Carlos Eduardo Correa-73139  
Joy Castañeda-63907  
Andres Camilo Velasquez-63111

**Siniestros viales**

**Objetivo Laboratorio**

Mejorar cualquiera de las siguientes métricas de desempeño de los modelos de

Arboles de decisión, Random Forest y Xgboost:

* Precisión
* Recall
* F1-score

**Desarrollo Laboratorio**

**Modelo Arboles de decisión**

**Código  
  
Texto

Descripción generada automáticamente**

**Resultados  
  
Imagen de la pantalla de un celular con letras y números

Descripción generada automáticamente con confianza media**

**Procesos Realizados**1. **Partición del Conjunto de Datos**Se particionan los datos en conjuntos de entrenamiento y prueba para evaluar el desempeño del modelo de manera efectiva.

* Se utiliza train\_test\_split para dividir los datos en conjuntos de entrenamiento y prueba:
* test\_size=0.30: Se reserva el 30% de los datos para el conjunto de prueba, mientras que el 70% restante se usa para el entrenamiento del modelo.
* random\_state=42: Se establece una semilla para asegurar que la partición de los datos sea reproducible en ejecuciones futuras.
* stratify=Y: Se asegura de que las proporciones de las clases en el conjunto de prueba sean las mismas que en el conjunto de entrenamiento, para mantener una distribución representativa de las clases en ambos conjuntos.

**2. Imputación de Valores Faltantes**

Se maneja la presencia de valores faltantes en los datos para que el modelo pueda ser entrenado sin errores.

* Se usa SimpleImputer con la estrategia de 'mean' para reemplazar valores faltantes:
* strategy='mean': Los valores faltantes se reemplazan con la media de cada columna, lo que ayuda a mantener la consistencia en los datos y mejora la calidad del entrenamiento del modelo.

**3. Definición y Entrenamiento del Modelo**

Se configura y entrena un modelo de árbol de decisión para predecir las clases en el conjunto de datos.

* Se define un DecisionTreeClassifier con parámetros específicos para mejorar el rendimiento del modelo:
* max\_depth=10: Se establece una profundidad máxima del árbol en 10 para evitar el sobreajuste y mejorar la generalización del modelo a nuevos datos.
* criterion='gini': Se utiliza el índice de Gini para medir la impureza de las particiones. Este criterio evalúa la calidad de una división en función de la pureza de las clases en cada nodo.
* random\_state=42: Se fija una semilla para que los resultados del modelo sean reproducibles en cada ejecución.

**4. Evaluación del Modelo**

Se evalúa el desempeño del modelo usando métricas de rendimiento y se visualiza la matriz de confusión para una mejor comprensión de los resultados.

* Se calculan las métricas de precisión, recall y F1-score:
* metrics.classification\_report: Proporciona un informe detallado con las métricas de precisión, recall y F1-score para cada clase, lo que permite evaluar la calidad general del modelo.
* Se visualiza la matriz de confusión para entender el desempeño del modelo en términos de verdaderos positivos, verdaderos negativos, falsos positivos y falsos negativos:
* confusion\_matrix: Crea una matriz que compara las etiquetas reales con las predicciones del modelo.
* Se usa seaborn para generar un heatmap de la matriz de confusión

**5. Resumen de Decisiones de Diseño**

* **Se estableció max\_depth=10** para prevenir el sobreajuste, limitando la complejidad del modelo y mejorando su capacidad de generalización a datos no vistos.
* **Se eligió criterion='gini'** sobre 'entropy' porque el índice de Gini es más eficiente computacionalmente y ofrece resultados similares en términos de impureza de las particiones.
* **Se fijó random\_state=42** para asegurar la reproducibilidad de los resultados del modelo.

Resumen del Código

Aquí está el código con comentarios explicativos, que muestra todos los pasos realizados:  
  
Texto

Descripción generada automáticamente  
  
Interfaz de usuario gráfica, Aplicación

Descripción generada automáticamente

**Modelo Random Forest**

**Código**  
  
Interfaz de usuario gráfica, Texto

Descripción generada automáticamente  
**Resultados:**  
  
Pantalla de celular con aplicaciones

Descripción generada automáticamente con confianza media  
**Procesos Realizados para Mejorar el Modelo RandomForestClassifier**

**1. Preparación de Datos**

* Convertimos y\_train a un array 1D para evitar el warning de DataConversionWarning que ocurre si y\_train no está en el formato correcto.
* Esta conversión es necesaria porque RandomForestClassifier requiere que y\_train sea un vector de una sola dimensión.

**2. Definición del Modelo**

* Se configura un Random Forest con un número mayor de árboles, estableciendo n\_estimators a 200. Esto mejora el rendimiento del modelo al tener más árboles en el bosque.
* Se establece una profundidad máxima de 20 (max\_depth) para prevenir el sobreajuste (overfitting) y mejorar la capacidad de generalización del modelo.
* Se fija el parámetro random\_state en 42 para asegurar que los resultados sean reproducibles en cada ejecución del modelo.

**3. Entrenamiento del Modelo**

* El modelo RandomForestClassifier se entrena usando los datos de características (X\_train) y las etiquetas correspondientes (y\_train).
* Este paso permite que el modelo aprenda patrones a partir de los datos para hacer predicciones sobre datos no vistos.

**4. Predicción de Nuevos Datos**

* Se hacen predicciones sobre el conjunto de prueba (X\_test) utilizando el modelo entrenado.
* Estas predicciones se comparan con las etiquetas verdaderas (y\_test) para evaluar el desempeño del modelo en datos que no fueron usados durante el entrenamiento.

**5. Evaluación del Modelo**

* Se genera un informe de clasificación que incluye métricas de evaluación como precisión, recall y F1-score.
* Estas métricas ayudan a entender el rendimiento del modelo en términos de su capacidad para identificar correctamente cada clase en el conjunto de prueba.

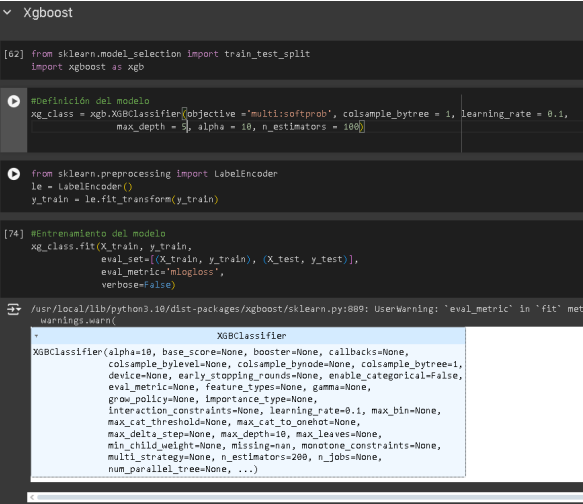
**6. Visualización de la Importancia de las Características**

* Se obtiene la importancia de las características del modelo para identificar cuáles variables tienen mayor impacto en las predicciones del modelo.
* Se crea un gráfico de barras que muestra la importancia de cada característica, ayudando a entender cuáles variables son más relevantes para la clasificación.  
    
  Texto

  Descripción generada automáticamente  
    
  Gráfico, Histograma

  Descripción generada automáticamente

**Modelo Xgboost**

**Código:  
  
Texto

Descripción generada automáticamente  
  
Resultados:   
Pantalla de computadora con letras

Descripción generada automáticamente con confianza media**  
  
**Procesos Realizados:**Se realiza pruebas en la definición del modelo para mejorar los valores de precisión y se establece lo siguiente:

**1. Definición del Modelo XGBoost**

* Configuración de parámetros: Se ajustan los parámetros del XGBClassifier de la siguiente manera:
* max\_depth: Se establece en 5 para limitar la profundidad máxima de los árboles, evitando el sobreajuste.
* n\_estimators: Se fija en 100 para especificar el número total de árboles en el modelo, balanceando rendimiento y tiempo de entrenamiento.
* learning\_rate: Se define en 0.1 para controlar la contribución de cada árbol al modelo final, afectando la velocidad de convergencia.
* alpha: Se ajusta a 10 para aplicar una penalización L1 a los coeficientes del modelo y reducir su complejidad.

**2. Preprocesamiento de Datos**

* Transformación de etiquetas: Se utiliza LabelEncoder para convertir las etiquetas de clase (y\_train) en valores numéricos. Esto es necesario para que el modelo XGBClassifier pueda procesar las etiquetas correctamente.

**3. Entrenamiento del Modelo**

* **Ajuste del modelo: Se entrena el modelo utilizando los datos de entrenamiento (X\_train, y\_train) y se evalúa su desempeño en un conjunto de validación usando eval\_set. Esto permite monitorear el rendimiento del modelo durante el entrenamiento y ajustar sus parámetros.**

**4. Validación Cruzada**

* Evaluación mediante k-fold: Se aplica validación cruzada con KFold para evaluar la estabilidad y el rendimiento general del modelo. Esto se realiza dividiendo los datos en múltiples pliegues y calculando métricas de evaluación en cada iteración.

**5. Evaluación del Modelo**

* Cálculo de métricas de desempeño: Se calculan métricas como precisión, recall y F1-score sobre el conjunto de prueba (X\_test, y\_test) para evaluar la capacidad predictiva del modelo en datos no vistos.

**6. Visualización de la Importancia de las Características**

* Análisis de características: Se visualiza la importancia de cada característica en el modelo mediante gráficos de barras. Esto proporciona información sobre qué variables tienen mayor impacto en las predicciones del modelo.
* Estos pasos reflejan un enfoque sistemático para definir, entrenar y evaluar un modelo XGBClassifier, con el objetivo de mejorar la precisión y asegurar su generalización en problemas de clasificación multiclase.

Texto

Descripción generada automáticamente  
Interfaz de usuario gráfica, Texto

Descripción generada automáticamente