Entrega 2 – Proyecto

Integrantes:

Sergio Rojas
Cristhian Amaya
Andrés Beltrán
Gloria Ramos

Enlace del repositorio donde se encuentran todos los soportes:
https://github.com/SergioRojas86/Proyecto_final_Despliegue_de_soluciones_analiticas

¿Cuál es la probabilidad de que haya un accidente fatal en un accidente automovilístico en México?

❑ Resumen sobre el problema, pregunta, alcance y datos

México, un país latinoamericano con una extensa red de carreteras y alta movilidad, enfrenta una creciente preocupación debido al aumento de accidentes automovilísticos, registrándose alrededor de 296,000 en 2021, con más de 23,000 fatalidades, marcando un preocupante incremento respecto a años anteriores. Diversos factores contribuyen a esta situación, como el exceso de velocidad, consumo de alcohol, falta de cumplimiento de normas de tránsito y condiciones de las carreteras. La falta de implementación efectiva de políticas de seguridad vial también ha dejado una huella significativa.

Los impactos social y económico son notables, afectando a familias y comunidades, generando costos médicos y pérdidas de productividad. Frente a este escenario, surge la pregunta central del estudio: *¿Cuál es la probabilidad de que algún implicado peatonal, pasajero o conductor resulte en deceso en un accidente automovilístico?*

Este estudio busca abordar y reducir las consecuencias devastadoras de los accidentes automovilísticos en el país. Utilizando datos del Instituto Nacional de Estadística y Geografía (INEGI), se pretende analizar las tendencias específicas de 2021. El análisis de datos, respaldado por el Sistema de Información de la Comisión Nacional de Seguridad (SICATUS), permitirá identificar patrones y factores de riesgo asociados a accidentes fatales. Este enfoque proporcionará una base sólida para decisiones informadas y promoción de prácticas de conducción más seguras en México.

Para esto, el alcance del proyecto está dado por: 1). Recopilación de datos, 2). Limpieza y preparación de los datos, 3). Modelado de los datos, 4) Identificación de factores de riesgo, y finalmente, 5) Informe final.

De igual forma, para resolver esta pregunta de negocio se utilizarán fuentes de datos originadas por INEGI por medio de un cuestionario de accidentes de tránsito la cual recopila dentro de las 46 variables cuatro tipos de datos, registro de localización (código de municipio, zona urbana y zona suburbana), registro de hora y fecha en que sucede el accidente (Año, hora, minutos, día de la semana, día), datos del conductor y datos del vehículo. El nombre de la fuente de datos original es “Estadística de Accidentes de Tránsito Terrestre en Zonas Urbanas y Suburbanas”. Adicional se cuenta con una tabla de referencia de los Municipios de

México, que tiene información del código del municipio y el nombre asociado. Esta base es generada por el Instituto Nacional para el Federalismo y Desarrollo Municipal.

- ❑ **Modelos desarrollados y su evaluación**

Para las pruebas y desarrollos de los modelos, Durante la realización del análisis exploratorio de datos se realizaron preparación de los datos, dando como resultado un conjunto de datos con las siguientes características:

- El conjunto de datos comprende 52 columnas y 340.415 registros.

- No comprende 15.999 registros con datos nulos, los cuales fueron limpiados durante el análisis.

- 33 Columnas comprenden la codificación de 9 variables transformadas a dummies.

- 18 variables con el valor de entero.

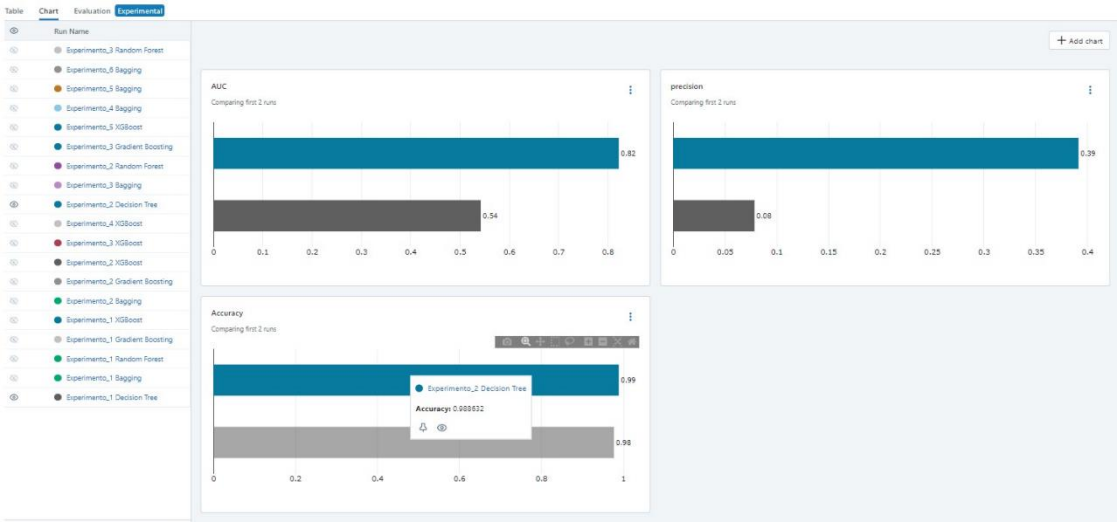
- La variable a predecir es 'CLASACC', la cual se encuentra convertida a variable dummie y representa el 1,1% del total de los datos.

Finalmente, se divide la base en 67% para entrenamiento y el 33% para el test y se realizan las pruebas con los siguientes modelos:

- ✓ **DecisionTreeClassifier**

Este modelo se basa en la construcción de un árbol de decisiones, donde cada nodo interno representa una característica o atributo, cada rama representa una decisión basada en esa característica, y cada hoja representa la etiqueta de clasificación. Durante el entrenamiento, el modelo selecciona las características que mejor dividen los datos en clases, utilizando criterios el accuracy. Una vez construido, el árbol puede clasificar nuevas instancias siguiendo el camino desde la raíz hasta una hoja. Los DecisionTreeClassifiers son conocidos por su interpretabilidad y capacidad para manejar conjuntos de datos complejos, aunque también pueden propensos al sobreajuste si no se controlan adecuadamente.

Al realizarse la prueba sobre el conjunto de datos, se observa que para el modelo DesicionTreeClassifier, con los parámetros establecidos de max_depth = 5, y una semilla =123, el mejor modelo fue el Experimento_2, con un Accuracy del 98%, una precisión de 39% y un AUC de 82%.



Parameters

Show diff only

Model Name	Experimento_2 Decision Tree	Experimento_1 Decision Tree
max_depth	5	None
max_features	None	None
semilla	123	123

Metrics

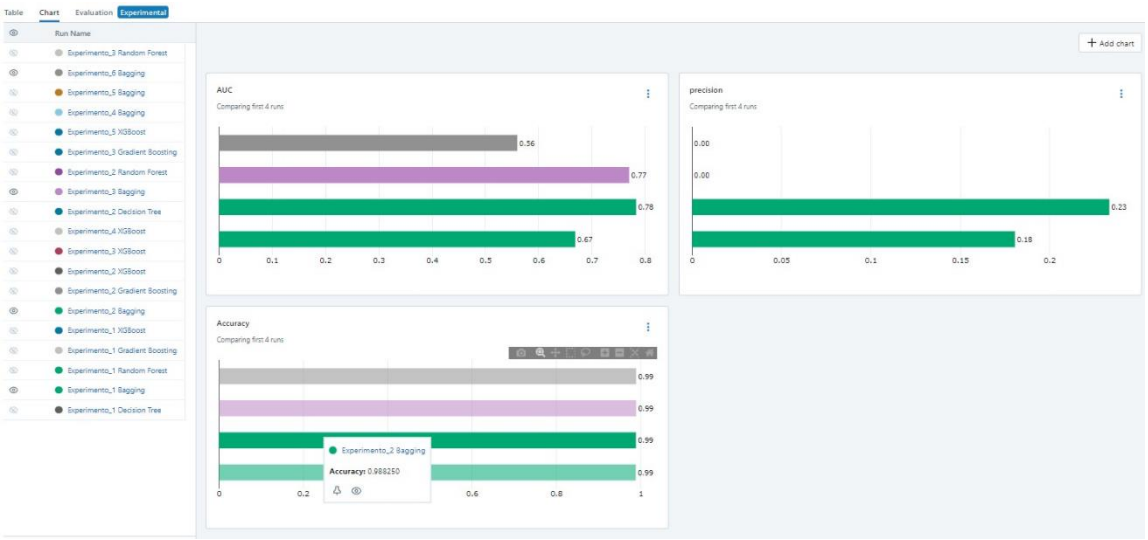
Show diff only

AUC	0.822	0.542
Accuracy	0.989	0.977
precision	0.391	0.078

- ✓ **BaggingClassifier**

El baggingClassifier utiliza la técnica de "bagging" (bootstrap aggregating) para mejorar la precisión y la robustez de los modelos de base. En particular, toma como entrada un clasificador base (como un DecisionTreeClassifier) y crea múltiples instancias de este clasificador entrenándolas en subconjuntos aleatorios del conjunto de entrenamiento original mediante el muestreo con reemplazo. Luego, combina las predicciones de estos clasificadores base mediante votación para producir una predicción final. La técnica de bagging ayuda a reducir la varianza y mejorar la generalización del modelo, especialmente cuando el clasificador base tiende a ser sensible al ruido o al sobreajuste. El BaggingClassifier es eficaz en la reducción de la variabilidad del modelo, lo que lo hace útil para mejorar el rendimiento en conjuntos de datos complejos.

Al realizarse las 4 pruebas sobre este modelo como se evidencia en las gráficas, se muestra que con unos hiperparámetros de max_features = 1, max_samples=1, n_estimators=100 , semilla = 123. Los resultados arrojaron al experimento número 2 como el mejor modelo de las pruebas de Bagging Classifier, dando como resultado un AUC de 78%, un Accuracy del 98% y una precisión del 23%.



Parameters

Show diff only

Model Name	Experimento_6 Bagging	Experimento_3 Bagging	Experimento_2 Bagging	Experimento_1 Bagging
max_features	3	1.0	1.0	1.0
max_samples	3	5	1.0	1.0
n_estimators	100	200	100	10
semilla	123	123	123	123

Metrics

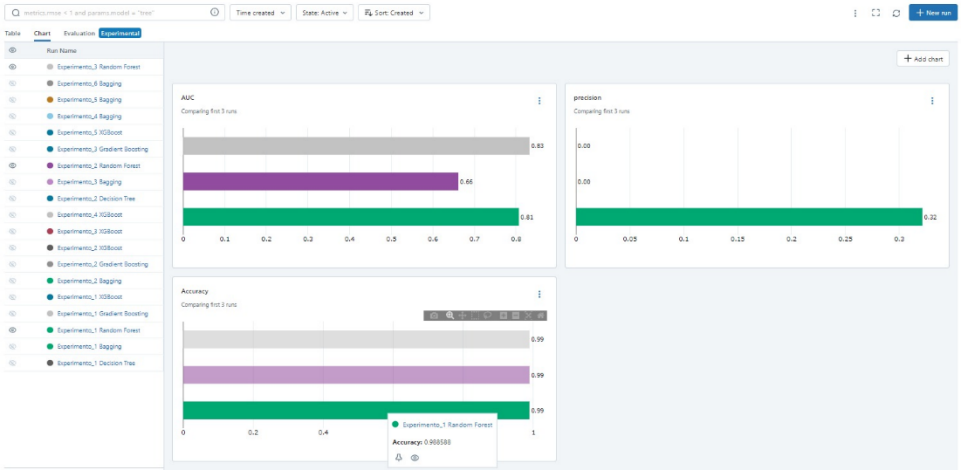
Show diff only

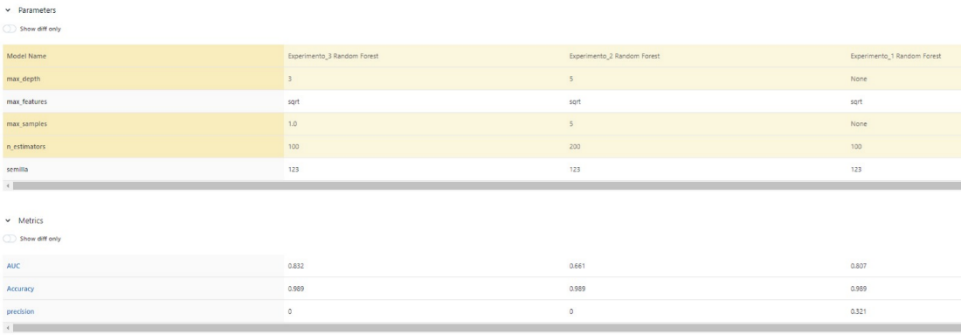
AUC	0.56	0.77	0.783	0.668
Accuracy	0.989	0.989	0.988	0.988
precision	0	0	0.233	0.181

- ✓ **RandomForestClassifier**

Este clasificador utiliza múltiples árboles de decisión durante el entrenamiento y la predicción. Durante el proceso de construcción de cada árbol, se selecciona aleatoriamente un subconjunto de características y se realiza el entrenamiento con un subconjunto de datos de entrenamiento (muestreo bootstrap). Luego, las predicciones de los árboles individuales se combinan mediante clasificación para determinar la clase final de una instancia. El RandomForestClassifier es conocido por su capacidad para manejar conjuntos de datos grandes, manejar características irrelevantes y reducir el riesgo de sobreajuste, lo que lo convierte en un modelo robusto y efectivo para clasificación en diversas aplicaciones.

Para el random forest, se realizaron 3 experimentos, de los cuales, el mejor fue el experimento número 1, con unos hiperparámetros de n_estimators = 100 y una semilla = 123, dando como resultado un AUC de 80%, un accuracy de 98% y una precisión del 32%

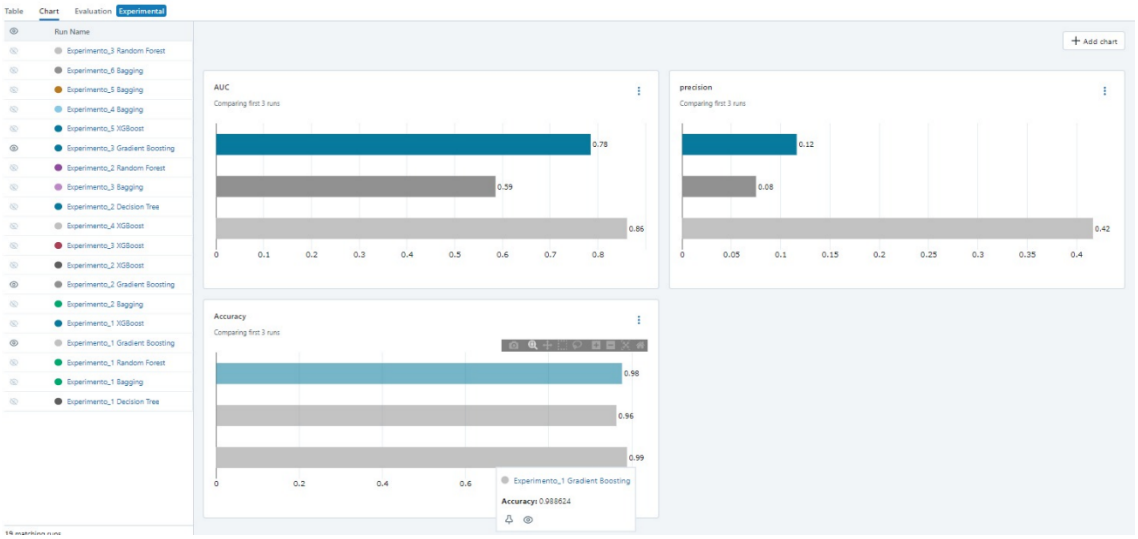




- ✓ **GradientBoostingClassifier**

Este algoritmo de ensamble construye un modelo de predicción mediante la combinación de múltiples modelos, generalmente árboles de decisión. A diferencia de los métodos de bagging como el RandomForest, el GradientBoosting construye los árboles de manera secuencial, donde cada nuevo árbol se enfoca en corregir los errores de predicción del conjunto existente. En cada iteración, el modelo ajusta la predicción hacia la dirección correcta mediante la optimización de la función de pérdida. Esto se hace de manera gradual para minimizar el error global. El GradientBoostingClassifier es efectivo para problemas de clasificación, siendo especialmente útil en conjuntos de datos desequilibrados y para capturar relaciones complejas entre las características. Sin embargo, también es más propenso a sobre ajustarse si no se ajustan adecuadamente los hiperparámetros.

De este modelo, se realizaron tres pruebas, de las cuales, el mejor modelo fue el experimento número 1, con unos hiperparámetros de learning_rate = 0.1, max_depth = 3, n_estimators = 100 y una semilla =123, dando como resultado un AUC de 86%, un accuracy del 98% y una precisión del 41%.



Parameters

Show diff only

Model Name	Experimento_3 Gradient Boosting	Experimento_2 Gradient Boosting	Experimento_1 Gradient Boosting
learning_rate	1.0	1.0	0.1
max_depth	5	1	3
max_features	None	None	None
n_estimators	200	100	100
semilla	123	123	123

Metrics

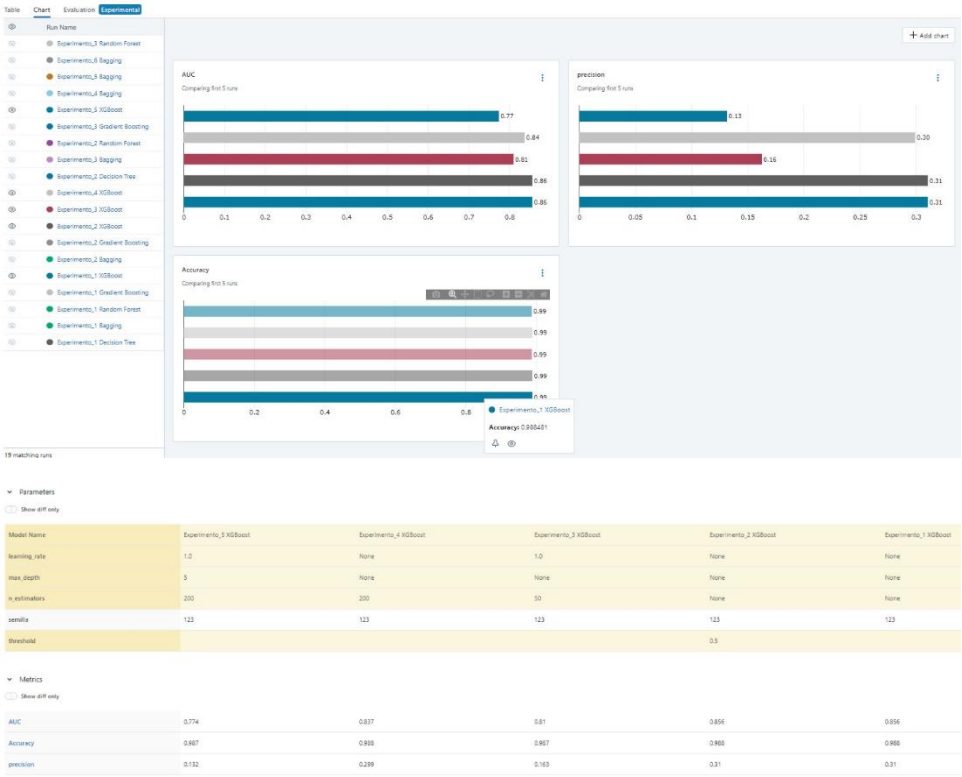
Show diff only

AUC	0.785	0.585	0.861
Accuracy	0.977	0.963	0.989
precision	0.116	0.075	0.417

- ✓ **XGBClassifier**

Finalmente, El XGBClassifier implementa técnicas como la poda de árboles, muestreo por columnas y la asignación ponderada de errores para mejorar la precisión y generalización del modelo. Además, ofrece opciones para ajustar la velocidad de aprendizaje y controlar la complejidad del modelo. XGBoost es conocido por su eficiencia computacional y su capacidad para manejar grandes conjuntos de datos, lo que lo convierte en una opción popular para problemas de clasificación en la comunidad de machine learning.

De este modelo, se realizaron tres pruebas, de las cuales, el mejor fue el experimento 1, con hipermarápetros de semilla = 123 dio como resultados un AUC de 85%, accuracy de 98% y una precisión de 29%.



❑ Observaciones y conclusiones sobre los modelos

Al comparar los mejores modelos de los experimentos utilizados en cada uno, se observa que todos tuvieron resultados similares en el AUC y en el Accuracy, con resultados sobre el 90%, sin embargo, lo que hizo la diferencia en el desempeño de la precisión, la cual fue mejor en el modelo GRADIENT BOOSTING, quien a diferencia de los demás experimentos tuvo una precisión del 41%.

❑ Tablero Desarrollado

Url del tablero: xxxxxxxxxxxxxxxxxxxxxxxxxxx

El tablero esta realizado con 'Dash', este es un framework de Python el cual ayudo a generar las gráficas interactivas, este se ejecuta a través de un archivo llamado "application.py" el cual se alimenta de las mismas carpetas dentro del aplicativo:

- Pages: contiene el código de las dos páginas y los gráficos a utilizar.
- Data: contiene los datos exgtraidos y utilizados para hacer las gráficas, este es la principal fuente de información para realizar las gráficas que se encuentran en el análisis descriptivo.
    - DIASEMANA_COUNT: csv extraido de la data principal el cual genera grafica de accidentes con sus fallecidos y heridos por dia de la semana
    - GeneroEdad: Datos para la generación de la pirámide de edad de accidentes
    - mapa_estado: csv extraído de la data principal el cual alimenta la información geográfica.
    - SEXO_count Datos para la generación del pie de genero
    - TIPACCD_count: Datos para la generación del filtro del gráfico de barras.

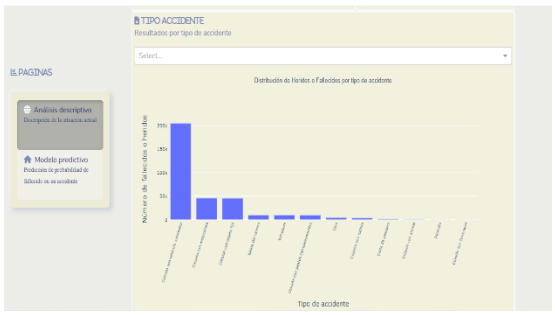
Estos datos se generan a partir de la extracción de los archivos principales y el análisis descriptivo realizado que se encuentra en el repositorio del proyecto.

- Componnets: esta carpeta contiene kpi (una opción para crear graficos), Maps (contiene el mapa de México y el código que alimenta las coordenadas del País y sus municipios), prediction(carpeta para adicionar el modelo o componentes en el futuro).

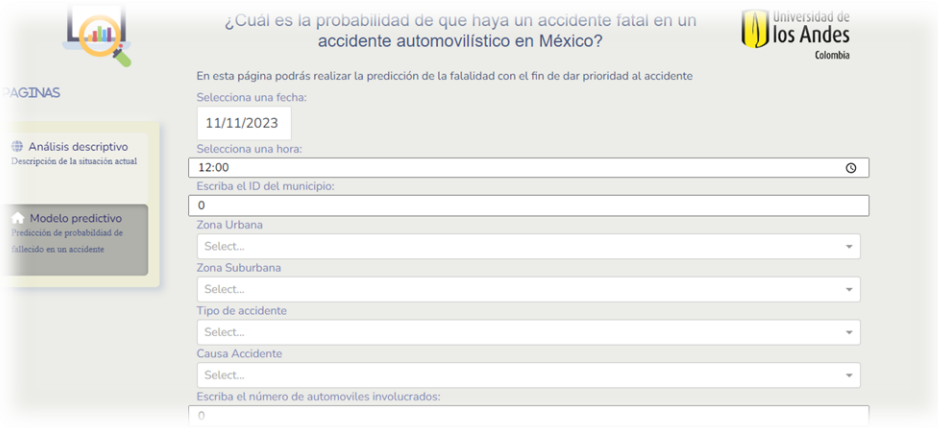
En la primera hoja del Dash , podemos observar en la parte izquierda el menú de navegación de las dos hojas:

- Hoja 1 - Análisis descriptivo: contiene un total de 5 gráficos que describen la data que alimenta el modelo y el comportamiento de los accidentes de tránsito en México, en primer lugar, un mapa que muestra los fallecidos y su número según la encuesta aplicada por los datos, en segundo lugar, la edad y genero del conductor del vehículo.

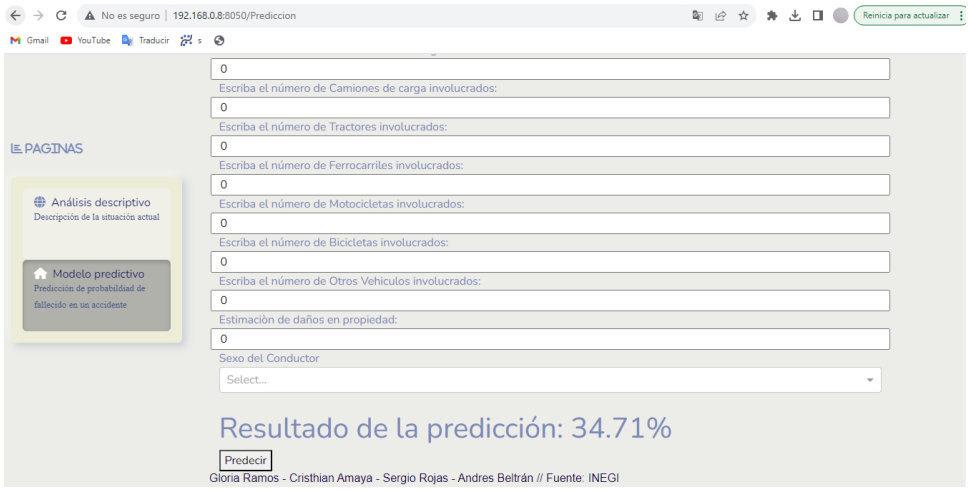
  Una tercera gráfica que a partir de un filtro muestra los heridos o fallecidos según la selección del usuario por cada uno de los tipos de accidente y al final una gráfica que muestra heridos y fallecidos en total según el día de la semana.



- Hoja 2 - Predicción: El objetivo de esta hoja es que usuario ingrese en los campos correspondientes las características del accidente como lo es la fecha y la hora (la fecha esta previamente diligenciada con la fecha actual), datos del lugar del accidente (Municipio, zona urbana, suburbana, etc.) y datos del accidente como vehículos de cada clase que estuvieron involucrados en el accidente



En la parte inferior se encuentra el botón de predicción que a partir de la información asignada generará la probabilidad de una muerte fatal en el accidente (Por el momento muestra un valor predeterminado). Con el fin de generar una priorización en la atención de emergencias en accidentes de tránsito.



❑ Reporte de trabajo en equipo

Para el desarrollo de esta entrega nos dividimos el trabajo en 3 frentes. El primero fue el desarrollo del modelo y documentación el cual fue liderado por Gloria Ramos. Se documentaron tanto el desarrollo del modelo final, como las fases de desarrollo de experimentos en MLFlow y el diseño del tablero. El segundo frente fue el desarrollo de los experimentos en MLFlow, del cual se encargó Sergio Rojas, en este punto, dados los resultados de indicadores de alta importancia, como AUC, precisión y accuracy, seleccionamos el modelo con mejor desempeño. El tercer frente fue el desarrollo preliminar del tablero con las dos secciones. La primera un análisis descriptivo de los datos presentados y la segunda sección es la de predicciones. Este tablero se desarrolló en dos partes, la primera la página en HTML y la segunda las gráficas presentadas en el análisis. De este último frente, Andres Beltran y Cristhian Amaya lideraron este desarrollo. Z

Adicionalmente, es importante mencionar que el integrante Sergio Rojas realizó estos commits, pero no se evidencia en el repositorio. Aparece en el Log de la actividad actual pero no en el Log general por un problema en el asignamiento de usuario.