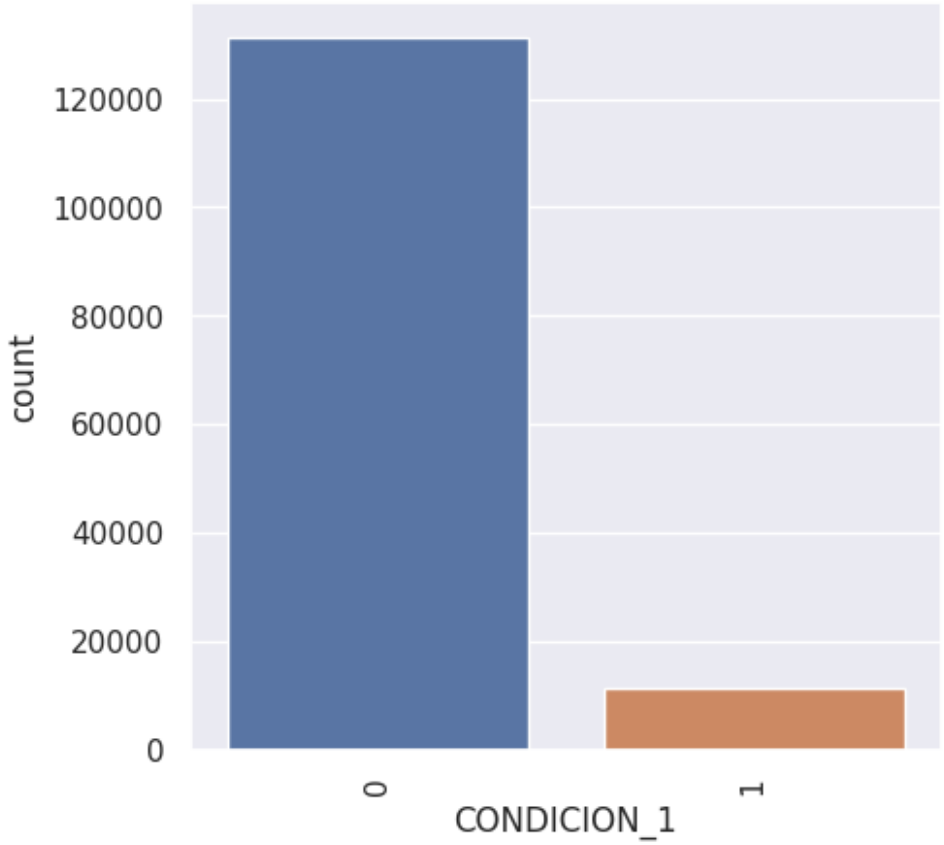
```
#Transformamos la variable target en Dicotomica
df['CONDICION_1'] = df['CONDICION_1'].replace({'LESIONADO': 0, 'ILESO': 0, 'NO IDEN
TIFICADO': 0, 'FALLECIDO': 1})
```

In [13]:

```
# Defino el tamaño de los graficos
sns.set(rc={'figure.figsize':(5,5)})
target = pd.DataFrame({'count' : df.groupby( ['CONDICION_1'] ).size()}).reset_index()
ax = sns.barplot(x="CONDICION_1", y="count", data=target)
ax.set_xticklabels(ax.get_xticklabels(),rotation=90)
```

Out[13]:

[Text(0, 0, '0'), Text(1, 0, '1')]




In [14]:

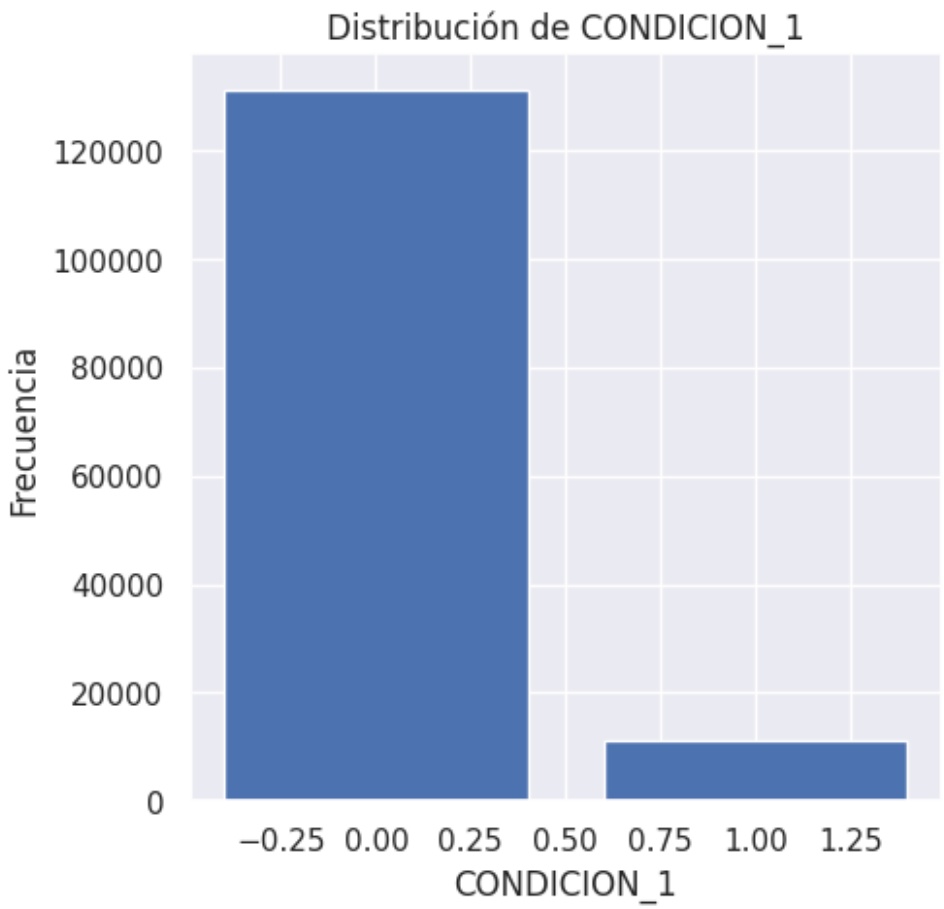
```
# Nombre del campo nominal a graficar
campo_nominal = 'CONDICION_1'

# Calcular la frecuencia de cada valor en el campo
frecuencias = df['CONDICION_1'].value_counts()

# Crear el gráfico de barras
plt.bar(frecuencias.index, frecuencias.values)

# Establecer el título y etiquetas de los ejes
plt.title('Distribución de {}'.format(campo_nominal))
plt.xlabel(campo_nominal)
plt.ylabel('Frecuencia')

# Mostrar el gráfico
plt.show()
```




**2.6- Filtros Iniciales¶**

In [15]:

```
#Implementacion de filtros basicos
#2017 al 2022
#Quitamos la condicion
filtro = (df["ANIO"] >= 2017) & (df["ANIO"] <= 2022) & (df["EDAD_1"] >= 0)
#Dataset de despliegue
filtro_val = (df["ANIO"] > 2022) & (df["EDAD_1"] >= 0)
df_val=df[filtro_val]
df = df[filtro]
```

In [16]:

```
df
```

Out[16]:

| | ID | ANIO | SINIESTROS | LESIONADOS | FALLECIDOS | ENTE_DE_CONTROL | LATITUD_Y | LONGITUD_X | DPA_1 | PROVINCIA | ... | TRICIMOTO |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 1 | 2017 | DMQ00001012017 | 1 | 0 | AGENCIA METROPOLITANA DE TRANSITO DE QUITO - AMT | -0.083501 | -78.417742 | 17 | PICHINCHA | ... | 0 |
| 1 | 2 | 2017 | ATM00002012017 | 1 | 0 | AGENCIA DE TRANSITO Y MOVILIDAD DE GUAYAQUIL -... | -2.246682 | -79.897754 | 9 | GUAYAS | ... | 0 |
| 2 | 3 | 2017 | PNE00003012017 | 1 | 0 | POLICIA NACIONAL DEL ECUADOR | -0.253881 | -79.217405 | 23 | SANTO DOMINGO DE LOS TSACHILAS | ... | 0 |
| 4 | 5 | 2017 | DMQ00005012017 | 0 | 0 | AGENCIA METROPOLITANA DE TRANSITO DE QUITO - AMT | -0.239721 | -78.512058 | 17 | PICHINCHA | ... | 0 |

In [17]:
#Obtenemos la distribución de la Columna de Diagnostico
df['CONDICION_1'].value_counts()

Out[17]:
0 79922
1 7680
Name: CONDICION_1, dtype: int64

**2.7.- Balanceo del Dataset en función del Target¶**

In [18]:
## Balancear el dataset y la clase de estudio que en este caso es la condición
# separar las instancias por categoría del target
fallecidos = df[df["CONDICION_1"] == 1]
no_fallecidos = df[df["CONDICION_1"] == 0]

# submuestrear la categoría mayoritaria
no_fallecidos_sub = resample(no_fallecidos, replace=False, n_samples=len(fallecidos), random_state=42)

# combinar las categorías submuestreadas con la categoría minoritaria
df = pd.concat([no_fallecidos_sub, fallecidos])

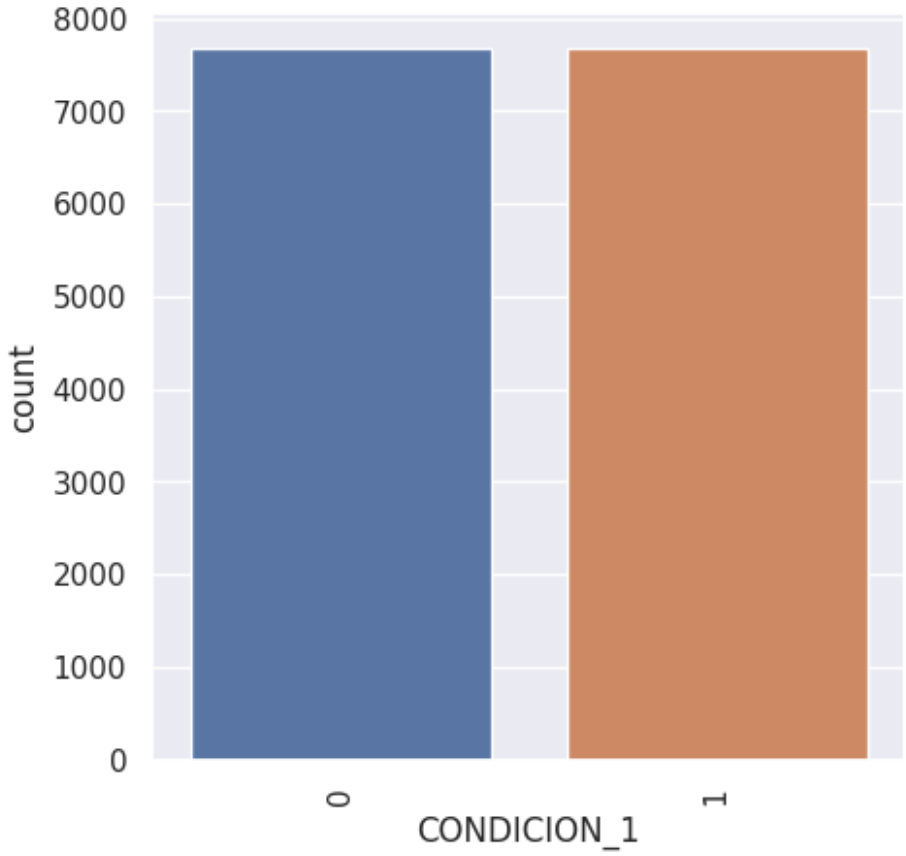
In [19]:
df['CONDICION_1'].value_counts()

Out[19]:
0 7680
1 7680
Name: CONDICION_1, dtype: int64

In [20]:
# Defino el tamaño de los graficos
sns.set(rc={'figure.figsize':(5,5)})
target = pd.DataFrame({'count' : df.groupby( ['CONDICION_1'] ).size()}).reset_index()

```
ax = sns.barplot(x="CONDICION_1", y="count", data=target)
ax.set_xticklabels(ax.get_xticklabels(),rotation=90)
```

Out[20]:
[Text(0, 0, '0'), Text(1, 0, '1')]




In [21]:
#Verificando el numero de registros
df.shape

Out[21]:
(15360, 56)

**2.8.- Análisis de Variables¶**

***Año¶***

In [22]:
## Analisis del variables
df['ANIO'].value_counts()

Out[22]:

```
2017    3104
2018    2764
2019    2698
2022    2528
2021    2333
2020    1933
Name: ANIO, dtype: int64
```

In [23]:

```
anio = pd.DataFrame({'count' : df.groupby( ['ANIO', 'CONDICION_1'] ).size()}).reset_index()
sns.set(rc={'figure.figsize':(10,5)})
ax = sns.barplot(x="ANIO", y="count", hue="CONDICION_1", data=anio)
ax.set_xticklabels(ax.get_xticklabels(),rotation=90)
# En que categorias se ven diferencias significativas?
```

Out[23]:

```
[Text(0, 0, '2017'),
 Text(1, 0, '2018'),
 Text(2, 0, '2019'),
 Text(3, 0, '2020'),
 Text(4, 0, '2021'),
 Text(5, 0, '2022')]
```


***Provincia¶***

In [24]:

```
df['PROVINCIA'].value_counts()
```

Out[24]:

```
PICHINCHA                       4361
GUAYAS                          2917
MANABI                           995
TUNGURAHUA                       851
AZUAY                            713
LOS RIOS                         711
CHIMBORAZO                       590
SANTO DOMINGO DE LOS TSACHILAS   569
COTOPAXI                         520
LOJA                             424
IMBABURA                         413
ESMERALDAS                       411
EL ORO                           334
CANIAR                           231
MORONA SANTIAGO                  196
ORELLANA                         178
BOLIVAR                          173
SANTA ELENA                      171
SUCUMBIOS                        148
CARCHI                           135
NAPO                             115
PASTAZA                          105
ZAMORA CHINCHIPE                  93
GALAPAGOS                          6
Name: PROVINCIA, dtype: int64
```

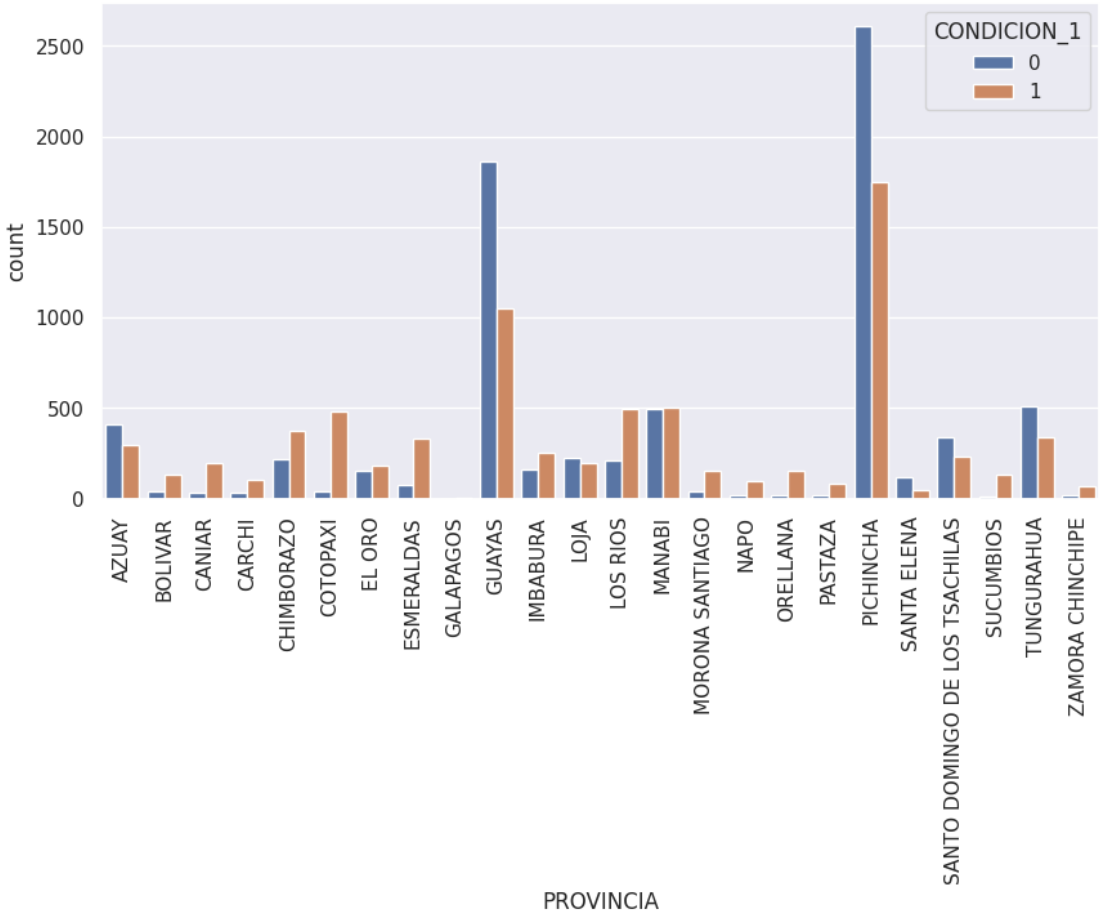
In [25]:

```
provincia = pd.DataFrame({'count' : df.groupby( ['PROVINCIA', 'CONDICION_1'] ).size()}).res
et_index()
sns.set(rc={'figure.figsize':(10,5)})
ax = sns.barplot(x="PROVINCIA", y="count", hue="CONDICION_1", data=provincia)
ax.set_xticklabels(ax.get_xticklabels(),rotation=90)
```

Out[25]:

```
[Text(0, 0, 'AZUAY'),
 Text(1, 0, 'BOLIVAR'),
 Text(2, 0, 'CANIAR'),
 Text(3, 0, 'CARCHI'),
 Text(4, 0, 'CHIMBORAZO'),
 Text(5, 0, 'COTOPAXI'),
 Text(6, 0, 'EL ORO'),
 Text(7, 0, 'ESMERALDAS'),
 Text(8, 0, 'GALAPAGOS'),
 Text(9, 0, 'GUAYAS'),
 Text(10, 0, 'IMBABURA'),
 Text(11, 0, 'LOJA'),
 Text(12, 0, 'LOS RIOS'),
```

Text(13, 0, 'MANABI'),
Text(14, 0, 'MORONA SANTIAGO'),
Text(15, 0, 'NAPO'),
Text(16, 0, 'ORELLANA'),
Text(17, 0, 'PASTAZA'),
Text(18, 0, 'PICHINCHA'),
Text(19, 0, 'SANTA ELENA'),
Text(20, 0, 'SANTO DOMINGO DE LOS TSACHILAS'),
Text(21, 0, 'SUCUMBIOS'),
Text(22, 0, 'TUNGURAHUA'),
Text(23, 0, 'ZAMORA CHINCHIPE')]




***Zona***¶

In [26]:
df['ZONA'].value_counts()

Out[26]:
URBANA 8787
RURAL 6470

```
Rural      103
Name: ZONA, dtype: int64
```

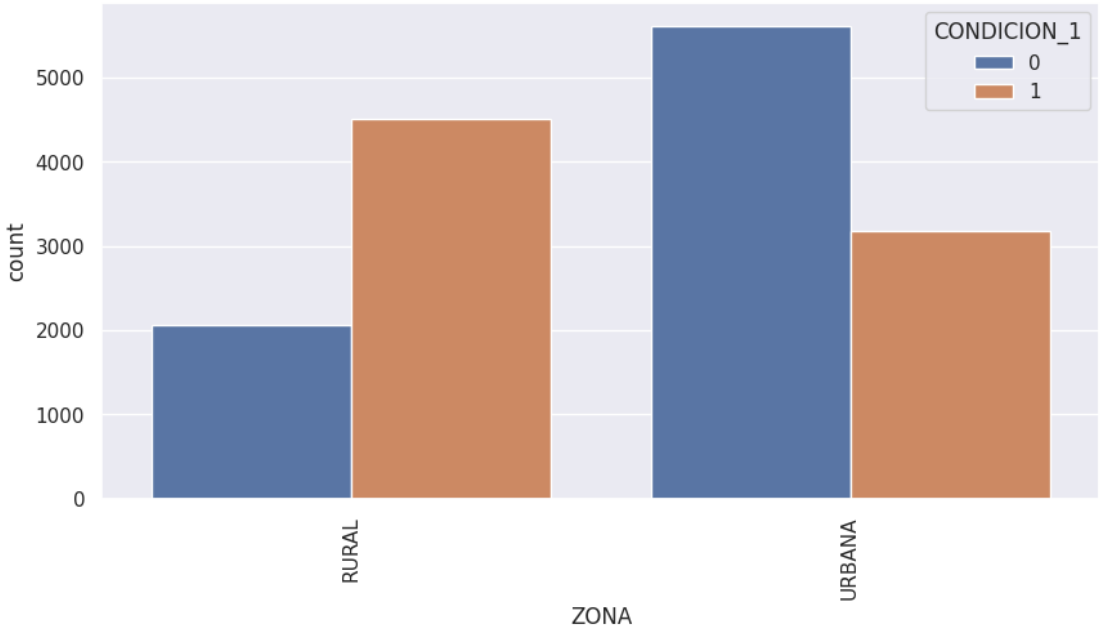
In [27]:
```
df["ZONA"] = df["ZONA"].replace("Rural", "RURAL")
```

In [28]:
```
df['ZONA'].value_counts()
```

Out[28]:
```
URBANA    8787
RURAL     6573
Name: ZONA, dtype: int64
```

In [29]:
```
zona = pd.DataFrame({'count' : df.groupby( ['ZONA', 'CONDICION_1'] ).size()}).reset_index()
sns.set(rc={'figure.figsize':(10,5)})
ax = sns.barplot(x="ZONA", y="count", hue="CONDICION_1", data=zona)
ax.set_xticklabels(ax.get_xticklabels(),rotation=90)
```

Out[29]:
```
[Text(0, 0, 'RURAL'), Text(1, 0, 'URBANA')]
```




***Periodo¶***

In [30]:
```
df['PERIODO_1'].value_counts()
```

Out[30]:

```
DE 19H00 A 19H59    1099
DE 20H00 A 20H59     886
DE 18H00 A 18H59     829
DE 21H00 A 21H59     740
DE 17H00 A 17H59     723
DE 16H00 A 16H59     717
DE 07H00 A 07H59     716
DE 06H00 A 06H59     698
DE 15H00 A 15H59     697
DE 13H00 A 13H59     661
DE 08H00 A 08H59     643
DE 22H00 A 22H59     641
DE 05H00 A 05H59     615
DE 14H00 A 14H59     613
DE 12H00 A 12H59     584
DE 11H00 A 11H59     573
DE 10H00 A 10H59     539
DE 23H00 A 23H59     532
DE 00H00 A 00H59     510
DE 09H00 A 09H59     504
DE 01H00 A 01H59     488
DE 02H00 A 02H59     471
DE 04H00 A 04H59     451
DE 03H00 A 03H59     430
Name: PERIODO_1, dtype: int64
```

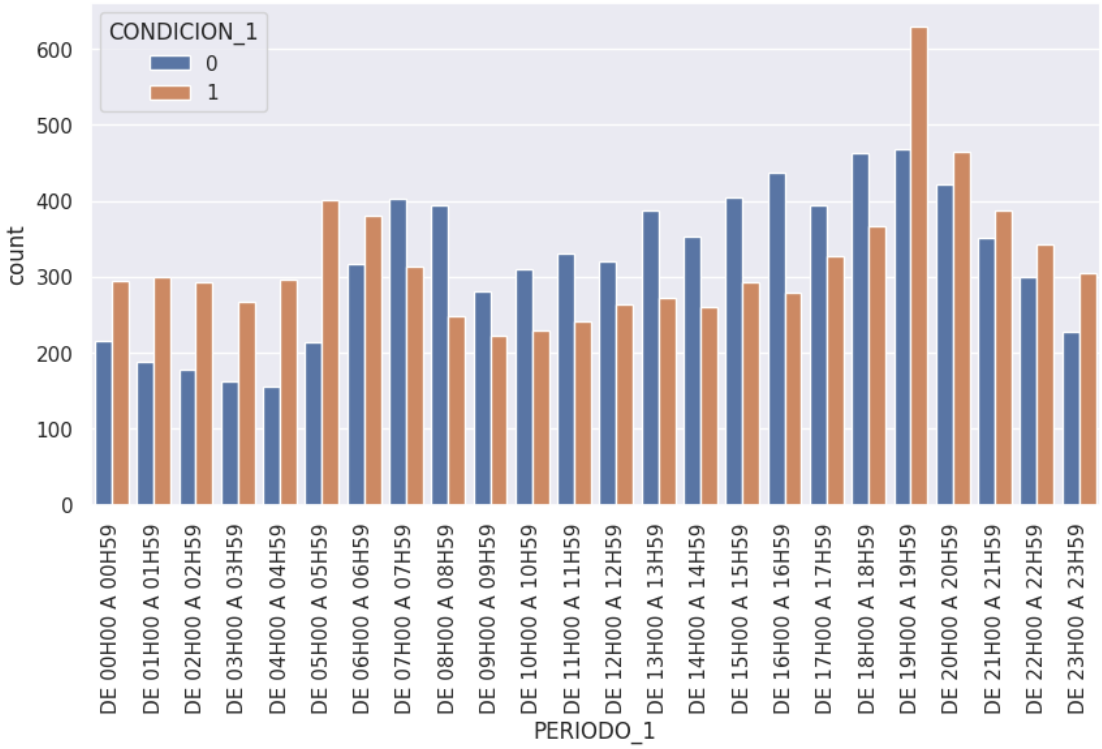
In [31]:

```
hora = pd.DataFrame({'count' : df.groupby( ['PERIODO_1', 'CONDICION_1'] ).size()}).reset_in
dex()
sns.set(rc={'figure.figsize':(10,5)})
ax = sns.barplot(x="PERIODO_1", y="count", hue="CONDICION_1", data=hora)
ax.set_xticklabels(ax.get_xticklabels(),rotation=90)
```

Out[31]:

```
[Text(0, 0, 'DE 00H00 A 00H59'),
 Text(1, 0, 'DE 01H00 A 01H59'),
 Text(2, 0, 'DE 02H00 A 02H59'),
 Text(3, 0, 'DE 03H00 A 03H59'),
 Text(4, 0, 'DE 04H00 A 04H59'),
 Text(5, 0, 'DE 05H00 A 05H59'),
 Text(6, 0, 'DE 06H00 A 06H59'),
 Text(7, 0, 'DE 07H00 A 07H59'),
 Text(8, 0, 'DE 08H00 A 08H59'),
 Text(9, 0, 'DE 09H00 A 09H59'),
 Text(10, 0, 'DE 10H00 A 10H59'),
 Text(11, 0, 'DE 11H00 A 11H59'),
 Text(12, 0, 'DE 12H00 A 12H59'),
```

Text(13, 0, 'DE 13H00 A 13H59'),
Text(14, 0, 'DE 14H00 A 14H59'),
Text(15, 0, 'DE 15H00 A 15H59'),
Text(16, 0, 'DE 16H00 A 16H59'),
Text(17, 0, 'DE 17H00 A 17H59'),
Text(18, 0, 'DE 18H00 A 18H59'),
Text(19, 0, 'DE 19H00 A 19H59'),
Text(20, 0, 'DE 20H00 A 20H59'),
Text(21, 0, 'DE 21H00 A 21H59'),
Text(22, 0, 'DE 22H00 A 22H59'),
Text(23, 0, 'DE 23H00 A 23H59')]



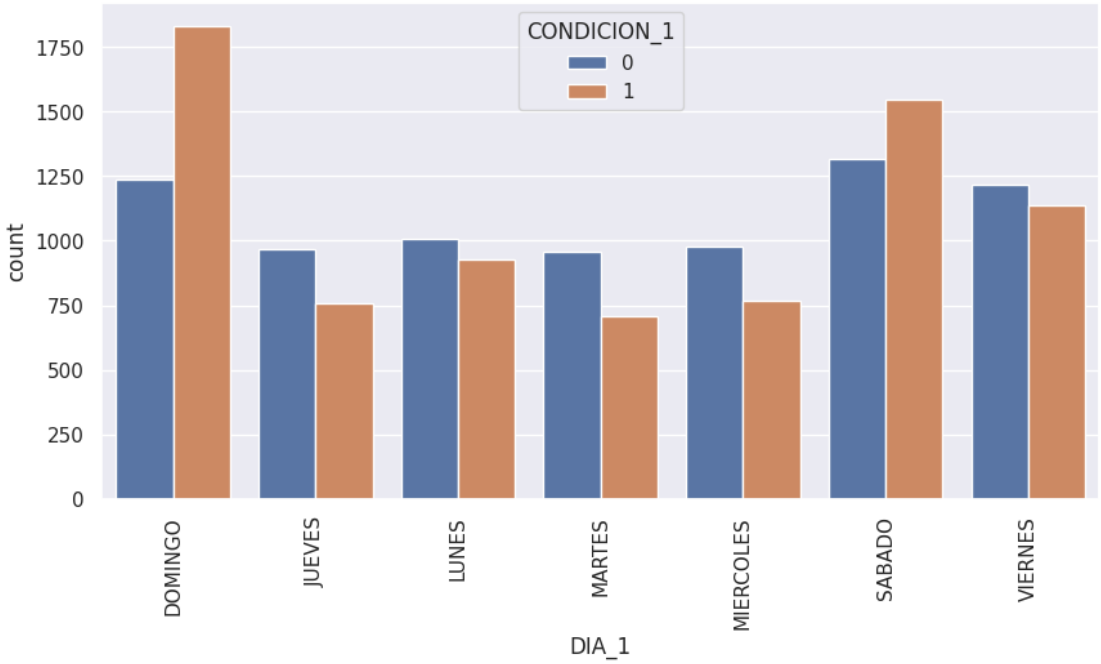

***Día¶***

In [32]:
df['DIA_1'].value_counts()

Out[32]:
DOMINGO 3068
SABADO 2867
VIERNES 2355
LUNES 1936
MIERCOLES 1747
JUEVES 1721
MARTES 1666
Name: DIA_1, dtype: int64

In [33]:
```
dia = pd.DataFrame({'count' : df.groupby( ['DIA_1', 'CONDICION_1'] ).size()}).reset_index()
sns.set(rc={'figure.figsize':(10,5)})
ax = sns.barplot(x="DIA_1", y="count", hue="CONDICION_1", data=dia)
ax.set_xticklabels(ax.get_xticklabels(),rotation=90)
```

Out[33]:
```
[Text(0, 0, 'DOMINGO'),
 Text(1, 0, 'JUEVES'),
 Text(2, 0, 'LUNES'),
 Text(3, 0, 'MARTES'),
 Text(4, 0, 'MIERCOLES'),
 Text(5, 0, 'SABADO'),
 Text(6, 0, 'VIERNES')]
```




***Mes¶***

In [34]:
```
df['MES_1'].value_counts()
```

Out[34]:
```
DICIEMBRE     1447
SEPTIEMBRE    1344
NOVIEMBRE     1320
ENERO         1317
OCTUBRE       1304
AGOSTO        1296
JULIO         1278
```

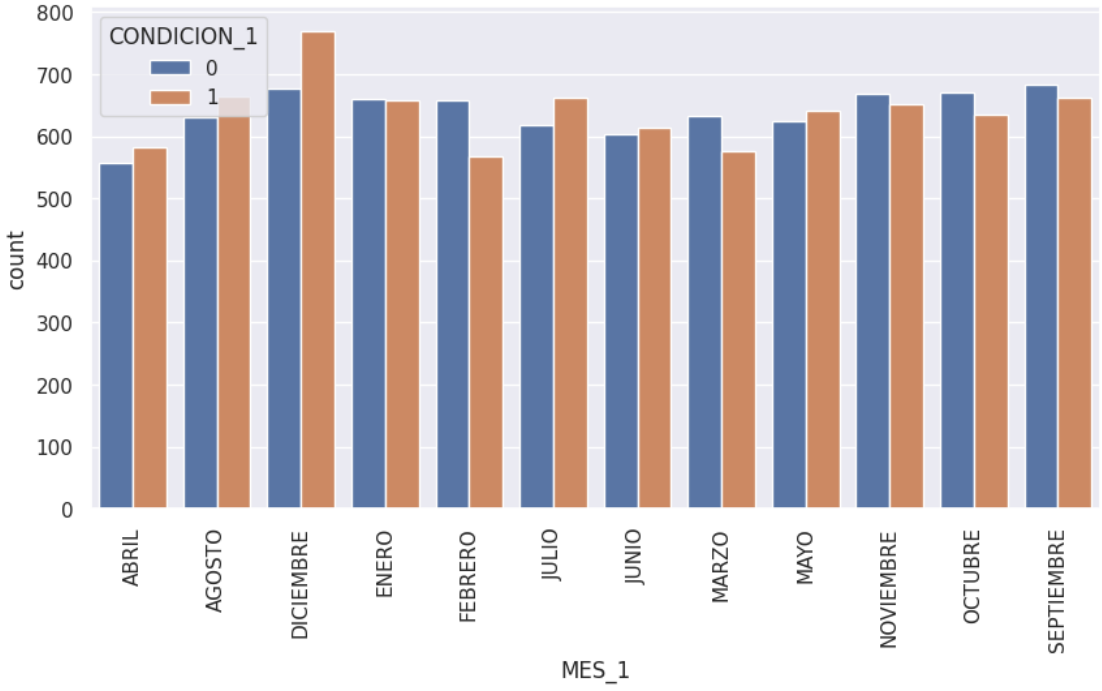
```
MAYO         1267
FEBRERO      1224
JUNIO        1216
MARZO        1207
ABRIL        1140
Name: MES_1, dtype: int64
```

In [35]:

```
mes = pd.DataFrame({'count' : df.groupby( ['MES_1', 'CONDICION_1'] ).size()}).reset_index()
sns.set(rc={'figure.figsize':(10,5)})
ax = sns.barplot(x="MES_1", y="count", hue="CONDICION_1", data=mes)
ax.set_xticklabels(ax.get_xticklabels(),rotation=90)
```

Out[35]:

```
[Text(0, 0, 'ABRIL'),
 Text(1, 0, 'AGOSTO'),
 Text(2, 0, 'DICIEMBRE'),
 Text(3, 0, 'ENERO'),
 Text(4, 0, 'FEBRERO'),
 Text(5, 0, 'JULIO'),
 Text(6, 0, 'JUNIO'),
 Text(7, 0, 'MARZO'),
 Text(8, 0, 'MAYO'),
 Text(9, 0, 'NOVIEMBRE'),
 Text(10, 0, 'OCTUBRE'),
 Text(11, 0, 'SEPTIEMBRE')]
```



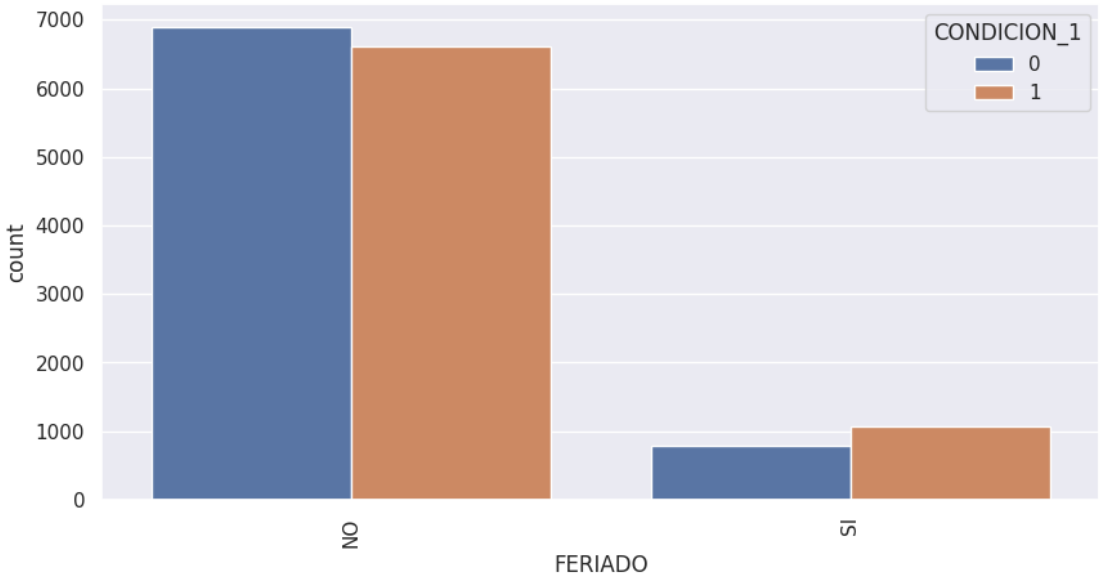
***Feriado¶***

In [36]:
df['FERIADO'].value_counts()

Out[36]:
NO    13497
SI     1863
Name: FERIADO, dtype: int64

In [37]:
feriado = pd.DataFrame({'count' : df.groupby( ['FERIADO', 'CONDICION_1'] ).size()}).reset_index()
sns.set(rc={'figure.figsize':(10,5)})
ax = sns.barplot(x="FERIADO", y="count", hue="CONDICION_1", data=feriado)
ax.set_xticklabels(ax.get_xticklabels(),rotation=90)

Out[37]:
[Text(0, 0, 'NO'), Text(1, 0, 'SI')]




***Causa Probable¶***

In [38]:
df['CAUSA_PROBABLE'].value_counts()

Out[38]:
CONDUCIR DESATENTO A LAS CONDICIONES DE TRANSITO (CELULAR, PANTALLAS DE VIDEO, COMIDA, MAQUILLAJE O CUALQUIER OTRO ELEMENTO DISTRACTOR ).    4448
CONDUCIR VEHICULO SUPERANDO LOS LIMITES MAXIMOS DE VELOCIDAD.

2289
NO RESPETAR LAS SENIALES REGLAMENTARIAS DE TRANSITO. (PARE, CEDA EL P
ASO, LUZ ROJA DEL SEMAFORO, ETC). 1573
CONDUCE BAJO LA INFLUENCIA DE ALCOHOL, SUSTANCIAS ESTUPEFACIENTES O
PSICOTROPICAS Y/O MEDICAMENTOS. 11
10
NO MANTENER LA DISTANCIA PRUDENCIAL CON RESPECTO AL VEHICULO QUE L
E ANTECEDE. 943
NO GUARDAR LA DISTANCIA LATERAL MINIMA DE SEGURIDAD ENTRE VEHICULO
S. 768
NO TRANSITAR POR LAS ACERAS O ZONAS DE SEGURIDAD DESTINADAS PARA EL
EFECTO. 638
REALIZAR CAMBIO BRUSCO O INDEBIDO DE CARRIL.
638
NO CEDER EL DERECHO DE VIA O PREFERENCIA DE PASO AL PEATON.
473
NO CEDER EL DERECHO DE VIA O PREFERENCIA DE PASO A VEHICULOS.
397
CONDUCIR EN SENTIDO CONTRARIO A LA VIA NORMAL DE CIRCULACION.
280
PEATON QUE CRUZA LA CALZADA SIN RESPETAR LA SENIALIZACION EXISTENTE (
SEMAFOROS O SENIALES MANUALES). 2
63
CONDICIONES AMBIENTALES Y/O ATMOSFERICAS (NIEBLA, NEBLINA, GRANIZO, L
LUVIA). 260
CASO FORTUITO O FUERZA MAYOR (EXPLOSION DE NEUMATICO NUEVO, DERRU
MBE, INUNDACION, CAIDA DE PUENTE, ARBOL, PRESENCIA INTEMPESTIVA E IMPR
EVISTA DE SEMOVIENTES EN LA VIA, ETC.). 225
CONDUCIR EN ESTADO DE SOMNOLENCIA O MALAS CONDICIONES FISICAS (SUEN
IO, CANSANCIO Y FATIGA). 205
BAJARSE O SUBIRSE DE VEHICULOS EN MOVIMIENTO SIN TOMAR LAS PRECAUCI
ONES DEBIDAS. 128
DEJAR O RECOGER PASAJEROS EN LUGARES NO PERMITIDOS.
125
FALLA MECANICA EN LOS SISTEMAS Y/O NEUMATICOS (SISTEMA DE FRENOS, DIR
ECCION, ELECTRONICO O MECANICO). 11
5
MALAS CONDICIONES DE LA VIA Y/O CONFIGURACION. (ILUMINACION Y DISENIO
). 108
ADELANTAR O REBASAR A OTRO VEHICULO EN MOVIMIENTO EN ZONAS O SITIOS
PELIGROSOS TALES COMO: CURVAS, PUENTES, TUNELES, PENDIENTES, ETC.
88
PRESENCIA DE AGENTES EXTERNOS EN LA VIA (AGUA, ACEITE, PIEDRA, LASTRE,
ESCOMBROS, MADEROS, ETC.). 87
DANIOS MECANICOS PREVISIBLES.
66

PEATON TRANSITA BAJO INFLUENCIA DE ALCOHOL, SUSTANCIAS ESTUPEFACIEN
TES O PSICOTROPICAS Y/O MEDICAMENTOS.
55
MAL ESTACIONADO - EL CONDUCTOR QUE DETENGA O ESTACIONE VEHICULOS E
N SITIOS O ZONAS QUE ENTRANIEN PELIGRO, TALES COMO ZONA DE SEGURIDAD,
CURVAS, PUENTES, TUNELES, PENDIENTES. 32
PESO Y VOLUMEN - NO CUMPLIR CON LAS NORMAS DE SEGURIDAD NECESARIAS
AL TRANSPORTAR CARGAS. 18
DISPOSITIVO REGULADOR DE TRANSITO EN MAL ESTADO DE FUNCIONAMIENTO (
SEMAFORO). 16
NO RESPETAR LAS SENIALES MANUALES DEL AGENTE DE TRANSITO.
12
Name: CAUSA_PROBABLE, dtype: int64

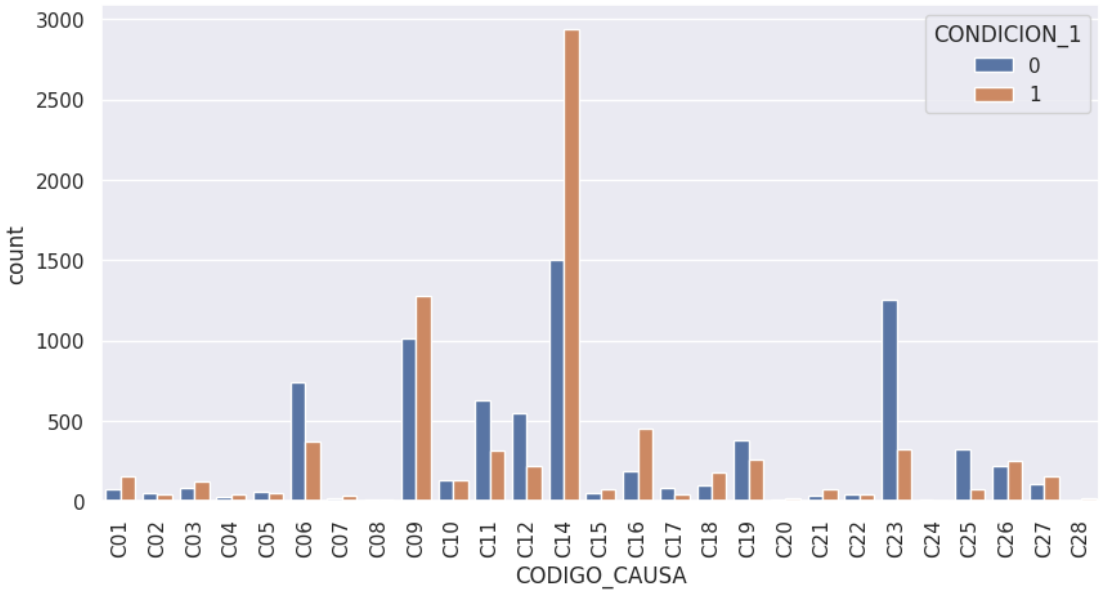
In [39]:

```
causa = pd.DataFrame({'count' : df.groupby( ['CODIGO_CAUSA', 'CONDICION_1'] ).size()}).r
eset_index()
sns.set(rc={'figure.figsize':(10,5)})
ax = sns.barplot(x="CODIGO_CAUSA", y="count", hue="CONDICION_1", data=causa)
ax.set_xticklabels(ax.get_xticklabels(),rotation=90)
```

Out[39]:

```
[Text(0, 0, 'C01'),
 Text(1, 0, 'C02'),
 Text(2, 0, 'C03'),
 Text(3, 0, 'C04'),
 Text(4, 0, 'C05'),
 Text(5, 0, 'C06'),
 Text(6, 0, 'C07'),
 Text(7, 0, 'C08'),
 Text(8, 0, 'C09'),
 Text(9, 0, 'C10'),
 Text(10, 0, 'C11'),
 Text(11, 0, 'C12'),
 Text(12, 0, 'C14'),
 Text(13, 0, 'C15'),
 Text(14, 0, 'C16'),
 Text(15, 0, 'C17'),
 Text(16, 0, 'C18'),
 Text(17, 0, 'C19'),
 Text(18, 0, 'C20'),
 Text(19, 0, 'C21'),
 Text(20, 0, 'C22'),
 Text(21, 0, 'C23'),
 Text(22, 0, 'C24'),
 Text(23, 0, 'C25'),
 Text(24, 0, 'C26'),
```

Text(25, 0, 'C27'),
Text(26, 0, 'C28')]




***TIPO DE SINIESTRO***¶

In [40]:
df['TIPO_DE_SINIESTRO'].value_counts()

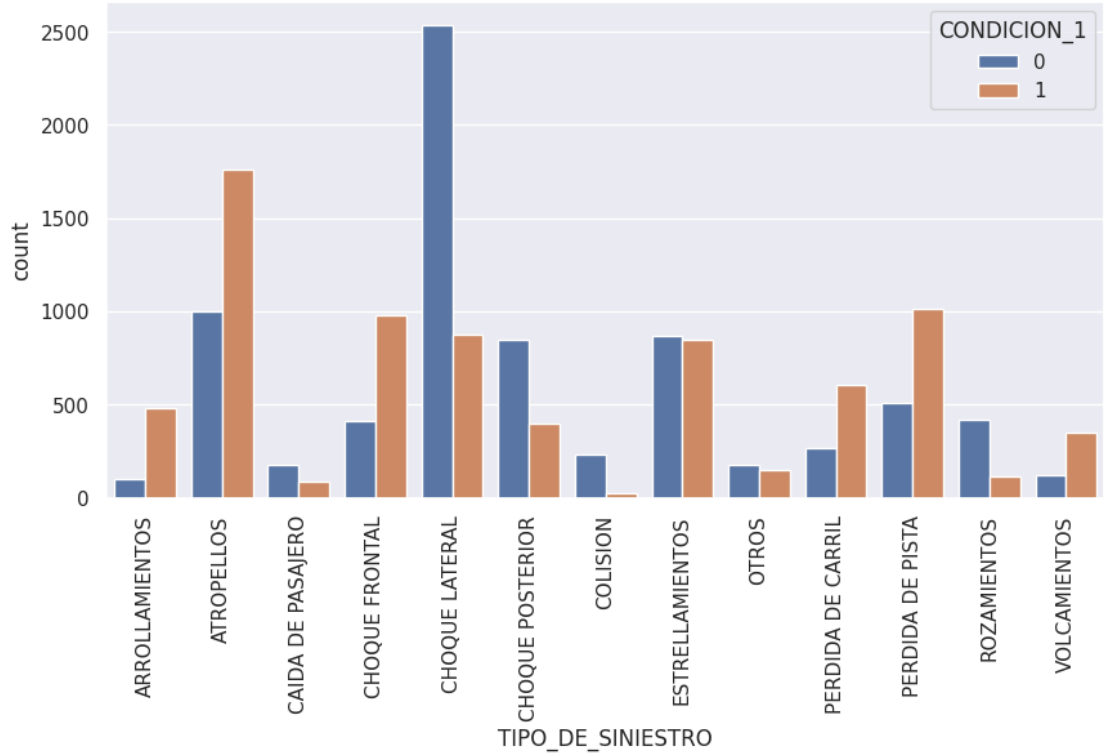
Out[40]:
CHOQUE LATERAL 3415
ATROPELLOS 2760
ESTRELLAMIENTOS 1711
PERDIDA DE PISTA 1521
CHOQUE FRONTAL 1393
CHOQUE POSTERIOR 1245
PERDIDA DE CARRIL 871
ARROLLAMIENTOS 584
ROZAMIENTOS 533
VOLCAMIENTOS 473
OTROS 329
CAIDA DE PASAJERO 263
COLISION 262
Name: TIPO_DE_SINIESTRO, dtype: int64

In [41]:
tipo_siniestro = pd.DataFrame({'count' : df.groupby( ['TIPO_DE_SINIESTRO', 'CONDICION_1'] ).size()}).reset_index()
sns.set(rc={'figure.figsize':(10,5)})
ax = sns.barplot(x="TIPO_DE_SINIESTRO", y="count", hue="CONDICION_1", data=tipo_sin

```
iestro)
ax.set_xticklabels(ax.get_xticklabels(),rotation=90)
```

Out[41]:

```
[Text(0, 0, 'ARROLLAMIENTOS'),
 Text(1, 0, 'ATROPELLOS'),
 Text(2, 0, 'CAIDA DE PASAJERO'),
 Text(3, 0, 'CHOQUE FRONTAL'),
 Text(4, 0, 'CHOQUE LATERAL'),
 Text(5, 0, 'CHOQUE POSTERIOR'),
 Text(6, 0, 'COLISION'),
 Text(7, 0, 'ESTRELLAMIENTOS'),
 Text(8, 0, 'OTROS'),
 Text(9, 0, 'PERDIDA DE CARRIL'),
 Text(10, 0, 'PERDIDA DE PISTA'),
 Text(11, 0, 'ROZAMIENTOS'),
 Text(12, 0, 'VOLCAMIENTOS')]
```




***TIPO DE VEHÍCULO¶***

In [42]:

```
df['TIPO_DE_VEHICULO_1'].value_counts()
```

Out[42]:

```
MOTOCICLETA          4117
AUTOMOVIL            4022
```

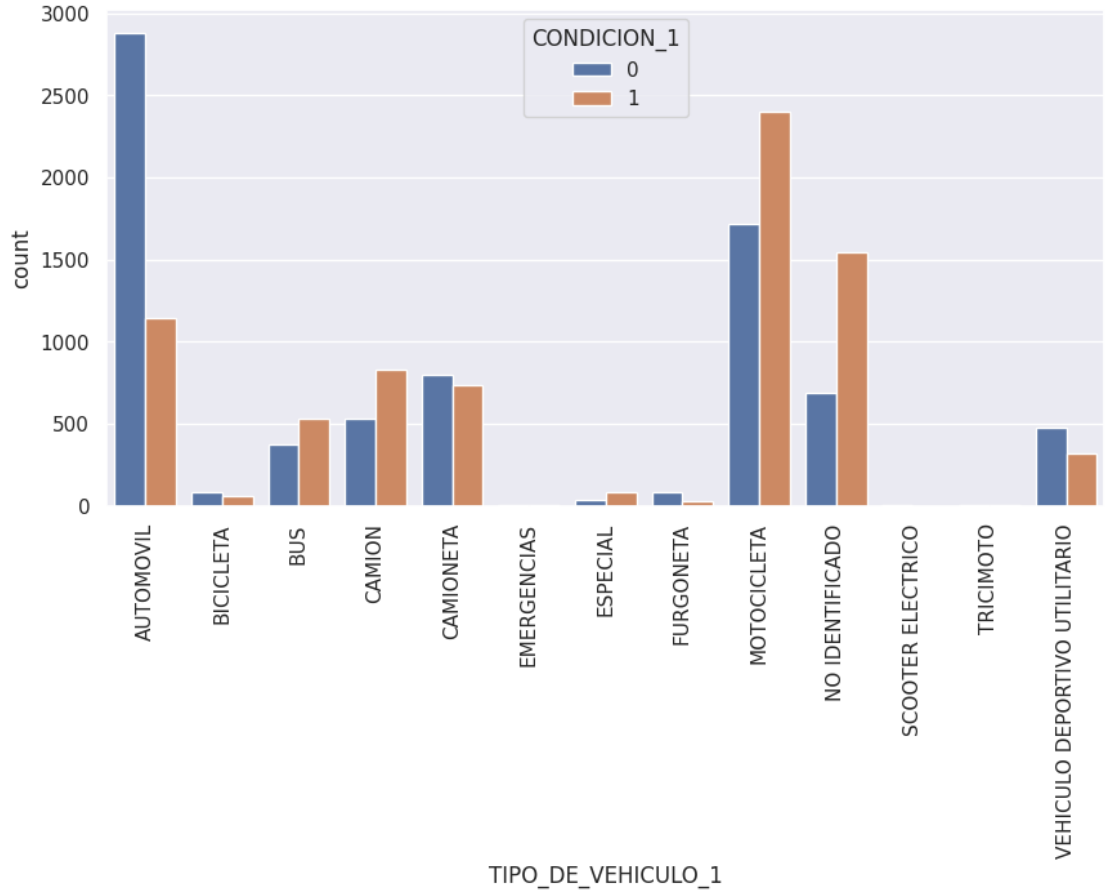
```
NO IDENTIFICADO                2230
CAMIONETA                      1535
CAMION                         1354
BUS                             908
VEHICULO DEPORTIVO UTILITARIO   798
BICICLETA                       147
ESPECIAL                        123
FURGONETA                       114
EMERGENCIAS                       6
TRICIMOTO                         4
SCOOTER ELECTRICO                 2
Name: TIPO_DE_VEHICULO_1, dtype: int64
```

In [43]:

```
tipo_vehiculo = pd.DataFrame({'count' : df.groupby( ['TIPO_DE_VEHICULO_1', 'CONDICIO
N_1'] ).size()}).reset_index()
sns.set(rc={'figure.figsize':(10,5)})
ax = sns.barplot(x="TIPO_DE_VEHICULO_1", y="count", hue="CONDICION_1", data=tipo_
vehiculo)
ax.set_xticklabels(ax.get_xticklabels(),rotation=90)
```

Out[43]:

```
[Text(0, 0, 'AUTOMOVIL'),
 Text(1, 0, 'BICICLETA'),
 Text(2, 0, 'BUS'),
 Text(3, 0, 'CAMION'),
 Text(4, 0, 'CAMIONETA'),
 Text(5, 0, 'EMERGENCIAS'),
 Text(6, 0, 'ESPECIAL'),
 Text(7, 0, 'FURGONETA'),
 Text(8, 0, 'MOTOCICLETA'),
 Text(9, 0, 'NO IDENTIFICADO'),
 Text(10, 0, 'SCOOTER ELECTRICO'),
 Text(11, 0, 'TRICIMOTO'),
 Text(12, 0, 'VEHICULO DEPORTIVO UTILITARIO')]
```




***SERVICIO***¶

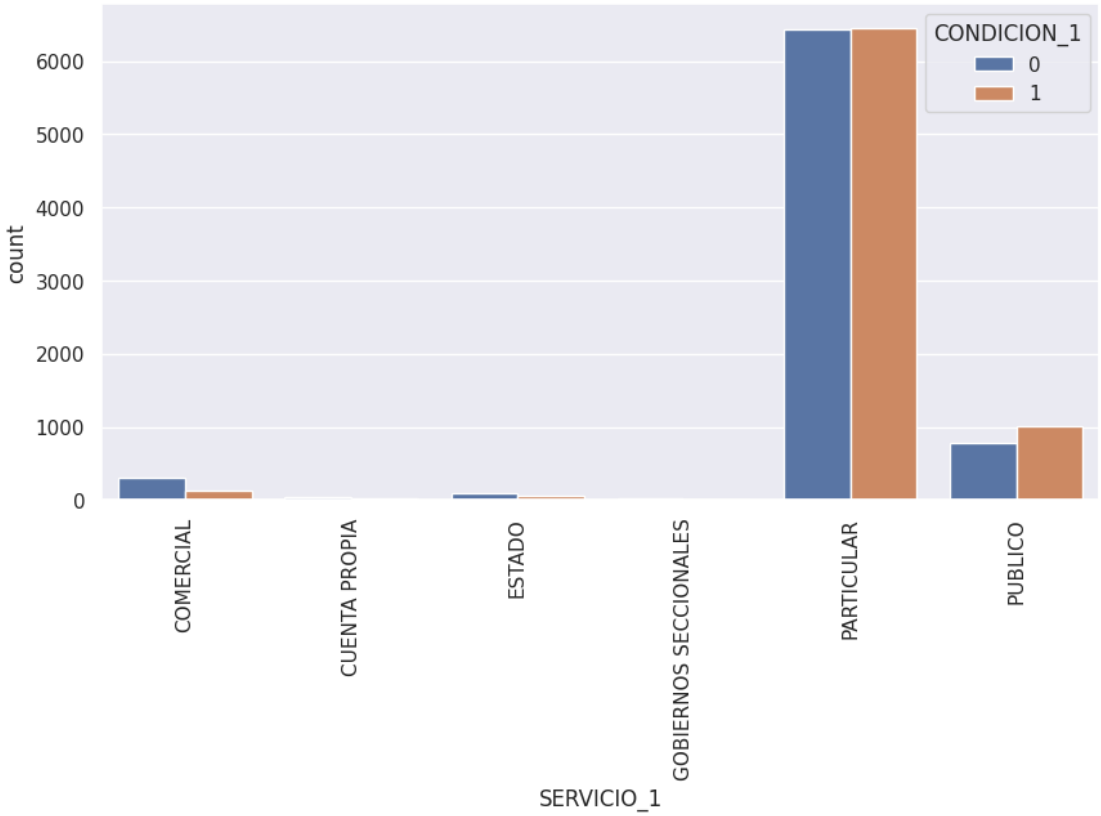
In [44]:
df['SERVICIO_1'].value_counts()

Out[44]:
PARTICULAR 12890
PUBLICO 1785
COMERCIAL 427
ESTADO 163
CUENTA PROPIA 67
GOBIERNOS SECCIONALES 28
Name: SERVICIO_1, dtype: int64

In [45]:
```
servicio = pd.DataFrame({'count' : df.groupby( ['SERVICIO_1', 'CONDICION_1'] ).size()}).rese
t_index()
sns.set(rc={'figure.figsize':(10,5)})
ax = sns.barplot(x="SERVICIO_1", y="count", hue="CONDICION_1", data=servicio)
ax.set_xticklabels(ax.get_xticklabels(),rotation=90)
```

Out[45]:

[Text(0, 0, 'COMERCIAL'),
 Text(1, 0, 'CUENTA PROPIA'),
 Text(2, 0, 'ESTADO'),
 Text(3, 0, 'GOBIERNOS SECCIONALES'),
 Text(4, 0, 'PARTICULAR'),
 Text(5, 0, 'PUBLICO')]




***SUMA_DE_VEHICULOS***¶

In [46]:
df['SUMA_DE_VEHICULOS'].value_counts()

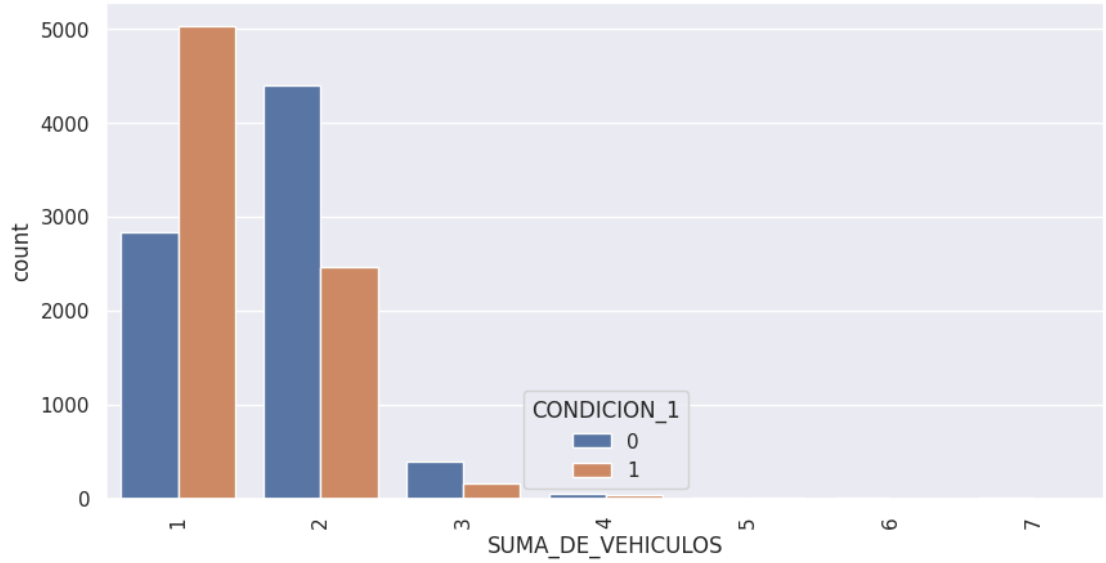
Out[46]:
1 7863
2 6862
3 539
4 77
5 16
6 2
7 1
Name: SUMA_DE_VEHICULOS, dtype: int64

In [47]:

```
suma_vehiculos = pd.DataFrame({'count' : df.groupby( ['SUMA_DE_VEHICULOS', 'CONDICION_1'] ).size()}).reset_index()
sns.set(rc={'figure.figsize':(10,5)})
ax = sns.barplot(x="SUMA_DE_VEHICULOS", y="count", hue="CONDICION_1", data=suma_vehiculos)
ax.set_xticklabels(ax.get_xticklabels(),rotation=90)
```

Out[47]:

```
[Text(0, 0, '1'),
 Text(1, 0, '2'),
 Text(2, 0, '3'),
 Text(3, 0, '4'),
 Text(4, 0, '5'),
 Text(5, 0, '6'),
 Text(6, 0, '7')]
```




***EDAD***¶

In [48]:

```
df['EDAD_1'].value_counts()
```

Out[48]:

```
25    552
28    519
30    506
26    505
27    496
     ...
93      4
91      4
95      3
```

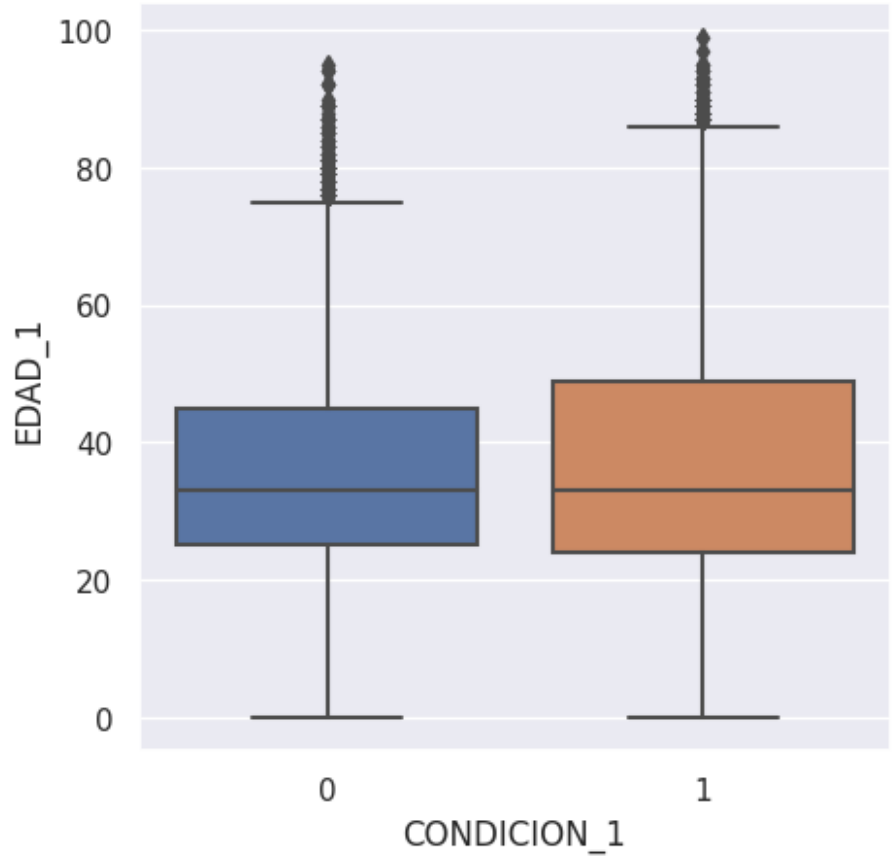
```
97     1
99     1
Name: EDAD_1, Length: 98, dtype: int64
```

In [49]:

```
sns.set(rc={'figure.figsize':(5,5)})
ax=sns.boxplot(x='CONDICION_1', y='EDAD_1', data=df)
ax.set_xticklabels(ax.get_xticklabels(),rotation=0)
```

Out[49]:

```
[Text(0, 0, '0'), Text(1, 0, '1')]
```




In [50]:

```
df[['EDAD_1']].describe(percentiles=[.01,.99])
```

Out[50]:

| | EDAD_1 |
|---|---|
| count | 15360.000000 |
| mean | 36.607292 |
| std | 17.360393 |

```
min     0.000000
1%      0.000000
50%     33.000000
99%     84.000000
max     99.000000
```

In [51]:

```
percentil_99=df['EDAD_1'].quantile(0.99)
print('percentil 99 de edad: '+ str(percentil_99))
```

```
percentil 99 de edad: 84.0
```

In [52]:

```
n_percentil_99 = df[df['EDAD_1']>percentil_99].shape[0]
print('El porcentaje de personas con edad mayor a '+str(percentil_99) +
      ' es ' +str(round(n_percentil_99/df.shape[0] *100)) + '%' )
```

```
El porcentaje de personas con edad mayor a 84.0 es 1%
```

In [53]:

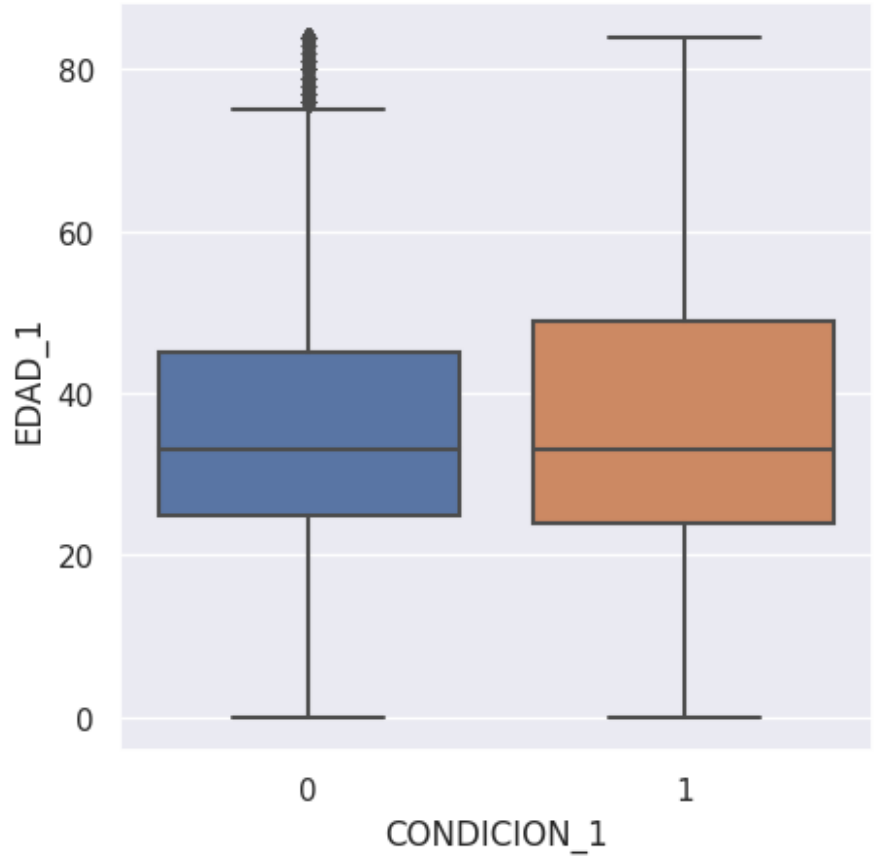
```
#imputamos con el percentil 99 las edades outliers
df['EDAD_1'] = np.where(df['EDAD_1']> percentil_99,percentil_99, df['EDAD_1'] )
```

In [54]:

```
sns.set(rc={'figure.figsize':(5,5)})
ax=sns.boxplot(x='CONDICION_1', y='EDAD_1', data=df)
ax.set_xticklabels(ax.get_xticklabels(),rotation=0)
```

Out[54]:

```
[Text(0, 0, '0'), Text(1, 0, '1')]
```




***SEXO***¶

In [55]:
df['SEXO_1'].value_counts()

Out[55]:
HOMBRE 12655
MUJER 2612
NO IDENTIFICADO 93
Name: SEXO_1, dtype: int64

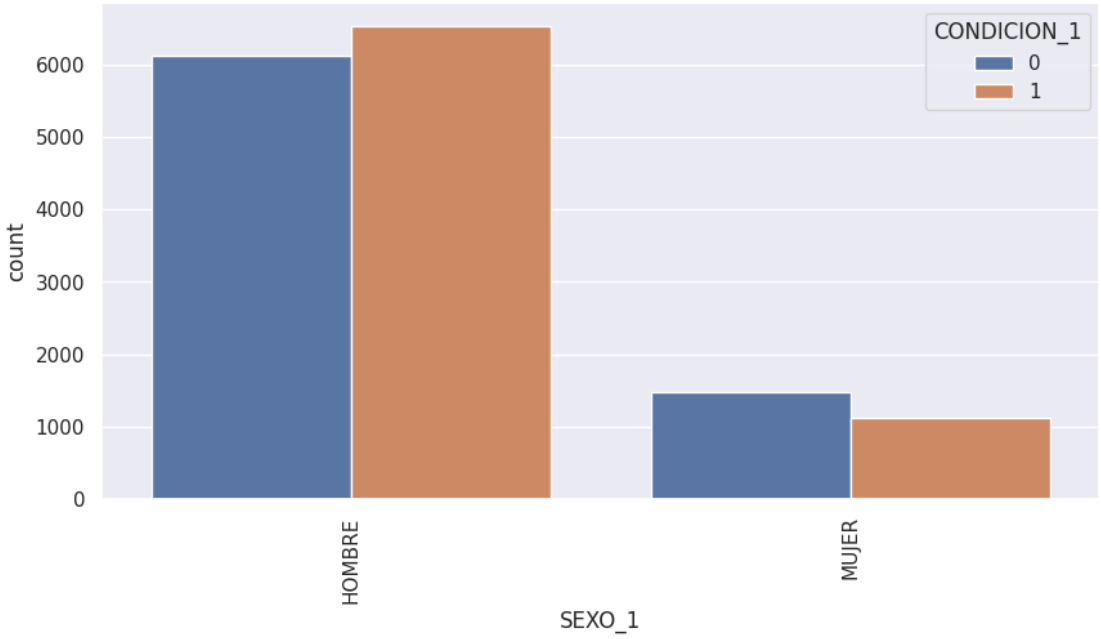
In [56]:
filtro = (df["SEXO_1"] != "NO IDENTIFICADO")
df = df[filtro]

In [57]:
df['SEXO_1'].value_counts()

Out[57]:
HOMBRE 12655
MUJER 2612
Name: SEXO_1, dtype: int64

In [58]:
```
sexo = pd.DataFrame({'count' : df.groupby( ['SEXO_1', 'CONDICION_1'] ).size()}).reset_index(
)
sns.set(rc={'figure.figsize':(10,5)})
ax = sns.barplot(x="SEXO_1", y="count", hue="CONDICION_1", data=sexo)
ax.set_xticklabels(ax.get_xticklabels(),rotation=90)
```

Out[58]:
[Text(0, 0, 'HOMBRE'), Text(1, 0, 'MUJER')]




***PARTICIPANTE***¶

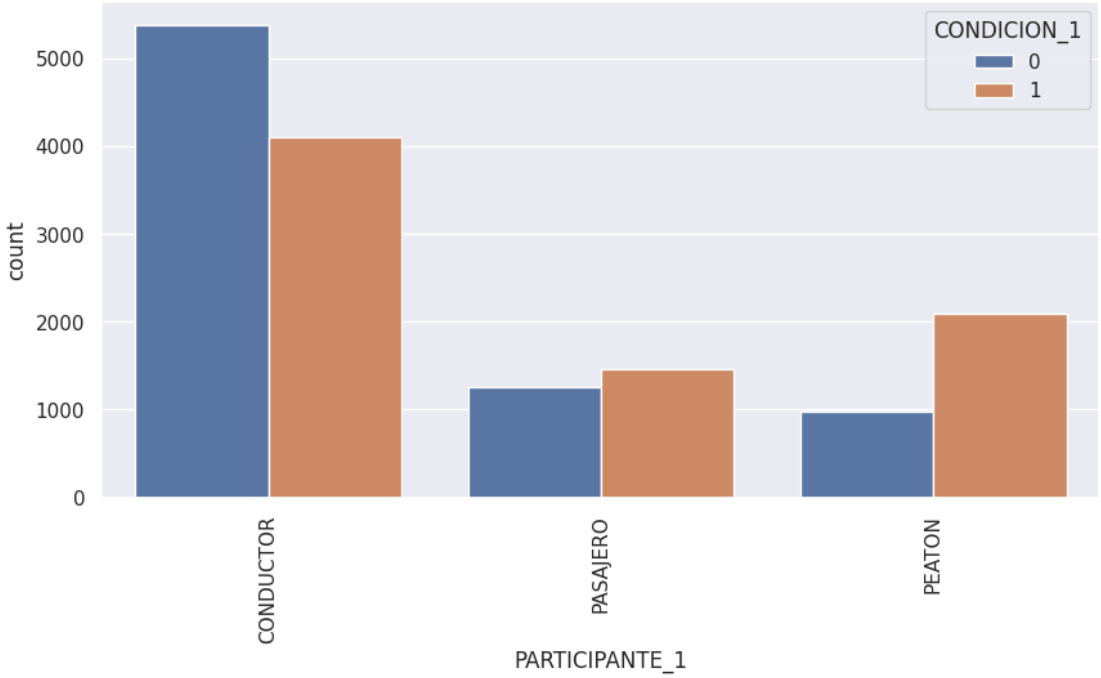
In [59]:
```
df['PARTICIPANTE_1'].value_counts()
```

Out[59]:
```
CONDUCTOR    9486
PEATON       3062
PASAJERO     2719
Name: PARTICIPANTE_1, dtype: int64
```

In [60]:
```
participante = pd.DataFrame({'count' : df.groupby( ['PARTICIPANTE_1', 'CONDICION_1'] ).siz
e()}).reset_index()
sns.set(rc={'figure.figsize':(10,5)})
ax = sns.barplot(x="PARTICIPANTE_1", y="count", hue="CONDICION_1", data=participante)
ax.set_xticklabels(ax.get_xticklabels(),rotation=90)
```

Out[60]:

[Text(0, 0, 'CONDUCTOR'), Text(1, 0, 'PASAJERO'), Text(2, 0, 'PEATON')]



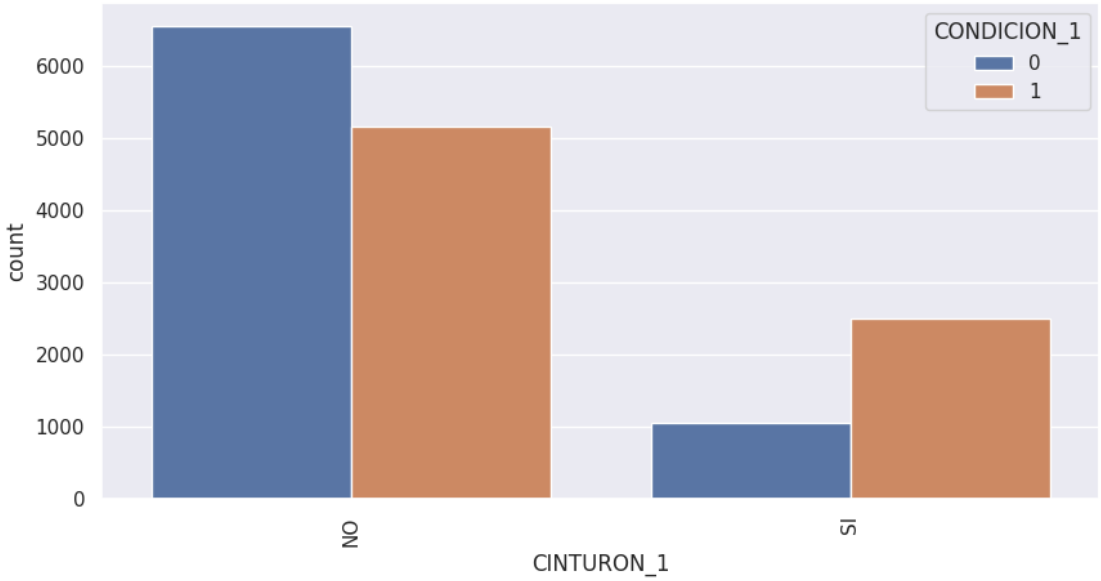

***CINTURÓN***¶

In [61]:
df['CINTURON_1'].value_counts()

Out[61]:
NO 11719
SI 3548
Name: CINTURON_1, dtype: int64

In [62]:
cinturon = pd.DataFrame({'count' : df.groupby( ['CINTURON_1', 'CONDICION_1'] ).size()}).reset_index()
sns.set(rc={'figure.figsize':(10,5)})
ax = sns.barplot(x="CINTURON_1", y="count", hue="CONDICION_1", data=cinturon)
ax.set_xticklabels(ax.get_xticklabels(),rotation=90)

Out[62]:
[Text(0, 0, 'NO'), Text(1, 0, 'SI')]




**2.9.- Consultando y Quitando Duplicados¶**

In [63]:
df_duplicados = df.loc[:,['DPA_1','PROVINCIA','PERIODO_1','PERIODO_2','DIA_1','DIA_2','MES_1','MES_2','CODIGO_CAUSA','CAUSA_PROBABLE']]

In [64]:
df_duplicados.sample(5)

Out[64]:

| | DPA_1 | PROVINCIA | PERIODO_1 | PERIODO_2 | DIA_1 | DIA_2 | MES_1 | MES_2 | CODIGO_CAUSA | CAUSA_PROBABLE |
|---|---|---|---|---|---|---|---|---|---|---|
| 54411 | 12 | LOS RIOS | DE 15H00 A 15H59 | 15 | DOMINGO | 7 | DICIEMBRE | 12 | C11 | NO MANTENER LA DISTANCIA PRUDENCIAL CON RESPEC... |
| 90405 | 17 | PICHINCHA | DE 11H00 A 11H59 | 11 | DOMINGO | 7 | SEPTIEMBRE | 9 | C09 | CONDUCIR VEHICULO SUPERANDO LOS LIMITES |

| | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| | | | | | | | | | | MAXIMO... |
| 85328 | 9 | GUAYAS | DE 15H00 A 15H59 | 15 | VIERNES | 5 | MAYO | 5 | C23 | NO RESPETAR LAS SENIALES REGLAMENTARIAS DE TRA... |
| 112487 | 23 | SANTO DOMINGO DE LOS TSACHILAS | DE 13H00 A 13H59 | 13 | SABADO | 6 | OCTUBRE | 10 | C23 | NO RESPETAR LAS SENIALES REGLAMENTARIAS DE TRA... |
| 109505 | 23 | SANTO DOMINGO DE LOS TSACHILAS | DE 13H00 A 13H59 | 13 | MARTES | 2 | AGOSTO | 8 | C19 | REALIZAR CAMBIO BRUSCO O INDEBIDO DE CARRIL. |

In [65]:

```
#quitando variables repetidas
#df_limpio = df.loc[:,['ANIO','LESIONADOS','FALLECIDOS','DPA_1','ZONA','PERIODO_2','DIA_2','FERIADO','CODIGO_CAUSA','TIPO_DE_SINIESTRO','TIPO_DE_VEHICULO_1','SERVICIO_1','SUMA_DE_VEHICULOS','EDAD_1','SEXO_1','PARTICIPANTE_1','CINTURON_1','CONDICION_1']]
#df_limpio = df['ANIO','LESIONADOS','FALLECIDOS','LATITUD_Y','LONGITUD_X']
#df_limpio
df_limpio = df.loc[:,["PROVINCIA", "ZONA", "PERIODO_2","DIA_2","MES_2","FERIADO","CODIGO_CAUSA", "TIPO_DE_SINIESTRO", "TIPO_DE_VEHICULO_1", "SERVICIO_1", "SEXO_1","PARTICIPANTE_1","CINTURON_1","CASCO_1","ANIO",'AUTOMOVIL','BICICLETA','BUS','CAMION','CAMIONETA','EMERGENCIAS','ESPECIAL','FURGONETA','MOTOCICLETA','NO_IDENTIFICADO','SCOOTER_ELECTRICO','TRICIMOTO','VEHICULO_DEPORTIVO_UTILITARIO', "SUMA_DE_VEHICULOS", "EDAD_1","CONDICION_1"]]
df_limpio
# Obtener las columnas
columnas = df_limpio.columns
```

```
# Mostrar las columnas
print(columnas)
```

```
Index(['PROVINCIA', 'ZONA', 'PERIODO_2', 'DIA_2', 'MES_2', 'FERIADO',
       'CODIGO_CAUSA', 'TIPO_DE_SINIESTRO', 'TIPO_DE_VEHICULO_1', 'SERVICIO_1',
       'SEXO_1', 'PARTICIPANTE_1', 'CINTURON_1', 'CASCO_1', 'ANIO',
       'AUTOMOVIL', 'BICICLETA', 'BUS', 'CAMION', 'CAMIONETA', 'EMERGENCIAS',
       'ESPECIAL', 'FURGONETA', 'MOTOCICLETA', 'NO_IDENTIFICADO',
       'SCOOTER_ELECTRICO', 'TRICIMOTO', 'VEHICULO_DEPORTIVO_UTILITARIO',
       'SUMA_DE_VEHICULOS', 'EDAD_1', 'CONDICION_1'],
      dtype='object')
```

In [66]:

```
#Verificando el numero de registros
df_limpio.shape
```

Out[66]:

```
(15267, 31)
```

## 3. Data Preparation¶

### 3.1 Estandarización de variables numericas¶

In [67]:

```
df_limpio.dtypes
```

Out[67]:

```
PROVINCIA                 object
ZONA                      object
PERIODO_2                  int64
DIA_2                      int64
MES_2                      int64
FERIADO                   object
CODIGO_CAUSA              object
TIPO_DE_SINIESTRO         object
TIPO_DE_VEHICULO_1        object
SERVICIO_1                object
SEXO_1                    object
PARTICIPANTE_1            object
CINTURON_1                object
CASCO_1                   object
ANIO                       int64
AUTOMOVIL                  int64
BICICLETA                  int64
BUS                        int64
CAMION                     int64
```

```
CAMIONETA                       int64
EMERGENCIAS                     int64
ESPECIAL                        int64
FURGONETA                       int64
MOTOCICLETA                     int64
NO_IDENTIFICADO                 int64
SCOOTER_ELECTRICO               int64
TRICIMOTO                       int64
VEHICULO_DEPORTIVO_UTILITARIO   int64
SUMA_DE_VEHICULOS               int64
EDAD_1                        float64
CONDICION_1                     int64
dtype: object
```

In [ ]:

```
#ZONA_dummies = pd.get_dummies(df_limpio.ZONA, prefix='ZONA')
#FERIADO_dummies = pd.get_dummies(df_limpio.FERIADO, prefix='FERIADO')
#CODIGO_CAUSA_dummies = pd.get_dummies(df_limpio.CODIGO_CAUSA, prefix='CODI
GO_CAUSA')
#TIPO_DE_SINIESTRO_dummies = pd.get_dummies(df_limpio.TIPO_DE_SINIESTRO, prefi
x='TIPO_DE_SINIESTRO')
#TIPO_DE_VEHICULO_1_dummies = pd.get_dummies(df_limpio.TIPO_DE_VEHICULO_1,
prefix='TIPO_DE_VEHICULO_1')
#SERVICIO_1_dummies = pd.get_dummies(df_limpio.SERVICIO_1, prefix='SERVICIO_1')
#SEXO_1_dummies = pd.get_dummies(df_limpio.SEXO_1, prefix='SEXO_1')
#PARTICIPANTE_1_dummies = pd.get_dummies(df_limpio.PARTICIPANTE_1, prefix='PART
ICIPANTE_1')
#CINTURON_1_dummies = pd.get_dummies(df_limpio.CINTURON_1, prefix='CINTURON_
1')
#CONDICION_1_dummies = pd.get_dummies(df_limpio.CONDICION_1, prefix='CONDICIO
N_1')
```

In [ ]:

```
# Concatenar el dataframe original con el one-hot encoding
#df_limpio = pd.concat([df_limpio, ZONA_dummies,FERIADO_dummies,CODIGO_CAUSA_
dummies,TIPO_DE_SINIESTRO_dummies,TIPO_DE_VEHICULO_1_dummies,SERVICIO_1
_dummies,SEXO_1_dummies,PARTICIPANTE_1_dummies,CINTURON_1_dummies], axis=1)
#df_limpio = pd.concat([df_limpio, ZONA_dummies,CODIGO_CAUSA_dummies,TIPO_DE_S
INIESTRO_dummies,TIPO_DE_VEHICULO_1_dummies,SERVICIO_1_dummies,PARTICIPA
NTE_1_dummies], axis=1)
```

In [68]:

```
# Preprocesamiento de datos
X = df_limpio.drop("CONDICION_1", axis=1)  # Variables predictoras
y = df_limpio["CONDICION_1"]  # Variable objetivo
```

In [69]:

```
# Codificar variables categóricas
categorical_features = ["PROVINCIA", "ZONA","FERIADO","CODIGO_CAUSA", "TIPO_DE_SINIESTRO", "TIPO_DE_VEHICULO_1", "SERVICIO_1", "SEXO_1","PARTICIPANTE_1","CINTURON_1","CASCO_1"]
label_encoders = {}
for feature in categorical_features:
    label_encoders[feature] = LabelEncoder()
    X[feature] = label_encoders[feature].fit_transform(X[feature])
```

In [70]:

```
# Normalizar variables numéricas
numerical_features = ["ANIO",'AUTOMOVIL',"PERIODO_2","DIA_2","MES_2",'BICICLETA','BUS','CAMION','CAMIONETA','EMERGENCIAS','ESPECIAL','FURGONETA','MOTOCICLETA','NO_IDENTIFICADO','SCOOTER_ELECTRICO','TRICIMOTO','VEHICULO_DEPORTIVO_UTILITARIO', "SUMA_DE_VEHICULOS", "EDAD_1"]
scaler = StandardScaler()
X[numerical_features] = scaler.fit_transform(X[numerical_features])
```

In [71]:

X

| | PROVINCIA | ZONA | PERIODO_2 | DIA_2 | MES_2 | FERIADO | CODIGO_CAUSA | TIPO_DE_SINIESTRO | TIPO_DE_VEHICULO_1 | SERVICIO_1 | ... | EMER |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **102642** | 18 | 1 | 0.667802 | -1.179792 | -0.467092 | 0 | 21 | 4 | 0 | 4 | ... | |
| **82312** | 18 | 1 | 0.371202 | -0.204358 | -1.325502 | 0 | 5 | 7 | 0 | 4 | ... | |
| **39057** | 13 | 0 | -1.111799 | 1.258794 | -0.467092 | 1 | 21 | 7 | 4 | 4 | ... | |
| **54038** | 9 | 1 | -0.666898 | -1.179792 | 1.535864 | 1 | 5 | 6 | 0 | 4 | ... | |
| **11025** | 22 | 1 | 0.667802 | 0.771077 | -0.467092 | 0 | 1 | 4 | 0 | 4 | ... | |
| **...** | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... |
| **139114** | 18 | 1 | 0.074602 | 0.771077 | 1.535864 | 1 | 14 | 1 | 8 | 4 | ... | |
| **139123** | 0 | 1 | 0.519502 | 0.771077 | 1.535864 | 1 | 12 | 4 | 8 | 4 | ... | |
| **139130** | 18 | 1 | 0.816102 | 0.771077 | 1.535864 | 1 | 19 | 9 | 0 | 4 | ... | |
| **139138** | 18 | 1 | 0.964402 | 0.771077 | 1.535864 | 1 | 24 | 1 | 9 | 4 | ... | |
| **139146** | 18 | 0 | 1.261002 | 0.771077 | 1.535864 | 1 | 8 | 7 | 9 | 4 | ... | |

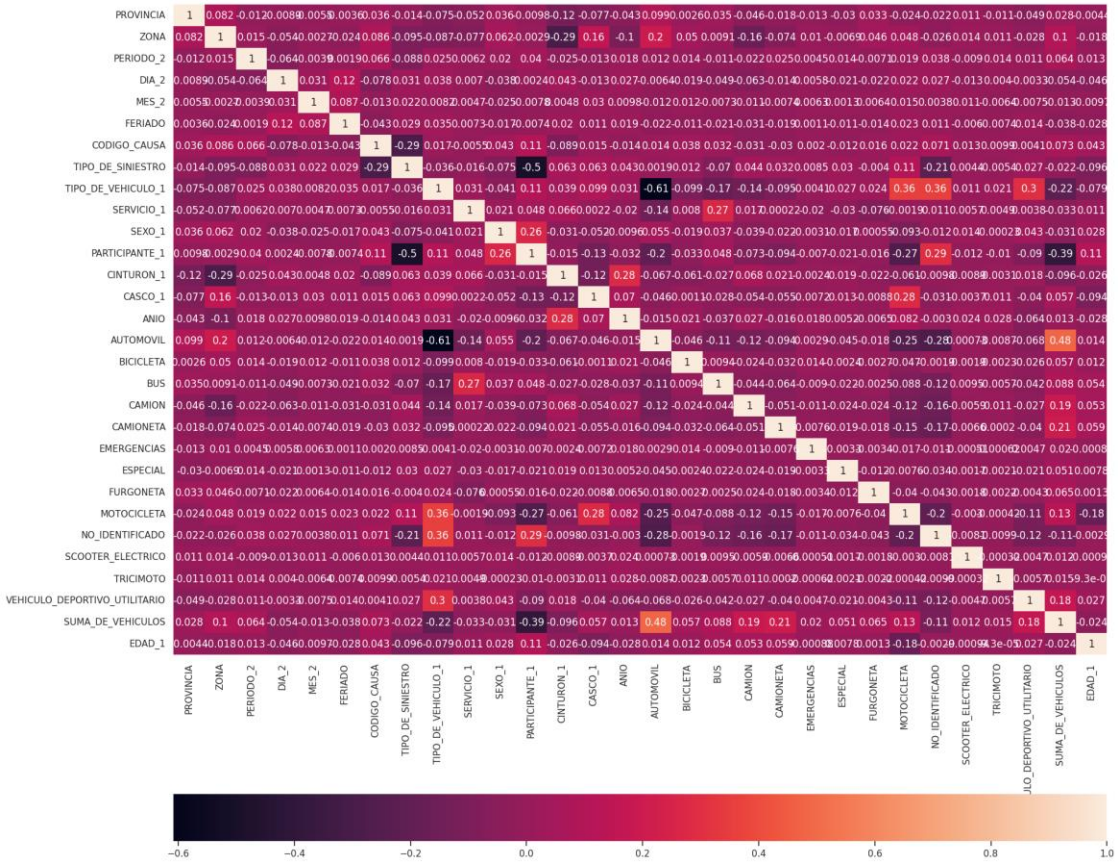
15267 rows × 30 columns

In [72]:

```
f, ax = plt.subplots(figsize=(20, 20))
sns.heatmap(X.corr(), annot = True,  cbar_kws= {'orientation': 'horizontal'})
```

Out[72]:

<Axes: >




In [73]:

```
# Eliminar columnas repetidas
#df_limpio = df_limpio.drop('ZONA', axis=1)
#df_limpio = df_limpio.drop('FERIADO', axis=1)
#df_limpio = df_limpio.drop('CODIGO_CAUSA', axis=1)
#df_limpio = df_limpio.drop('TIPO_DE_SINIESTRO', axis=1)
#df_limpio = df_limpio.drop('TIPO_DE_VEHICULO_1', axis=1)
#df_limpio = df_limpio.drop('SERVICIO_1', axis=1)
#df_limpio = df_limpio.drop('SEXO_1', axis=1)
#df_limpio = df_limpio.drop('PARTICIPANTE_1', axis=1)
#df_limpio = df_limpio.drop('CINTURON_1', axis=1)
#df_limpio = df_limpio.drop('CONDICION_1', axis=1)
```

In [ ]:

```
#df_limpio
```

## 4. Modeling¶

### 4.1 Dataset Entrenamiento y Prueba¶

In [74]:

```
#df_encoded=df_limpio
# Separar el conjunto de datos en datos de entrenamiento y prueba
```

```
#X = df_encoded.drop('CONDICION_1', axis=1) # todas las columnas excepto la columna objeti
vo
#y = df_encoded['CONDICION_1'] # la columna objetivo
#X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_state=42)
```

In [75]:

```
# Dividir el dataset en conjuntos de entrenamiento y prueba
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
```

### 4.2 Clasificador Bayesiano¶

In [76]:

```
# Entrenar el clasificador bayesiano
gnb = GaussianNB()
# Entrenar el modelo
cb=gnb.fit(X_train, y_train)
```

### 4.3 Maquinas de Soporte Vectorial¶

In [77]:

```
#Maquinas de Soporte
# Entrenar la SVM
svm = SVC(kernel='linear', probability=True)
msv=svm.fit(X_train, y_train)
```

### 4.4 Implementación Regresión Logística¶

In [79]:

```
## Regresion logistica multinomial
# Entrenar el clasificador de regresión logística
lr = LogisticRegression(solver='lbfgs', max_iter=20000)
rl = lr.fit(X_train, y_train)
```

## 5. Evaluation¶

### *5.1 Clasificador Bayesiano*¶

In [80]:

```
# Realizar predicciones en el conjunto de prueba
y_pred_cb = cb.predict(X_test)
y_prob_cb = cb.predict_proba(X_test)
```

In [81]:

```
accuracy = accuracy_score(y_test, y_pred_cb)
precision = precision_score(y_test, y_pred_cb, average='macro')
recall = recall_score(y_test, y_pred_cb, average='macro')
```

In [82]:

```
# Imprimir las métricas de rendimiento
print('Exactitud:', accuracy)
print('Precisión:', precision)
print('Sensibilidad:', recall)
```

```
Exactitud: 0.6830386378519974
Precisión: 0.7244108929467369
Sensibilidad: 0.6826154083680717
```

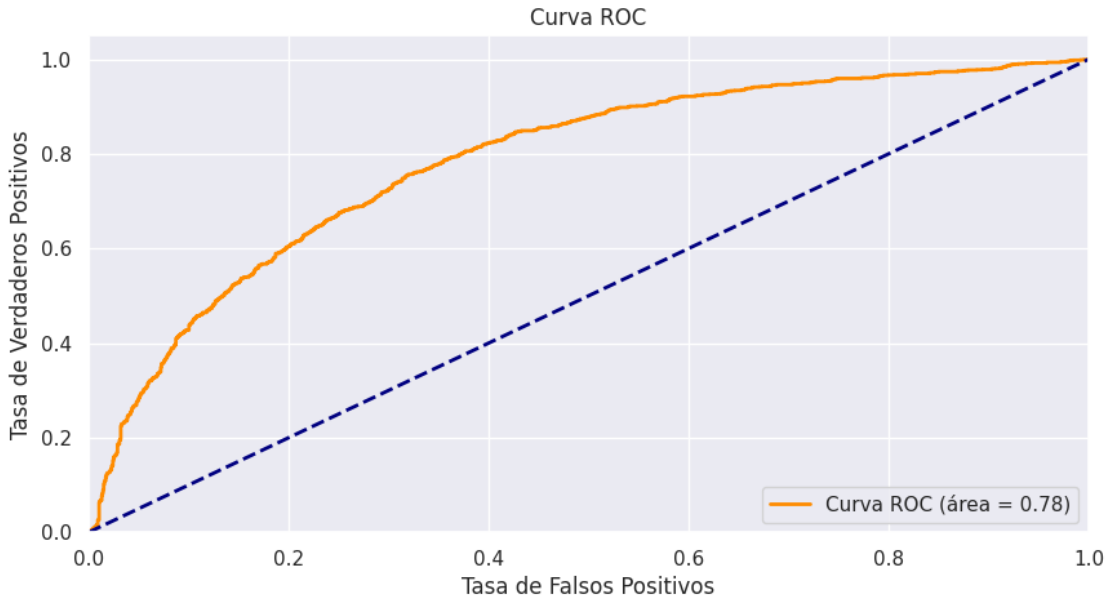
### 5.1.1 Curva Roc -- Clasificador Bayesiano¶

In [83]:

```
#Curva ROC
# Calcular la probabilidad de las predicciones
y_score = cb.predict_proba(X_test)

# Calcular la curva ROC
fpr, tpr, thresholds = roc_curve(y_test, y_score[:,1], pos_label=1)
roc_auc = auc(fpr, tpr)

# Graficar la curva ROC
plt.figure()
plt.plot(fpr, tpr, color='darkorange', lw=2, label='Curva ROC (área = %0.2f)' % roc_auc)
plt.plot([0, 1], [0, 1], color='navy', lw=2, linestyle='--')
plt.xlim([0.0, 1.0])
plt.ylim([0.0, 1.05])
plt.xlabel('Tasa de Falsos Positivos')
plt.ylabel('Tasa de Verdaderos Positivos')
plt.title('Curva ROC')
plt.legend(loc="lower right")
plt.show()
```




### 5.1.2 Matriz de Confusión -- Clasificador Bayesiano¶

In [84]:
```
confusion_matrix(y_test, y_pred_cb)
```

Out[84]:
```
array([[ 712,  812],
       [ 156, 1374]])
```

In [85]:
```
(712+1374)/(712+1374+812+156)
```

Out[85]:
```
0.6830386378519974
```

### *5.2 Máquinas de Soporte Vectorial*¶

In [86]:
```
# Hacer predicciones en el conjunto de prueba
y_pred_msv = msv.predict(X_test)
y_prob_msv = msv.predict_proba(X_test)
```

In [87]:
```
# Calcular la precisión (accuracy) de las predicciones
accuracy = accuracy_score(y_test, y_pred_msv)
precision = precision_score(y_test, y_pred_msv, average='macro')
recall = recall_score(y_test, y_pred_msv, average='macro')
```

In [88]:

```
# Imprimir las métricas de rendimiento
print('Exactitud:', accuracy)
print('Precisión:', precision)
print('Sensibilidad:', recall)
```

```
Exactitud: 0.731827111984283
Precisión: 0.7327610156051805
Sensibilidad: 0.7317636765992486
```

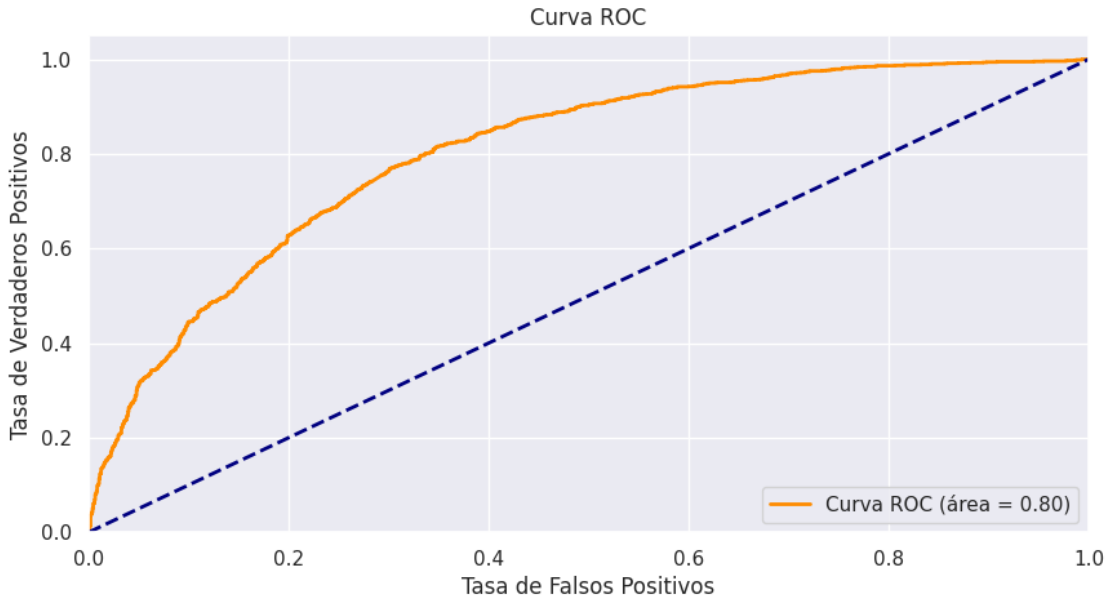
### 5.2.1 Curva Roc -- Maquinas de Soporte Vectorial¶

In [89]:

```
# Calcular probabilidad de predicción para cada clase
y_score = msv.predict_proba(X_test)

# Calcular la curva ROC
fpr, tpr, thresholds = roc_curve(y_test, y_score[:,1], pos_label=1)
roc_auc = auc(fpr, tpr)

# Graficar la curva ROC
plt.figure()
plt.plot(fpr, tpr, color='darkorange', lw=2, label='Curva ROC (área = %0.2f)' % roc_auc)
plt.plot([0, 1], [0, 1], color='navy', lw=2, linestyle='--')
plt.xlim([0.0, 1.0])
plt.ylim([0.0, 1.05])
plt.xlabel('Tasa de Falsos Positivos')
plt.ylabel('Tasa de Verdaderos Positivos')
plt.title('Curva ROC')
plt.legend(loc="lower right")
plt.show()
```




### 5.2.2 Matrix de Confusión -- Maquinas de Soporte Vectorial¶

In [90]:
```
confusion_matrix(y_test, y_pred_msv)
```

Out[90]:
```
array([[1066,  458],
       [ 361, 1169]])
```

In [93]:
```
(1066+1170)/(1066+1170+458+360)
```

Out[93]:
0.7321545514079896

### *5.3 Regresión Logística*¶

In [91]:
```
# Hacer predicciones en el conjunto de prueba
y_pred_rl = rl.predict(X_test)
y_prob_rl = rl.predict_proba(X_test)
```

In [92]:
```
# Calcular la precisión (accuracy) de las predicciones
accuracy = accuracy_score(y_test, y_pred_rl)
precision = precision_score(y_test, y_pred_rl, average='macro')
recall = recall_score(y_test, y_pred_rl, average='macro')
```

In [94]:

```
# Imprimir las métricas de rendimiento
print('Exactitud:', accuracy)
print('Precisión:', precision)
print('Sensibilidad:', recall)
```

```
Exactitud: 0.7288801571709234
Precisión: 0.7290845256657714
Sensibilidad: 0.7288495188101487
```

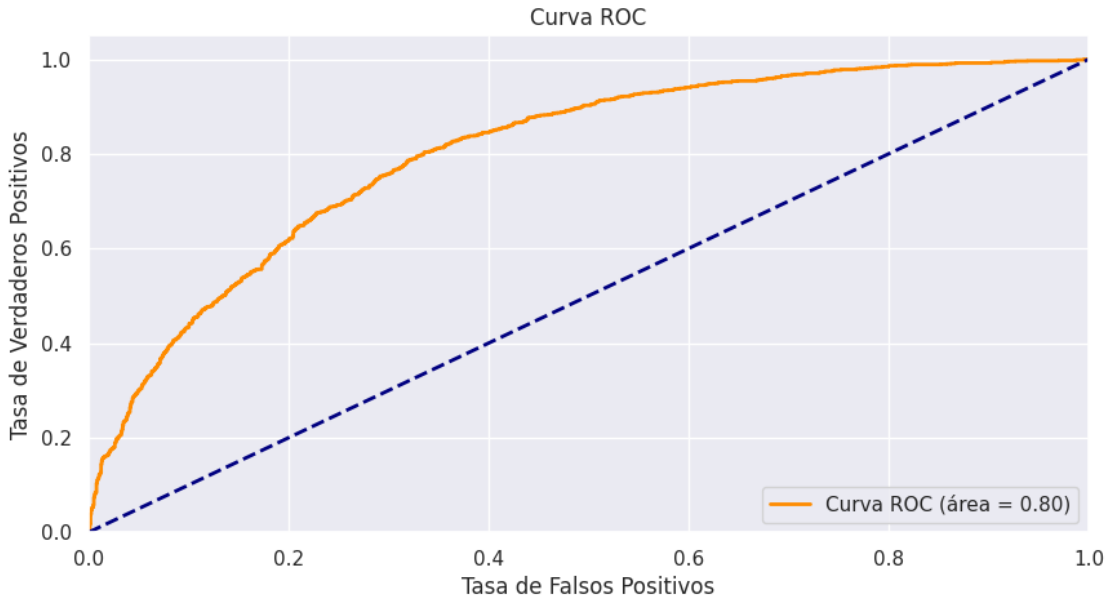
### 5.3.1 Curva Roc -- Regresión Logística¶

In [95]:

```
# Calcular probabilidad de predicción para cada clase
y_score = rl.predict_proba(X_test)

# Calcular la curva ROC
fpr, tpr, thresholds = roc_curve(y_test, y_score[:,1], pos_label=1)
roc_auc = auc(fpr, tpr)

# Graficar la curva ROC
plt.figure()
plt.plot(fpr, tpr, color='darkorange', lw=2, label='Curva ROC (área = %0.2f)' % roc_auc)
plt.plot([0, 1], [0, 1], color='navy', lw=2, linestyle='--')
plt.xlim([0.0, 1.0])
plt.ylim([0.0, 1.05])
plt.xlabel('Tasa de Falsos Positivos')
plt.ylabel('Tasa de Verdaderos Positivos')
plt.title('Curva ROC')
plt.legend(loc="lower right")
plt.show()
```




### 5.3.1 Matrix de Confusión -- Regresión Logística¶

In [96]:
```
confusion_matrix(y_test, y_pred_rl)
```

Out[96]:
```
array([[1087,  437],
       [ 391, 1139]])
```

In [97]:
```
(1087+1139)/(1087+1139+437+391)
```

Out[97]:
```
0.7288801571709234
```

## 6. Deployment¶

In [98]:
```
df_val
```

| | ID | ANIO | SINIESTROS | LESIONADOS | FALLECIDOS | ENTE_DE_CONTROL | LATITUD_Y | LONGITUD_X | DPA_1 | PROVINCIA | ... | TRICIMOTO |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| **139156** | 139157 | 2023 | MCU00001012023 | 1 | 0 | EMPRESA PUBLICA DE MOVILIDAD, TRANSITO Y TRANS... | -2.900646 | -78.984631 | 1 | AZUAY | ... | 0 |
| **139158** | 139159 | 2023 | DMQ00002012023 | 0 | 0 | AGENCIA METROPOLITANA DE TRANSITO DE QUITO - AMT | -0.284215 | -78.516339 | 17 | PICHINCHA | ... | 0 |
| **139161** | 139162 | 2023 | MAM00005012023 | 3 | 0 | DIRECCION DE TRANSITO, TRASPORTE TERRESTRE Y S... | -1.252839 | -78.629924 | 18 | TUNGURAHUA | ... | 0 |
| **139162** | 139163 | 2023 | DMQ00006012023 | 6 | 0 | AGENCIA METROPOLITANA DE TRANSITO DE QUITO - AMT | -0.057577 | -78.464477 | 17 | PICHINCHA | ... | 0 |
| **139164** | 139165 | 2023 | PNE00001012023 | 0 | 1 | POLICIA NACIONAL DEL ECUADOR | -0.901179 | -80.237069 | 13 | MANABI | ... | 0 |
| **...** | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... |
| **142577** | 142578 | 2023 | MPO00671022023 | 0 | 0 | EMPRESA PUBLICA MUNICIPAL DE TRANSITO, TRANSPO... | -1.063441 | -80.456982 | 13 | MANABI | ... | 0 |
| **142578** | 142579 | 2023 | CTE00485022023 | 1 | 0 | COMISION DE TRANSITO DEL ECUADOR - CTE | -1.006694 | -80.686833 | 13 | MANABI | ... | 0 |
| **142579** | 142580 | 2023 | DMQ00672022023 | 0 | 0 | AGENCIA METROPOLITANA DE TRANSITO DE QUITO - AMT | -0.187918 | -78.491713 | 17 | PICHINCHA | ... | 0 |

In [99]:

```
# guardar modelo
from joblib import dump, load
dump(msv, 'SVM_Bank.joblib')
```

Out[99]:

```
['SVM_Bank.joblib']
```

In [100]:

```
# cargar modelo
loaded_model = load('SVM_Bank.joblib')
```

In [101]:

```
df_val.reset_index(inplace=True)
```

In [102]:

```
df_val = df_val.loc[:,["PROVINCIA", "ZONA", "PERIODO_2","DIA_2","MES_2","FERIADO
","CODIGO_CAUSA", "TIPO_DE_SINIESTRO", "TIPO_DE_VEHICULO_1", "SERVICIO_1
", "SEXO_1","PARTICIPANTE_1","CINTURON_1","CASCO_1","ANIO",'AUTOMOVIL','BI
CICLETA','BUS','CAMION','CAMIONETA','EMERGENCIAS','ESPECIAL','FURGONETA','M
OTOCICLETA','NO_IDENTIFICADO','SCOOTER_ELECTRICO','TRICIMOTO','VEHICULO
_DEPORTIVO_UTILITARIO', "SUMA_DE_VEHICULOS", "EDAD_1","CONDICION_1"]]
df_val
```

| | PROVINCIA | ZONA | PERIODO_2 | DIA_2 | MES_2 | FERIADO | CODIGO_CAUSA | TIPO_DE_SINIESTRO | TIPO_DE_VEHICULO_1 | SERVICIO_1 | ... | ESPECI |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | AZUAY | URBANA | 0 | 7 | 1 | SI | C09 | PERDIDA DE CARRIL | AUTOMOVIL | PARTICULAR | ... | |
| 1 | PICHINCHA | URBANA | 0 | 7 | 1 | SI | C10 | CHOQUE LATERAL | FURGONETA | PARTICULAR | ... | |
| 2 | TUNGURAHUA | URBANA | 0 | 7 | 1 | SI | C09 | ATROPELLOS | NO IDENTIFICADO | PARTICULAR | ... | |
| 3 | PICHINCHA | RURAL | 0 | 7 | 1 | SI | C06 | ATROPELLOS | AUTOMOVIL | PARTICULAR | ... | |
| 4 | MANABI | RURAL | 1 | 7 | 1 | SI | C09 | ESTRELLAMIENTOS | NO IDENTIFICADO | PARTICULAR | ... | |
| ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | |
| 2187 | MANABI | URBANA | 19 | 2 | 2 | NO | C23 | CHOQUE POSTERIOR | AUTOMOVIL | PARTICULAR | ... | |
| 2188 | MANABI | RURAL | 21 | 2 | 2 | NO | C18 | CHOQUE FRONTAL | AUTOMOVIL | PARTICULAR | ... | |
| 2189 | PICHINCHA | URBANA | 21 | 2 | 2 | NO | C12 | OTROS | MOTOCICLETA | PARTICULAR | ... | |
| 2190 | GUAYAS | URBANA | 22 | 2 | 2 | NO | C11 | CHOQUE POSTERIOR | NO IDENTIFICADO | PARTICULAR | ... | |
| 2191 | GUAYAS | URBANA | 23 | 2 | 2 | NO | C14 | ESTRELLAMIENTOS | MOTOCICLETA | PARTICULAR | ... | |

2192 rows × 31 columns

In [103]:

```
# Preprocesamiento de datos
X = df_val.drop("CONDICION_1", axis=1)  # Variables predictoras
y = df_val["CONDICION_1"]  # Variable objetivo
```

In [104]:

```
X
```

Out[104]:

| | PROVINCIA | ZONA | PERIODO_2 | DIA_2 | MES_2 | FERIADO | CODIGO_CAUSA | TIPO_DE_SINIESTRO | TIPO_DE_VEHICULO_1 | SERVICIO_1 | ... | EMERG |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | AZUAY | URBANA | 0 | 7 | 1 | SI | C09 | PERDIDA DE CARRIL | AUTOMOVIL | PARTICULAR | ... | |
| 1 | PICHINCHA | URBANA | 0 | 7 | 1 | SI | C10 | CHOQUE LATERAL | FURGONETA | PARTICULAR | ... | |
| 2 | TUNGURAHUA | URBANA | 0 | 7 | 1 | SI | C09 | ATROPELLOS | NO IDENTIFICADO | PARTICULAR | ... | |
| 3 | PICHINCHA | RURAL | 0 | 7 | 1 | SI | C06 | ATROPELLOS | AUTOMOVIL | PARTICULAR | ... | |
| 4 | MANABI | RURAL | 1 | 7 | 1 | SI | C09 | ESTRELLAMIENTOS | NO IDENTIFICADO | PARTICULAR | ... | |
| ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | |
| 2187 | MANABI | URBANA | 19 | 2 | 2 | NO | C23 | CHOQUE POSTERIOR | AUTOMOVIL | PARTICULAR | ... | |
| 2188 | MANABI | RURAL | 21 | 2 | 2 | NO | C18 | CHOQUE FRONTAL | AUTOMOVIL | PARTICULAR | ... | |
| 2189 | PICHINCHA | URBANA | 21 | 2 | 2 | NO | C12 | OTROS | MOTOCICLETA | PARTICULAR | ... | |
| 2190 | GUAYAS | URBANA | 22 | 2 | 2 | NO | C11 | CHOQUE POSTERIOR | NO IDENTIFICADO | PARTICULAR | ... | |
| 2191 | GUAYAS | URBANA | 23 | 2 | 2 | NO | C14 | ESTRELLAMIENTOS | MOTOCICLETA | PARTICULAR | ... | |

2192 rows × 30 columns

In [105]:

```
# Codificar variables categóricas
categorical_features = ["PROVINCIA", "ZONA","FERIADO","CODIGO_CAUSA", "TIPO_D
E_SINIESTRO", "TIPO_DE_VEHICULO_1", "SERVICIO_1", "SEXO_1","PARTICIPANTE_1
","CINTURON_1","CASCO_1"]
label_encoders = {}
for feature in categorical_features:
```

```
    label_encoders[feature] = LabelEncoder()
    X[feature] = label_encoders[feature].fit_transform(X[feature])
```

In [106]:

```
# Normalizar variables numéricas , "PERIODO_1","DIA_1","MES_1"
numerical_features = ["ANIO",'AUTOMOVIL',"PERIODO_2","DIA_2","MES_2",'BICICLETA
',"BUS",'CAMION','CAMIONETA','EMERGENCIAS','ESPECIAL','FURGONETA','MOTOCIC
LETA','NO_IDENTIFICADO','SCOOTER_ELECTRICO','TRICIMOTO','VEHICULO_DEPOR
TIVO_UTILITARIO', "SUMA_DE_VEHICULOS", "EDAD_1"]
scaler = StandardScaler()
X[numerical_features] = scaler.fit_transform(X[numerical_features])
```

In [107]:

```
# prediccion
y_pred_msv = loaded_model.predict(X)
y_prob_msv = loaded_model.predict_proba(X)
```

In [108]:

```
# transformo a dataframe la probabilidad
df_bin_val = pd.DataFrame(y_prob_msv[:, 1], columns=['Probabilidad'])
```

In [109]:

```
y_pred_msv
```

Out[109]:

```
array([1, 0, 1, ..., 0, 0, 0])
```

In [110]:

```
from collections import Counter

# Contar la frecuencia de cada elemento en el arreglo
frecuencia = Counter(y_pred_msv)

# Imprimir el resultado
for elemento, cantidad in frecuencia.items():
    print("Elemento:", elemento, "- Cantidad:", cantidad)
```

```
Elemento: 1 - Cantidad: 1041
Elemento: 0 - Cantidad: 1151
```