Laboratorio 4-Sinistros Viales

Juan Sebastian Cortes Chia

Jhonatan Rios Tapiero

Santiago Guerrero Chamorro

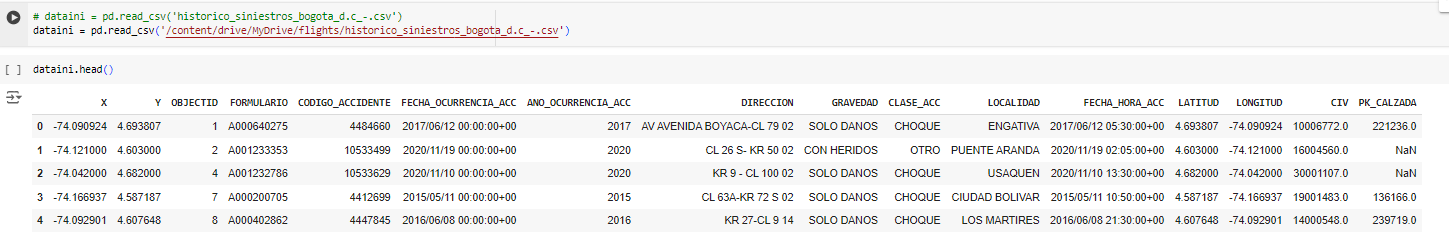
Seminario Big Data

Elias Buitrago Bolivar

Universidad ECCI

Facultad de ingeniería

Desarrollo

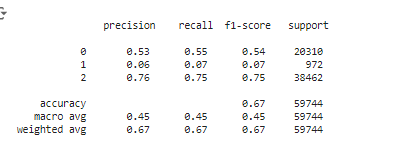
Lo primero que se realizó en este análisis fue importar lo datos proporcionados en un .csv, estos datos los podemos visualizar en una tabla usando la librería de pandas:  


También se analizan los datos importantes como lo son Gravedad, Clase del accidente, localidad, fecha, latitud y longitud. Estas variables serán muy importantes en nuestro análisis debido a que con estas variables podremos realizar la predicción de accidentes. 

Luego de realizar este proceso, y el proceso de transformación de datos, partición de datos para distribuirlos en los conjuntos de entrenamiento, prueba y validación. Se empiezan a realizar el análisis de los modelos supervisados:

Arboles de decisión:

Como resultado este modelo supervisado, podemos concluir lo siguiente a través de esta grafica:



* Clase 0: Tiene una precisión de 0.53, una recall de 0.55 y un f1-score de 0.54 con un soporte de 20310.
* Clase 1: Tiene una precisión de 0.06, una recall de 0.07 y un f1-score de 0.07 con un soporte de 972.
* Clase 2: Tiene una precisión de 0.76, una recall de 0.75 y un f1-score de 0.75 con un soporte de 38462.

Desempeño general del modelo:

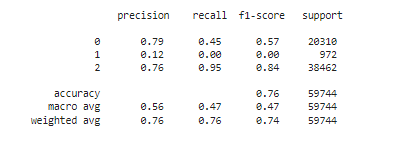
* Accuracy: La precisión global del modelo es 0.67.
* Macro avg: La media de las métricas de todas las clases (sin ponderar) es 0.45 para precisión, recall y f1-score.
* Weighted avg: La media ponderada de las métricas, que considera el soporte de cada clase, es 0.67 para precisión, recall y f1-score.

Como conclusión de este modelo podemos validar que:

* El modelo de árboles de decisión muestra un sesgo hacia la clase mayoritaria (Clase 2), ya que su rendimiento es significativamente mejor en esta clase.
* El modelo no maneja bien las clases minoritarias, especialmente la Clase 1.
* Podríamos considerar técnicas de balanceo de clases, como el sobre muestreo de las clases minoritarias o el submuestreo de la clase mayoritaria, para mejorar el rendimiento en las clases menos representadas.
* También podría ser útil explorar modelos más complejos o ajustar los hiperparámetros del modelo de árboles de decisión para mejorar el rendimiento global.

Random Forests

Como resultado este modelo supervisado, podemos concluir lo siguiente a través de esta gráfica:



* Clase 0: Tiene una precisión de 0.79, una recall de 0.45 y un f1-score de 0.57 con un soporte de 20310.
* Clase 1: Tiene una precisión de 0.12, una recall de 0.00 y un f1-score de 0.00 con un soporte de 972.
* Clase 2: Tiene una precisión de 0.76, una recall de 0.95 y un f1-score de 0.84 con un soporte de 38462

Desempeño general del modelo:

* Accuracy: La precisión global del modelo es 0.76.
* Macro avg: La media de las métricas de todas las clases (sin ponderar) es 0.56 para precisión, 0.47 para recall y 0.47 para f1-score.
* Weighted avg: La media ponderada de las métricas, que considera el soporte de cada clase, es 0.76 para precisión, recall y 0.74 para f1-score.

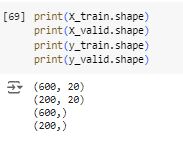
Como conclusión de este modelo podemos validar que:

* El modelo Random Forest muestra un buen rendimiento en la clase mayoritaria (Clase 2), similar al modelo de árboles de decisión.
* La Clase 1 no nos brinda un buen rendimiento debido a que cuenta con métricas extremadamente bajas, especialmente en recall (0.00), indicando que el modelo casi nunca identifica correctamente las instancias de esta clase.
* El rendimiento en la Clase 0 ha mejorado en términos de precisión en comparación con el modelo de árboles de decisión, pero el recall sigue siendo bajo, lo que sugiere que muchas instancias de la Clase 0 no se están clasificando correctamente.
* Se pueden considerar técnicas de balanceo de clases para mejorar el rendimiento en las clases menos representadas. Técnicas como SMOTE (Synthetic Minority Over-sampling Technique) o ajustes en los pesos de las clases durante el entrenamiento pueden ser útiles.
* Se pueden realizar exploraciones para la afinación de los hiperparámetros del modelo Random Forest para mejorar el rendimiento global.

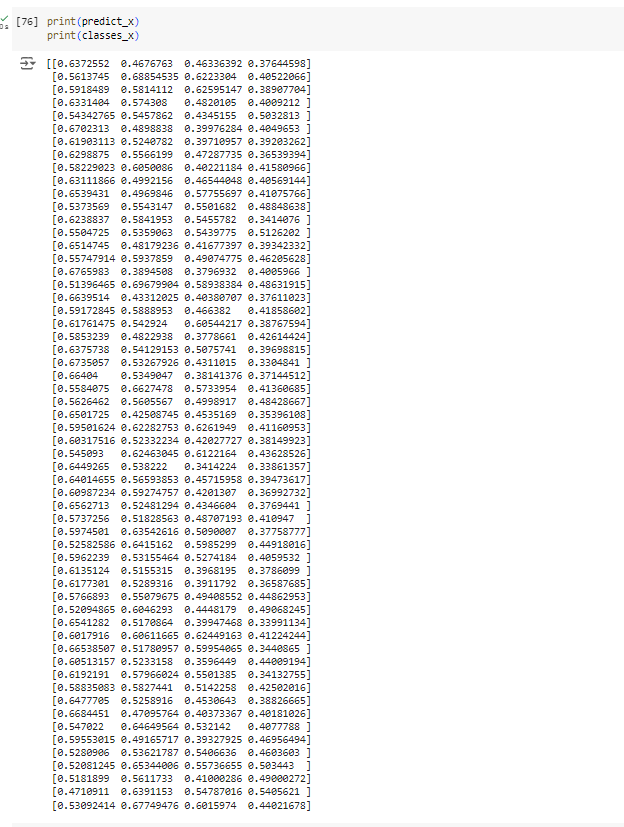
Redes Neuronales Artificiales

Al realizar el análisis con estas redes neunorales artificiales se puede determinar lo siguiente:

Lo primero que se realiza es entrenar el modelo en las siguientes variables:



* X\_train tiene una forma de (600, 20), lo que significa que hay 600 muestras en el conjunto de entrenamiento, cada una con 20 características.
* X\_valid tiene una forma de (200, 20), lo que significa que hay 200 muestras en el conjunto de validación, cada una con 20 características.
* y\_train tiene una forma de (600,), lo que significa que hay 600 etiquetas en el conjunto de entrenamiento, correspondientes a las 600 muestras.
* y\_valid tiene una forma de (200,), lo que significa que hay 200 etiquetas en el conjunto de validación, correspondientes a las 200 muestras



Esta imagen nos muestra la salida de una función llamada predict que parece ser el resultado de un modelo de red neuronal. Que se puede analizar de la siguiente manera:

* classes\_x: Este array contiene valores numéricos que van del 0 al 9, que probablemente representan las etiquetas de clase en una tarea de clasificación. En el contexto de una red neuronal, esto podría sugerir que el modelo se ha entrenado para clasificar los datos en 10 clases diferentes.
* Array sin etiqueta: Este array consta de números de punto flotante organizados en filas y columnas, posiblemente representando las probabilidades o puntuaciones que corresponden a cada etiqueta de clase para diferentes instancias. En un modelo de red neuronal, estas podrían ser las probabilidades de salida del modelo para cada clase.

En términos de lo que se puede validar con este conjunto de datos en una red neuronal, hay varias posibilidades como las que vemos a continuación, para proyectos futuros:

* Precisión del modelo: Podemos comparar las predicciones del modelo (es decir, las clases predichas) con las etiquetas reales para calcular la precisión del modelo.
* Matriz de confusión: Podemos crear una matriz de confusión para visualizar el rendimiento del modelo, lo que te permitirá ver fácilmente cuántas instancias se clasificaron correctamente y cuántas se clasificaron incorrectamente.
* Curvas ROC y AUC: Podemos trazar las curvas ROC para cada clase y calcular el AUC para obtener una medida de cuán bien el modelo puede distinguir entre las diferentes clases.
* Validación cruzada: Podemos realizar una validación cruzada para evaluar la robustez del modelo. Esto implica dividir el conjunto de datos en varios subconjuntos y entrenar y probar el modelo en diferentes combinaciones de estos subconjuntos.

Con base a los datos de precisión de la red neuronal podemos inferir lo siguiente:

Precisión (Precision):

* La precisión mide la proporción de verdaderos positivos sobre el total de positivos predichos (verdaderos positivos + falsos positivos).
* Precisión para la clase 0: 0.3214 (aproximadamente 32%)
* Precisión para la clase 1: 0.3043 (aproximadamente 30%)
* Precisión para la clase 2: 0.2143 (aproximadamente 21%)
* La clase 3 no tiene predicciones (0)

Recall:

* El recall (sensibilidad) mide la proporción de verdaderos positivos sobre el total de positivos reales (verdaderos positivos + falsos negativos).
* Recall para la clase 0: 0.4675 (aproximadamente 47%)
* Recall para la clase 1: 0.3559 (aproximadamente 36%)
* Recall para la clase 2: 0.0469 (aproximadamente 5%)
* La clase 3 no tiene predicciones (0)

F1-Score:

El F1-score es la media armónica de la precisión y el recall, proporcionando una sola métrica que considera ambos.

* F1-score para la clase 0: 0.3810 (aproximadamente 38%)
* F1-score para la clase 1: 0.3281 (aproximadamente 33%)
* F1-score para la clase 2: 0.0769 (aproximadamente 8%)
* La clase 3 no tiene predicciones (0)

Soporte (Support):

El soporte indica el número de ocurrencias reales de cada clase en los datos que se evaluan

* Soporte para la clase 0: 77
* Soporte para la clase 1: 59
* Soporte para la clase 2: 64
* Soporte para la clase 3: 0 (no hay instancias de la clase 3 en los datos)

Lo que podemos determinar con base a estos resultados es lo siguiente:

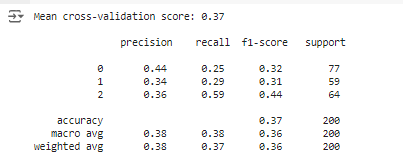
* La clase 0 tiene un desempeño moderado con una precisión de aproximadamente 32% y un recall de aproximadamente 47%. Esto significa que, aunque el modelo predice correctamente una buena cantidad de verdaderos positivos, también tiene una cantidad significativa de falsos positivos.
* La clase 1 tiene un desempeño similar con una precisión de aproximadamente 30% y un recall de aproximadamente 36%. Esto también indica un número considerable de falsos positivos y falsos negativos.
* La clase 2 tiene un desempeño bastante pobre con una precisión de aproximadamente 21% y un recall de aproximadamente 5%. Esto sugiere que el modelo tiene dificultades para predecir correctamente las instancias de esta clase.
* La clase 3 no tiene predicciones ni instancias, lo que podría indicar un problema con la representación de esta clase en el conjunto de datos o un sesgo del modelo que no reconoce esta clase.

Xgboost

Lo primero que podemos visualizar es el entrenamiento del modelo, el cual esta parametrizado con los siguientes parámetros:

* learning\_rate=0.1: Este es el ritmo con el que el modelo aprende. Un valor de 0.1 es bastante estándar y comúnmente usado. Un valor más bajo hace que el modelo aprenda más lentamente, pero puede mejorar la generalización.
* n\_estimators=100: Este parámetro define el número de árboles en el modelo. Con 100 árboles, el modelo está configurado para realizar 100 iteraciones de boosting. Es un número moderado; podrías necesitar ajustar este parámetro dependiendo de los resultados en la validación.
* max\_depth=5: Este parámetro controla la profundidad máxima de los árboles. Un valor de 5 sugiere que los árboles en el modelo no pueden crecer más allá de 5 niveles. Esto puede ayudar a evitar el sobreajuste, pero puede limitar la capacidad del modelo para capturar relaciones complejas en los datos.
* colsample\_bytree=1: Esto indica que se está utilizando el 100% de las características para construir cada árbol. Esta configuración puede hacer que el modelo sea más propenso al sobreajuste si el número de características es muy grande.
* gamma=None: Gamma controla la complejidad de los árboles mediante la regularización. Como no se ha especificado un valor, el modelo utilizará el valor predeterminado, que es 0. Esto significa que no se está aplicando regularización adicional en la división de los nodos.
* min\_child\_weight=None: Este parámetro controla el peso mínimo de la suma de los gradientes de los datos de las hojas. Al no especificar este parámetro, se usará el valor por defecto, que podría ser menos restrictivo para la división de los nodos.
* missing=nan: Indica que se trata con valores faltantes como NaN. Esto es útil para manejar datos incompletos.
* boosting\_type=None: Esto se refiere al tipo de boosting utilizado. Si no se especifica, se utilizará el predeterminado, que es el boosting de árboles de decisión.
* eval\_metric=None: No se ha especificado una métrica de evaluación. Esto significa que se utilizará la métrica predeterminada, que suele ser la exactitud para problemas de clasificación.
* early\_stopping\_rounds=None: No se ha definido un número de rondas para detener el entrenamiento temprano. Esto significa que el modelo se entrenará durante el número total de iteraciones especificado por n\_estimators sin detenerse antes.
* n\_jobs=None: No se ha especificado el número de trabajos en paralelo. Usualmente se usa el valor por defecto, que permite el uso de todos los núcleos disponibles en el CPU para el entrenamiento.

Estos parámetros lo que nos pueden ayudar es a realizar una tarea de clasificación de los siniestro viales presentados.

Al utilizar el entrenamiento y la validación cruzada mediante k-fold, podemos determinar lo siguiente de acuerdo al desempeño del modelo: 

Metricas por clase:

Clase 0:

Precisión: 0.44

Recall: 0.25

F1-score: 0.32

Clase 1:

Precisión: 0.34

Recall: 0.29

F1-score: 0.31

Clase 2:

Precisión: 0.36

Recall: 0.59

F1-score: 0.44

Métricas agregadas:

Exactitud (Accuracy): 0.37

Macro promedio:

Precisión: 0.38

Recall: 0.38

F1-score: 0.36

Promedio ponderado:

Precisión: 0.38

Recall: 0.37

F1-score: 0.36

Desempeño general del modelo:

La exactitud global del modelo es 0.37, lo que indica que el modelo está clasificando correctamente el 37% de las instancias en el conjunto de datos de prueba. Esto es relativamente bajo y sugiere que el modelo podría no estar capturando bien la estructura de los datos o podría necesitar ajustes adicionales dentro del modelo para que pueda realizar una mejor clasificación.

Desempeño por clase:

* Clase 0 tiene una precisión relativamente alta (0.44) pero un recall bajo (0.25). Esto significa que, cuando el modelo predice la clase 0, es correcto en un 44% de los casos, pero solo está capturando el 25% de todas las instancias reales de la clase 0. Esto sugiere que el modelo está perdiendo muchas instancias de la clase 0.
* Clase 1 muestra valores similares en precisión (0.34) y recall (0.29), indicando que el modelo tiene dificultades para identificar y clasificar correctamente las instancias de esta clase.
* Clase 2 tiene un mejor desempeño con un recall (0.59) y un f1-score (0.44) relativamente altos, lo que sugiere que el modelo es más eficaz en la identificación de esta clase en comparación con las otras dos.