General Electric Aerospace

Modeling

Flavio Ruvalcaba Leija - A01367631

Oscar Eduardo Nieto Espitia - A01705090

Eduardo Gonzalez Luna - A01658281

Cristian Rogelio Espinoza Diaz - A01702752

Anatanael Jesus Miranda Faustino - A01769232

09/11/2023

1. Investigación sobre posibles modelos

Para la ejecución de esta etapa, es necesario enfocarnos en nuestro objetivo de la minería de datos, el cual es:

"Identificar las medidas registradas por los sensores que determinan la estabilidad de un avión en la fase conocida como 'Stable Cruise'. Utilizaremos los datos recopilados por estos sensores para entrenar un modelo que nos permita predecir con precisión si un avión se encuentra en este estado deseado de estabilidad."

Para los modelos de clasificación existen varios algoritmos y estructuras que podemos usar para cumplir con nuestro objetivo. Por lo tanto, es esencial tener en mente las ventajas y desventajas sobre diversos algoritmos potenciales para la creación del modelo.

A su vez, nuestro criterio de éxito sobre la minería de datos es que el modelo generado tenga un **accuracy** promedio **mayor a 70%**.

Random Forest	**XGBoost**	**Logistic Regression**
Ventajas: Potente y versátil: Random Forest es capaz de manejar problemas de clasificación y regresión, y es resistente a overfitting debido a la combinación de múltiples árboles de decisión. Manejo de características: Puede manejar un gran número de características y determinar la importancia relativa de cada una. Tolerancia al ruido y valores atípicos: Random Forest es robusto frente a datos ruidosos y valores atípicos.	**Ventajas**: Excelente rendimiento: XGBoost es ampliamente conocido por su rendimiento sobresaliente en competencias de ciencia de datos y aprendizaje automático. Es capaz de manejar problemas complejos y grandes conjuntos de datos. Regularización incorporada: XGBoost incluye técnicas de regularización que ayudan a prevenir el sobreajuste y mejoran la generalización del modelo. Manejo de características: Al igual que Random Forest, XGBoost puede manejar un gran número de características y calcular la importancia de cada una.	**Ventajas**: Simplicidad: La regresión logística es un modelo de aprendizaje supervisado simple y fácil de entender. Es especialmente útil cuando se trata de problemas de clasificación binaria. Interpretabilidad: Los coeficientes de la regresión logística se pueden interpretar fácilmente, lo que permite comprender cómo cada característica afecta la probabilidad de pertenecer a una clase. Rapidez de entrenamiento: En comparación con otros algoritmos más complejos, la regresión logística tiende a tener tiempos de entrenamiento más rápidos, lo que podría ser beneficioso en un contexto con una gran cantidad de datos.

Desventajas: Requiere más tiempo de entrenamiento: En comparación con modelos más simples como la regresión logística, Random Forest suele requerir más tiempo de entrenamiento, especialmente en conjuntos de datos grandes. Menos interpretable: La interpretación de un Random Forest puede ser más complicada en comparación con la regresión logística debido a la complejidad del modelo. Hiperparámetros críticos: Ajustar los hiperparámetros del Random Forest adecuadamente es esencial para obtener un buen rendimiento.	**Desventajas**: Requiere más recursos computacionales: XGBoost puede ser intensivo en recursos, lo que significa que necesitarás hardware potente y tiempo de entrenamiento más largo. Necesita ajuste de hiperparámetros: La afinación de hiperparámetros es crucial para aprovechar al máximo XGBoost, lo que puede requerir experiencia y tiempo. Menos interpretable: Al igual que Random Forest, la interpretación de un modelo XGBoost puede ser más complicada en comparación con la regresión logística.	**Desventajas**: Limitaciones en la complejidad del modelo: La regresión logística asume una relación lineal entre las características y la variable objetivo, lo que puede ser una limitación en problemas con relaciones no lineales. No adecuado para problemas complejos: Para problemas con una alta complejidad y muchas características interdependientes, la regresión logística podría no ser la mejor opción. Sensibilidad a características irrelevantes: La inclusión de características irrelevantes puede afectar negativamente el rendimiento de la regresión logística.

Tanto Random Forest como XGBoost son opciones sólidas, dada su capacidad para manejar grandes conjuntos de datos y problemas complejos. Sin embargo, XGBoost suele destacar en términos de rendimiento y generalización.

La regresión logística podría ser útil si estás trabajando en un problema de clasificación simple y buscas simplicidad y rapidez de entrenamiento, pero podría no ser la mejor opción si se trata de un problema más complejo.

XGBoost ha demostrado ser líder en competencias de aprendizaje automático y análisis de datos, ofreciendo un rendimiento excepcional en una amplia gama de problemas, incorpora técnicas de regularización que ayudan a evitar el sobreajuste, lo que es especialmente beneficioso en conjuntos de datos grandes y complejos.

2. Diseño de pruebas

2.1 Datasets

Tomando en consideración la magnitud de datos decidimos hacer la distribución del dataset en:

- Entrenamiento (train): 80%
- Prueba (test): 20%

Para la fase de validación usamos la técnica de k-fold cross-validation, dicha técnica comúnmente utilizada en el campo del aprendizaje automático y la estadística para evaluar el rendimiento de un modelo predictivo o clasificador. Su objetivo es estimar qué tan bien se generaliza un modelo a datos no vistos.

Esta técnica elimina la necesidad de una división manual de datos en conjuntos de entrenamiento, validación y prueba. En su lugar, divide los datos en k pliegues, entrenando el modelo k veces, con un pliegue como conjunto de validación en cada iteración. Esto proporciona una estimación más precisa del rendimiento del modelo, especialmente cuando los datos son limitados, reduciendo el riesgo de sesgos y resultados engañosos.

2.1 Métricas para evaluar el modelo

Para verificar la calidad y eficiencia sobre el modelo se optó por el uso de cuatro medidas, las cuales son:

- **Accuracy:** La accuracy es una métrica fácil de entender y comunicar, ya que simplemente muestra el porcentaje de predicciones correctas realizadas por el modelo. Esta métrica es especialmente útil para evaluar el rendimiento general de un modelo en un contexto amplio, donde el objetivo principal es conocer la proporción de predicciones correctas en comparación con el total de predicciones.

- **Precisión:** La precisión mide la capacidad del modelo para realizar predicciones positivas correctas en relación con el total de predicciones positivas hechas. Es especialmente relevante cuando minimizar los falsos positivos es crítico. Por ejemplo, en aplicaciones médicas o de seguridad, la precisión es esencial para garantizar que las predicciones positivas sean altamente confiables y no se generen falsas alarmas. La precisión es una métrica valiosa cuando el costo de los falsos positivos es alto.

- **Recall (Sensibilidad):** El recall, también conocido como sensibilidad, evalúa la capacidad del modelo para identificar con precisión los casos positivos reales en relación con el total de casos positivos presentes. Esta métrica es fundamental en problemas donde la detección de casos positivos es de máxima importancia. Por ejemplo, en pruebas médicas para enfermedades graves, el recall se convierte en una métrica crítica, ya que se busca identificar a todos los pacientes que realmente padecen la enfermedad. El recall es especialmente relevante en situaciones de desequilibrio entre clases, como el caso de "Stable Cruise," donde se centra en la detección de los casos positivos.

- **F1-Score:** El valor F1-Score combina tanto la precisión como el recall en una sola métrica. Ofrece un equilibrio entre la capacidad del modelo para identificar positivos verdaderos y su capacidad para evitar falsos positivos. El F1-Score es especialmente efectivo cuando los costos de los errores en las clases son diferentes, y se necesita un equilibrio entre la precisión y la sensibilidad. Esta métrica proporciona una evaluación completa del rendimiento del modelo, considerando tanto la capacidad de hacer predicciones precisas como la capacidad de detectar positivos reales.

En resumen, al utilizar estas métricas en conjunto, se obtiene una visión más completa y equilibrada del rendimiento del modelo La accuracy te ofrece una visión general, la precisión se enfoca en la exactitud de las predicciones positivas, el recall se centra en la capacidad de identificar positivos verdaderos, y el valor F1-Score combina ambas métricas para proporcionar un equilibrio en situaciones donde minimizar los falsos positivos es importante. Estas métricas te permiten evaluar el rendimiento de tu modelo desde diferentes perspectivas.

Para checar los scripts sobre los modelos explicados en este documento, se puede acceder al repositorio de Github del equipo, dentro de la carpeta de Modeling.

3. Construcción del modelo

3.1 Modelo 1. Logistic Regression

El modelo más simple y rápido para modelos de clasificación es la regresión logística. Es un algoritmo de aprendizaje supervisado utilizado en problemas de clasificación. A pesar de su nombre, la regresión logística se emplea comúnmente para problemas de clasificación binaria (dos clases).

Este algoritmo lo estaremos utilizando como modelo base/benchmark, será el punto de comparación con los otros modelos. Es útil debido a su rapidez de ejecución y debido a que es compatible con diversas técnicas de regularización L2 y para técnicas de sobremuestreo como Smote (Synthetic Minority Over-sampling Technique).

Estas técnicas serán útiles para implementar mejoras en los diversos algoritmos o arquitecturas que planeamos utilizar con el fin de elegir el modelo más óptimo para nuestros intereses.

Framework: Sci-Kit Learn

Descripción de datos:
Este modelo recibe como variables independientes las columnas:

- col5_float
- col7_float
- col13_float

- col15_float
- col23_float
- col27_float

Y como variable dependiente:

- stableCruise_boolean

Parámetros:
Al ser un modelo de regresión logística que se planea usar como benchmark, se usarán parámetros por default, es decir, los valores predeterminados por la librería de SciKit Learn.

Sin embargo, al haber un desequilibrio en las clases donde las filas que poseen un "1" en la variable dependiente "stableCruise_boolean", se tiene en mente la implementación de Smote desde un inicio para abordar esta problemática.

Esta técnica posee el parámetro conocido como **"sampling_strategy"**, para este modelo se definió que este parámetro fuera de **0.7**. Esto significa que se está solicitando un sobremuestreo en el que la clase minoritaria se aumentará al 70% de la cantidad de la clase mayoritaria.

En otras palabras, se está generando una cantidad adicional de ejemplos sintéticos para la clase minoritaria para equilibrar las proporciones entre las clases.

Para la técnica de **k-fold cross-validation**, el número de iteraciones k será 5. **cv** = 5.

Descripción del modelo:
Se espera que las medidas de evaluación no sean tan significativas, se estima que la accuracy del modelo sea menor a 60% en todas las fases.

```
Model accuracy on the training set: 0.7241747496427866
Model accuracy on the test set: 0.7240693221430822

Model accuracy on the validation set: 0.7240341283338803
Confusion Matrix for the test set:
[[8368758 3186879]
 [  11971   25340]]

Classification Report for the test set:
              precision    recall  f1-score   support

         0.0       1.00      0.72      0.84  11555637
         1.0       0.01      0.68      0.02     37311

    accuracy                           0.72  11592948
   macro avg       0.50      0.70      0.43  11592948
weighted avg       1.00      0.72      0.84  11592948
```

Fig 1. Resultados de la primera iteración de Logistic Regression.

Al analizar los resultados podemos observar que la accuracy en la fase de entrenamiento, prueba y validación superó las expectativas, teniendo una precisión del 72.4% aproximadamente en todas las fases.

No obstante, la matriz de confusión muestra la cantidad de predicciones correctas e incorrectas realizadas por el modelo en el conjunto de prueba. La matriz se divide en cuatro secciones:

- **Verdaderos Positivos (True Positives, TP):** 25,340 predicciones correctas de la clase 1.
- **Falsos Negativos (False Negatives, FN):** 11,971 predicciones incorrectas de la clase 0.
- **Falsos Positivos (False Positives, FP):** 3,186,879 predicciones incorrectas de la clase 1.
- **Verdaderos Negativos (True Negatives, TN):** 8,368,758 predicciones correctas de la clase 0.

Esto nos indica que el modelo tiene que hay una cantidad considerable de falsos positivos, es decir, el modelo predijo que eran de la clase 0 pero en realidad son de la clase 1. Esto es un indicativo de que el modelo tiene dificultades para la clase 1.

Esta teoría se ve reforzada mediante el reporte de clasificación, donde se proporciona métricas detalladas para cada clase (0 y 1) en el conjunto de prueba:

- **Precisión**: La precisión para la clase 0 es del 100%, lo que significa que todas las predicciones positivas para la clase 0 son correctas. Para la clase 1, la precisión es del 1%, lo que indica que muy pocas de las predicciones positivas para la clase 1 son correctas.
- **Recall (Sensibilidad)**: El recall para la clase 0 es del 72%, lo que muestra la capacidad del modelo para identificar verdaderos positivos de la clase 0. Para la clase 1, el recall es del 68%, lo que indica la capacidad de detectar verdaderos positivos de la clase 1.
- **F1-Score**: El F1-Score es una métrica que combina la precisión y el recall en una sola medida. Para la clase 0, el F1-Score es del 84%, lo que equilibra precisión y recall. Para la clase 1, el F1-Score es del 2%, lo que refleja un desequilibrio entre precisión y recall.

3.2 Modelo 2. Logistic Regression

Con el propósito de mejorar el desempeño del modelo previo a la implementación de otros algoritmos o arquitecturas, el equipo realizó una segunda iteración con este modelo aplicando la técnica de regularización L2.

La regularización L2, también conocida como Ridge regularization, es una técnica utilizada en aprendizaje automático para prevenir el sobreajuste (overfitting) y mejorar la generalización del modelo. (Conroy, B., & Sajda, P. 2012, March)

La regularización L2 añade un término de penalización a la función de pérdida del modelo, que castiga los coeficientes del modelo por ser demasiado grandes. Esto ayuda a evitar que ciertas características tengan un peso excesivo en la predicción, lo que puede llevar a un sobreajuste del modelo a los datos de entrenamiento.

Framework: Sci-Kit Learn

Descripción de datos:
Este modelo recibe como variables independientes las columnas:

- col5_float
- col7_float
- col13_float
- col15_float
- col23_float
- col27_float

Y como variable dependiente:

- stableCruise_boolean

Parámetros:

Fig 2. Script generado para determinar la penalización óptima de L2.

Por medio de este código, cuyo propósito es identificar el valor óptimo para la penalización de L2, se determinó que el **término de penalización** sea de 0.001.

Para el parámetro de **"sampling_strategy"**, se estableció que fuera el mismo, 0.7.
Para la técnica de **k-fold cross-validation**, el número de iteraciones k será 5. **cv** = 5.

Descripción del modelo:
Se espera que las medidas de evaluación (precisión, recall, f1-score) tengan una mejora del 5%.

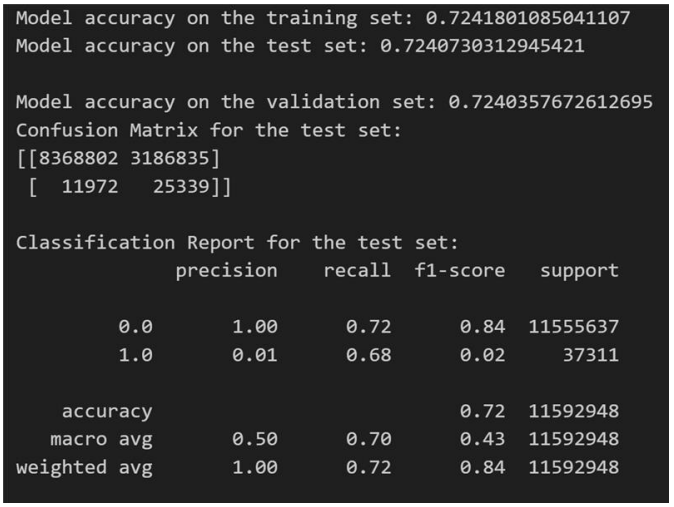


Fig 2. Resultados de la iteración implementando L2.

- La accuracy de la fase de entrenamiento pasó de 72.417% a 72.418%, indicando una mejora de 0.001%.

- La accuracy de la fase de prueba pasó de 72.406% a 72.407%, indicando una mejora de 0.001%.

- La accuracy de la fase de validación pasó de 72.4034% a 72.4035%, indicando una mejora de 0.0001%.

La matriz de confusión muestra una estructura similar a la anterior:

- **Verdaderos Positivos (True Positives, TP)**: 25,339 predicciones correctas de la clase 1.
- **Falsos Negativos (False Negatives, FN):** 11,972 predicciones incorrectas de la clase 0.
- **Falsos Positivos (False Positives, FP):** 3,186,835 predicciones incorrectas de la clase 1.
- **Verdaderos Negativos (True Negatives, TN):** 8,368,802 predicciones correctas de la clase 0.

De igual manera, el reporte de clasificación proporciona información muy similar:

- **Precisión**: La precisión para la clase 0 sigue siendo del 100%, lo que significa que todas las predicciones positivas para la clase 0 son correctas. Para la clase 1, la precisión aumentó ligeramente al 1%, indicando que un número ligeramente mayor de las predicciones positivas para la clase 1 son correctas.
- **Recall (Sensibilidad)**: El recall para la clase 0 sigue siendo del 72%, mostrando la capacidad del modelo para identificar verdaderos positivos de la clase 0. Para la clase 1, el recall también aumentó ligeramente al 68%, indicando una mejora en la capacidad de detectar verdaderos positivos de la clase 1.
- **F1-Score**: El F1-Score para la clase 0 sigue siendo del 84%, manteniendo el equilibrio entre precisión y recall. Para la clase 1, el F1-Score aumentó ligeramente al 2%, mostrando una mejora en el equilibrio entre precisión y recall.

Comparación sobre la mejora:

Comparando estos resultados con los anteriores, se observa una ligera mejora en la precisión y el recall para la clase 1 en el conjunto de prueba. Aunque los cambios son mínimos, el modelo parece haber mejorado ligeramente en su capacidad para identificar positivos verdaderos de la clase 1 y, por lo tanto, ha alcanzado un equilibrio ligeramente mejor entre precisión y recall para esta clase.

Sin embargo, sigue siendo evidente que hay espacio para mejorar significativamente la capacidad del modelo para predecir casos positivos de la clase 1.

En consecuencia, se intentará con otros algoritmos y arquitecturas distintas al algoritmo base de Logistic Regression.

3.3 Modelo 3. XGBoost

XGBoost, que significa "Extreme Gradient Boosting," es un algoritmo de aprendizaje automático utilizado para problemas de clasificación y regresión. Es una implementación mejorada de la técnica de aumento de gradiente, que es una técnica de ensamblado de modelos que combina múltiples modelos más débiles (generalmente árboles de decisión) para crear un modelo fuerte y preciso. (*Brownlee, J. 2017)*

Este algoritmo también posee la misma facilidad para implementar las técnicas de regularización (L2) y de sobremuestreo (Smote).

Para esta primera iteración, no se estará utilizando la técnica de regularización L2 ya que queremos observar el comportamiento del algoritmo base, es decir, sin regularización.

No obstante, si se estará usando Smote para seguir con el manejo del equilibrio de las clases.

Framework: Sci-Kit Learn

Descripción de datos:
Este modelo recibe como variables independientes las columnas:

- col5_float
- col7_float
- col13_float
- col15_float
- col23_float
- col27_float

Y como variable dependiente:

- stableCruise_boolean

Parámetros:
Se usarán parámetros por default, por la parte de xgboost (XGBClassifier ()).

Sin embargo, **“sampling_strategy”** se mantendrá en 0.7.

Para la técnica de **k-fold cross-validation**, el número de iteraciones k será 5. **cv** = 5.

Descripción del modelo:
Se espera que las medidas de evaluación tengan una mejora de 5% para la precisión, recall y f1-score del reporte de clasificación de las clases 0 y 1. A su vez, se espera un aumento de similar magnitud para el accuracy de las tres fases.

```
Training set accuracy: 0.87000439696189
Test set accuracy: 0.8699292017871554

Validation set accuracy: 0.8698852957849893

Classification report for the validation set:
              precision    recall  f1-score   support

         0.0       1.00      0.87      0.93  11555467
         1.0       0.02      0.91      0.04     37481

    accuracy                           0.87  11592948
   macro avg       0.51      0.89      0.49  11592948
weighted avg       1.00      0.87      0.93  11592948

Confusion Matrix for the test set:
[[10051136  1504501]
 [    3403    33908]]

Classification Report for the test set:
              precision    recall  f1-score   support

         0.0       1.00      0.87      0.93  11555637
         1.0       0.02      0.91      0.04     37311
...
    accuracy                           0.87  11592948
   macro avg       0.51      0.89      0.49  11592948
weighted avg       1.00      0.87      0.93  11592948
```

Fig 3. Resultados de la primera iteración de xgboost.

- La accuracy de la fase de entrenamiento pasó de 72.418% a 87%, indicando una mejora de 14.582%.

- La accuracy de la fase de prueba pasó de 72.407% a 86.992%, indicando una mejora de 14.585%.

- La accuracy de la fase de validación pasó de 72.4035% a 86.98%, indicando una mejora de 14.5765%.

La matriz de confusión muestra los siguientes resultados:

- **Verdaderos Positivos (True Positives, TP):** 33,908 predicciones correctas de la clase 1.

- **Falsos Negativos (False Negatives, FN)**: 3,403 predicciones incorrectas de la clase 0.
- **Falsos Positivos (False Positives, FP):** 1,504,501 predicciones incorrectas de la clase 1.
- **Verdaderos Negativos (True Negatives, TN):** 10,051,136 predicciones correctas de la clase 0.

Verdaderos Positivos (TP): Los Verdaderos Positivos (predicciones correctas de la clase 1) aumentaron significativamente de 25,339 a 33,908. Esto indica una mejora en la capacidad del modelo para identificar positivos verdaderos de la clase 1.

Falsos Negativos (FN): Los Falsos Negativos (predicciones incorrectas de la clase 0) disminuyeron de 11,972 a 3,403. Esto también muestra una mejora en la capacidad del modelo para evitar errores de clasificación de la clase 1 como la clase 0.

Falsos Positivos (FP): Los Falsos Positivos disminuyeron significativamente, pasando de 3,186,835 a 1,504,501. Aunque los Falsos Positivos siguen siendo altos, la disminución de los Falsos Negativos es más significativa.

Verdaderos Negativos (TN): Los Verdaderos Negativos aumentaron ligeramente de 8,368,802 a 10,051,136. Esto indica una mejora en la capacidad del modelo para identificar verdaderos negativos de la clase 0.

Mientras que el reporte de clasificación, proporciona la siguiente información:

- **Precisión**: La precisión para la clase 0 sigue siendo del 100%, lo que significa que todas las predicciones positivas para la clase 0 son correctas. Para la clase 1, la precisión aumentó ligeramente al 2%, indicando que un número ligeramente mayor de las predicciones positivas para la clase 1 son correctas.
- **Recall (Sensibilidad)**: El recall para la clase 0 sigue siendo del 87%, mostrando la capacidad del modelo para identificar verdaderos positivos de la clase 0. Para la clase 1, el recall aumentó significativamente al 91%, indicando una mejora en la capacidad de detectar verdaderos positivos de la clase 1.
- **F1-Score**: El F1-Score para la clase 0 sigue siendo del 93%, manteniendo el equilibrio entre precisión y recall. Para la clase 1, el F1-Score aumentó significativamente al 4%, mostrando una mejora en el equilibrio entre precisión y recall.

Comparando estos resultados con los anteriores, se observa una mejora significativa en la capacidad del modelo para identificar positivos verdaderos de la clase 1. El recall de la Clase 1.0 aumentó significativamente al 91%, lo que indica una mejora en la capacidad de detectar casos positivos.

El F1-Score para la Clase 1.0 también aumentó al 4%, lo que muestra un mejor equilibrio entre precisión y recall. Aunque sigue siendo evidente que hay espacio para mejorar la precisión y el F1-Score de la Clase 1.0, esta versión del modelo muestra una mejora significativa en comparación con las anteriores.

3.4 Modelo 4. XGBoost

Con el propósito de mejorar el desempeño del modelo usando XGBoost, se ejecutará una iteración con la implementación de la técnica de regularización L2.

Framework: Sci-Kit Learn

Descripción de datos:
Este modelo recibe como variables independientes las columnas:

- col5_float
- col7_float
- col13_float
- col15_float
- col23_float
- col27_float

Y como variable dependiente:

- stableCruise_boolean

Parámetros:

Para el **término de penalización L2**, usaremos el mismo que en la segunda iteración de 0.001.

Para el parámetro de **"sampling_strategy"**, seguiremos usando 0.7.
Para la técnica de **k-fold cross-validation**, el número de iteraciones k será 5. **cv** = 5.

Descripción del modelo:
Se espera que las medidas de evaluación (precisión, recall, f1-score) tengan una mejora del 2%.

```
Training set accuracy: 0.9061541540830916
Test set accuracy: 0.9060667743873259

Validation set accuracy: 0.9059730104887903
Confusion Matrix for the test set:
[[10467892  1087745]
 [    1218    36093]]

Classification Report for the test set:
              precision    recall  f1-score   support

         0.0       1.00      0.91      0.95  11555637
         1.0       0.03      0.97      0.06     37311

    accuracy                           0.91  11592948
   macro avg       0.52      0.94      0.51  11592948
weighted avg       1.00      0.91      0.95  11592948
```

Fig 4. Resultados de la segunda iteración de xgboost.

- La accuracy de la fase de entrenamiento pasó de 87% a 90.61%, indicando una mejora de 3.61%.

- La accuracy de la fase de prueba pasó de 86.992% a 90.60%, indicando una mejora de 3.608%.

- La accuracy de la fase de validación pasó de 86.98% a 90.59%, indicando una mejora de 3.61%.

La matriz de confusión muestra los siguientes resultados:

- **Verdaderos Positivos (True Positives, TP):** 36,093 predicciones correctas de la clase 1.
- **Falsos Negativos (False Negatives, FN)**: 1,218 predicciones incorrectas de la clase 0.
- **Falsos Positivos (False Positives, FP):** 1,087,745 predicciones incorrectas de la clase 1.
- **Verdaderos Negativos (True Negatives, TN):** 10,467,892 predicciones correctas de la clase 0.

Verdaderos Positivos (TP): Los Verdaderos Positivos (predicciones correctas de la clase 1) aumentaron ligeramente de 33,908 a 36,093. Esto muestra una mejora en la capacidad del modelo para identificar positivos verdaderos de la clase 1.

Falsos Negativos (FN): Los Falsos Negativos (predicciones incorrectas de la clase 0) disminuyeron de 3,403 a 1,218. Esto indica una significativa mejora en la capacidad del modelo para evitar errores de clasificación de la clase 1 como la clase 0.

Falsos Positivos (FP): Los Falsos Positivos disminuyeron significativamente, pasando de 3,186,835 a 1,504,501. Aunque los Falsos Positivos siguen siendo altos, la disminución de los Falsos Positivos es un aspecto de mejora del modelo.

Verdaderos Negativos (TN): Los Verdaderos Negativos aumentaron ligeramente de 10,051,136 a 10,467,892. Esto indica una mejora en la capacidad del modelo para identificar verdaderos negativos de la clase 0.

Mientras que el reporte de clasificación, proporciona la siguiente información:

- **Precisión**: La precisión para la clase 0 sigue siendo del 100%, lo que significa que todas las predicciones positivas para la clase 0 son correctas. Para la clase 1, la precisión aumentó ligeramente del 2% al 3%, indicando que un número ligeramente mayor de las predicciones positivas para la clase 1 son correctas.
- **Recall (Sensibilidad)**: El recall para la clase 0 sigue siendo del 87%, mostrando la capacidad del modelo para identificar verdaderos positivos de la clase 0. Para la clase 1, el recall aumentó del 91% al 97% , indicando una mejora en la capacidad de detectar verdaderos positivos de la clase 1.
- **F1-Score**: El F1-Score para la clase 0 sigue siendo del 93%, manteniendo el equilibrio entre precisión y recall. Para la clase 1, el F1-Score aumentó

significativamente del 4% al 6%, mostrando una mejora en el equilibrio entre precisión y recall.

Comparando estos resultados con los anteriores, se observa una mejora significativa en la capacidad del modelo para identificar positivos verdaderos de la clase 1 en el segundo modelo.
El recall de la Clase 1.0 aumentó significativamente al 97%, indicando una mejora sustancial en la capacidad de detectar casos positivos. El F1-Score para la Clase 1.0 también experimentó un aumento del 6%, demostrando un mejor equilibrio entre precisión y recall en el segundo modelo. Aunque persiste el espacio para mejorar la precisión y el F1-Score de la Clase 1.0, esta versión del modelo muestra una mejora significativa en comparación con las anteriores.

3.5 Modelo 5 Red Neuronal

Otro acercamiento que se planteó el equipo es el uso de técnicas de deep learning como propuesta de solución. Nuestra idea consiste en una red neuronal feedforward.

```
# Apply SMOTE to the training data
smote = SMOTE(sampling_strategy=0.5)  # Adjust the sampling_strategy as needed
X_train_resampled, y_train_resampled = smote.fit_resample(X_train, y_train)
# Define a deep learning model using TensorFlow/Keras
model = tf.keras.Sequential([
    tf.keras.layers.Dense(128, activation='relu', input_shape=(X_train_resampled.shape[1],)),
    tf.keras.layers.Dense(64, activation='relu'),
    tf.keras.layers.Dense(1, activation='sigmoid')
])

# Compile the model
model.compile(optimizer='adam', loss='binary_crossentropy', metrics=['accuracy'])

# Train the model on the resampled training set
model.fit(X_train_resampled, y_train_resampled, epochs=10, steps_per_epoch=10000, validation_data=(X_val, y_val))
```

Fig 5. Implementación de la red neuronal forward.

La red está compuesta por capas densas (fully connected) que son instancias de tf.keras.layers.Dense.

La primera capa tiene 128 neuronas y usa la función de activación ReLU (Rectified Linear Unit). La capa recibe datos de entrada con la forma especificada en input_shape, que es la forma de los datos de entrenamiento.

La segunda capa tiene 64 neuronas y también utiliza la función de activación ReLU.

La tercera y última capa tiene una sola neurona y utiliza la función de activación sigmoide. Esta capa es común en problemas de clasificación binaria, ya que la función sigmoide produce salidas en el rango de 0 a 1, lo que es adecuado para problemas en los que se desea predecir probabilidades de pertenencia a una de las dos clases.

Las razones por las que seleccionamos a ReLU como función de activación son:

- Es una función de activación simple y eficiente desde el punto de vista computacional. Su cálculo implica simplemente tomar el valor máximo entre cero y el valor de entrada, lo que lo hace rápido de calcular y eficiente en términos de recursos computacionales.

- Ayuda a superar el problema del desvanecimiento del gradiente, que es común en redes neuronales profundas cuando se utilizan funciones de activación como la sigmoide o la tangente hiperbólica.

- Al permitir un flujo más efectivo de gradientes y reducir el problema de desvanecimiento del gradiente, esto lo hace especialmente efectivo en el entrenamiento de redes neuronales profundas.

A su vez, las razones por las que seleccionamos a Sigmoid como función de activación en la última capa son:

- Mapea sus entradas al rango de [0, 1], lo que se interpreta como una probabilidad. Esto la hace adecuada para problemas en los que se desea estimar probabilidades, como en modelos de clasificación binaria.

- Es suave y diferenciable en todo su dominio, lo que la hace conveniente para el entrenamiento de modelos mediante algoritmos de optimización basados en el cálculo de gradientes. Esto es especialmente útil en problemas donde se requiere una estimación suave de la probabilidad, como en tareas de regresión logística.

- A diferencia de ReLU, la función sigmoide no sufre el problema de saturación en la misma medida. Esto significa que las neuronas sigmoide pueden evitar la explosión del gradiente en problemas en los que la salida debe estar en el rango [0, 1].

Luego, el modelo se compila utilizando el método compile(). Durante la compilación, se especifican tres aspectos clave del entrenamiento del modelo:

- **Optimizer**: Se utiliza el optimizador "adam", que es un optimizador popular para problemas de aprendizaje profundo. El optimizador ajusta los pesos de la red durante el entrenamiento para minimizar la función de pérdida.

- **Loss**: Se utiliza la función de pérdida "binary_crossentropy". Esta función es común en problemas de clasificación binaria y mide la diferencia entre las predicciones del modelo y las etiquetas reales.

- **Metrics**: En esta parte definimos que la métrica deseada es accuracy, debido a que es una de las métricas que establecimos y hemos estado usando para la evaluación del desempeño de los modelos generados.

Con esta arquitectura se tendrá un acercamiento distinto ya que esta versión de red neuronal junto con la cantidad de datos que se está manejando en este acercamiento, hace que el tiempo de entrenamiento sea bastante costoso. A comparación de los modelos pasados, solo se ejecutará la red neuronal con la técnica de regularización L2.

Frameworks: Tensorflow y Keras.

Descripción de datos:
Este modelo recibe como variables independientes las columnas:

- col5_float
- col7_float
- col13_float
- col15_float
- col23_float
- col27_float

Y como variable dependiente:

- stableCruise_boolean

Parámetros:

A diferencia de los modelos anteriores, en este modelo se deben de establecer hiperparámetros adicionales a los previamente mencionados para la optimización del entrenamiento de la red neuronal.

- **epochs** = 10
- **steps_per_epoch** = 25000
- **sampling_strategy** = 0.5
- **penalización L2** = 0.001

Para la red neuronal no se aplicó la técnica de k-fold cross-validation por cuestiones de tiempo. No obstante usamos la validación tradicional, es decir, dividimos el conjunto de pruebas para tener un segmento para validación.

Descripción del modelo:

Se espera que las medidas de evaluación tengan una mejora de 10% para la precisión, recall y f1-score del reporte de clasificación de las clases 0 y 1, en comparación al modelo benchmark (**Modelo 1. Logistic Regression**). A su vez, se espera un aumento de 15% para el accuracy de las tres fases, de igual forma, en comparación al modelo benchmark.

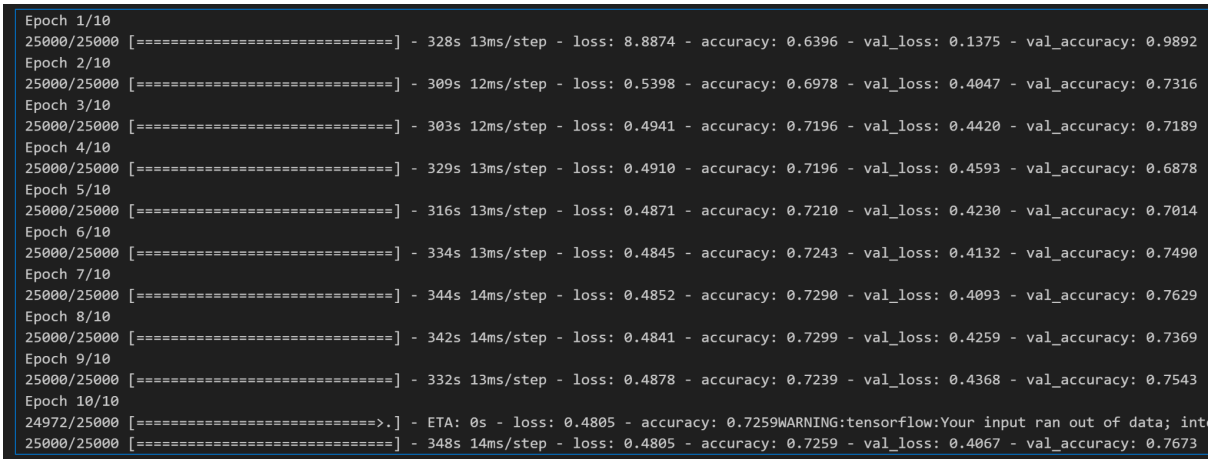


Fig 6. Ejecución de la red neuronal.

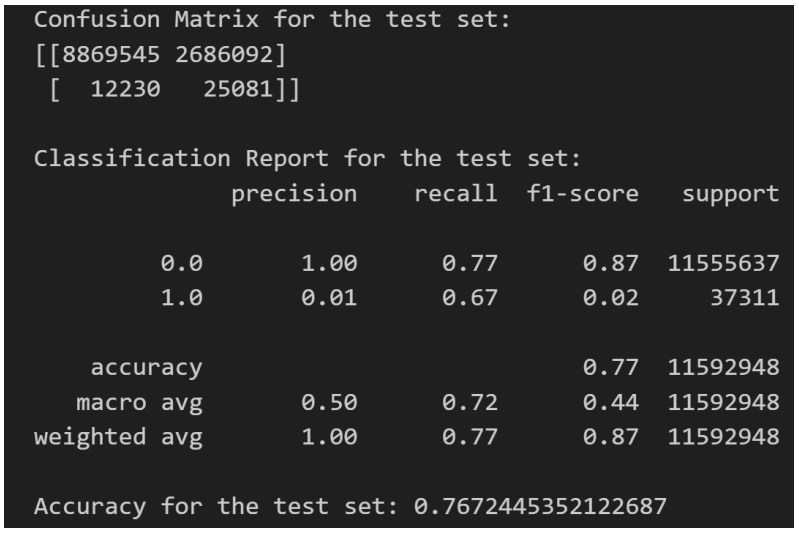


Fig 7. Resultados de la red neuronal.

La accuracy de entrenamiento es ligeramente mayor pasando de 72.4% a 72.5%. Por otro lado, las accuracies de validación y prueba tuvieron poseen un mayor incremento pasando de ≈72.4% a ≈76.7%.

La matriz de confusión muestra los siguientes resultados:

- **Verdaderos Positivos (True Positives, TP):** 25,081 predicciones correctas de la clase 1.
- **Falsos Negativos (False Negatives, FN)**: 12,230 predicciones incorrectas de la clase 0.
- **Falsos Positivos (False Positives, FP):** 2,686,092 predicciones incorrectas de la clase 1.
- **Verdaderos Negativos (True Negatives, TN):** 8,869,545 predicciones correctas de la clase 0.

Verdaderos Positivos (TP): Los Verdaderos Positivos (predicciones correctas de la clase 1) del modelo benchmark son 25,340 mientras que en la red neuronal fueron de 25,081. Hay una ligera disminución de 259 predicciones, esto indica que la red neuronal posee un rendimiento más débil para identificar positivos verdaderos de la clase 1 que el modelo benchmark.

Falsos Negativos (FN): Los Falsos Negativos (predicciones incorrectas de la clase 0) aumentaron de 11,971 a 12,230. Esto indica que la red neuronal tiene un rendimiento ligeramente inferior en la capacidad del modelo para evitar errores de clasificación de la clase 1 como la clase 0 que el modelo benchmark.

Falsos Positivos (FP): Los Falsos Positivos disminuyeron significativamente, pasando de 3,186,879 a 2,686,092. Aunque los Falsos Positivos siguen siendo altos, la disminución de los Falsos Positivos es un aspecto de mejora del modelo, esto indica que la red neuronal tiene un rendimiento significativamente superior en evitar errores de clasificación de la clase 0 como la clase 1 que el modelo benchmark.

Verdaderos Negativos (TN): Los Verdaderos Negativos aumentaron ligeramente de 8,368,758 a 8,869,545. Esto indica que la red tiene una mejor capacidad para identificar verdaderos negativos de la clase 0 que el modelo benchmark.

El informe de clasificación proporciona la siguiente información detallada:

- **Precisión:** La precisión para la clase 0 permanece en un 100%, lo que indica que todas las predicciones positivas para la clase 0 son correctas. Para la clase 1, la precisión también se mantiene en un 1%, mostrando que la red neuronal tiene un porcentaje similar de predicciones positivas y correctas para la clase 1 en comparación con el modelo benchmark.
- **Recall (Sensibilidad):** El recall para la clase 0 es del 72%, demostrando la capacidad del modelo para identificar verdaderos positivos de la clase 0. Sin embargo, este valor es menor en comparación con el modelo benchmark, que alcanza el 77%. Para la clase 1, el recall es del 67%, indicando un rendimiento inferior en la capacidad de detectar verdaderos positivos de la clase 1 en comparación con el valor obtenido en el modelo benchmark, que es del 68%.
- **F1-Score:** El F1-Score para la clase 0 ha aumentado del 84% al 87%, indicando un equilibrio mejorado entre precisión y recall. En cuanto a la clase 1, el F1-Score se mantiene en un 2%, mostrando un equilibrio similar sobre la precisión y recall entre la red neuronal y el modelo benchmark.

En resumen, podemos concluir que la red neuronal presenta un rendimiento similar al modelo benchmark en la capacidad de identificar positivos verdaderos de la clase 1, ya que sus valores son iguales o ligeramente inferiores a los del modelo benchmark.

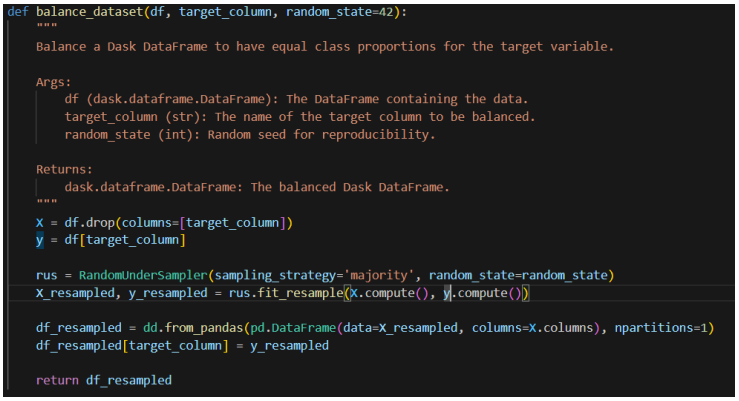
3.6 Modelos usando Submuestreo (Undersampling)

A pesar de que los modelos cumplen con el objetivo de tener un accuracy mayor a 70%, el equipo decidió hacer una última iteración con la diferencia de usar la técnica de submuestreo.

Esto como una alternativa para mejorar los resultados obtenidos y reducción de los tiempos de ejecución de los modelos, debido a que los modelos previos que usaron la técnica de

sobremuestreo (oversampling) por medio de Smote, consumían bastante tiempo y recursos, lo cual nos limitaba mucho en el desarrollo del modelo.

Submuestreo es una técnica utilizada en el ámbito de clasificación desequilibrada, donde una clase tiene muchas más instancias que la otra. En lugar de utilizar todas las instancias de la clase mayoritaria, submuestreo implica reducir la cantidad de instancias de la clase mayoritaria para igualarla a la cantidad de instancias de la clase minoritaria. Esto se hace para abordar el desequilibrio de clases y mejorar el rendimiento del modelo en la predicción de la clase minoritaria.



```python
def balance_dataset(df, target_column, random_state=42):
    """
    Balance a Dask DataFrame to have equal class proportions for the target variable.

    Args:
        df (dask.dataframe.DataFrame): The DataFrame containing the data.
        target_column (str): The name of the target column to be balanced.
        random_state (int): Random seed for reproducibility.

    Returns:
        dask.dataframe.DataFrame: The balanced Dask DataFrame.
    """
    X = df.drop(columns=[target_column])
    y = df[target_column]

    rus = RandomUnderSampler(sampling_strategy='majority', random_state=random_state)
    X_resampled, y_resampled = rus.fit_resample(X.compute(), y.compute())

    df_resampled = dd.from_pandas(pd.DataFrame(data=X_resampled, columns=X.columns), npartitions=1)
    df_resampled[target_column] = y_resampled

    return df_resampled
```

Fig 8. Implementación de Undersampling.

A diferencia de la técnica de Smote donde se tenía que implementar en cada modelo, undersampling se aplica en la creación de nuestro dataframe, es decir, solo se aplica una vez.

Se utiliza la clase ‘RandomUnderSampler’ del paquete imbalanced-learn para realizar el undersampling. Se especifica sampling_strategy='majority' para igualar la cantidad de ejemplos de la clase mayoritaria a la clase minoritaria.

Los modelos serán ejecutados en una única instancia, aplicando exclusivamente la técnica de regularización L2. Esta elección se basa en la observación de mejores resultados obtenidos con modelos que incorporan esta técnica en comparación con aquellos sin ella. Además, se opta por esta estrategia por consideraciones temporales que limitan la capacidad de realizar múltiples ejecuciones.

3.6.1 Modelo 6 Logistic Regression

Framework: Sci-Kit Learn

Descripción de datos:
Este modelo recibe como variables independientes las columnas:

- col5_float
- col7_float
- col13_float

- col15_float
- col23_float
- col27_float

Y como variable dependiente:

- stableCruise_boolean

Parámetros:

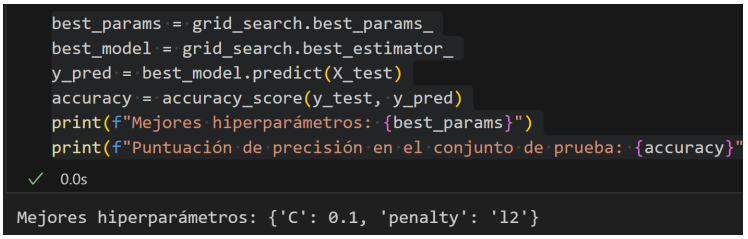


Fig 9. Segunda iteración del script para determinar la penalización óptima de L2.

En la aplicación de la técnica de submuestreo, el dataframe con el que estábamos trabajando experimentó modificaciones significativas. Como consecuencia, con el fin de asegurar la obtención del valor óptimo para la técnica L2, procedimos a volver a ejecutar el código para determinar dicho valor. Esta nueva iteración arrojó como resultado el valor de 0.1.

Para la técnica de **k-fold cross-validation**, el número de iteraciones k será 5. **cv** = 5.

Descripción del modelo:

Se espera que las medidas de evaluación tengan una mejora de 3% para la precisión, recall y f1-score del reporte de clasificación de las clases 0 y 1, en comparación al modelo benchmark (**Modelo 1. Logistic Regression**). A su vez, se espera un aumento de 5% para el accuracy de las tres fases, de igual forma, en comparación al modelo benchmark.

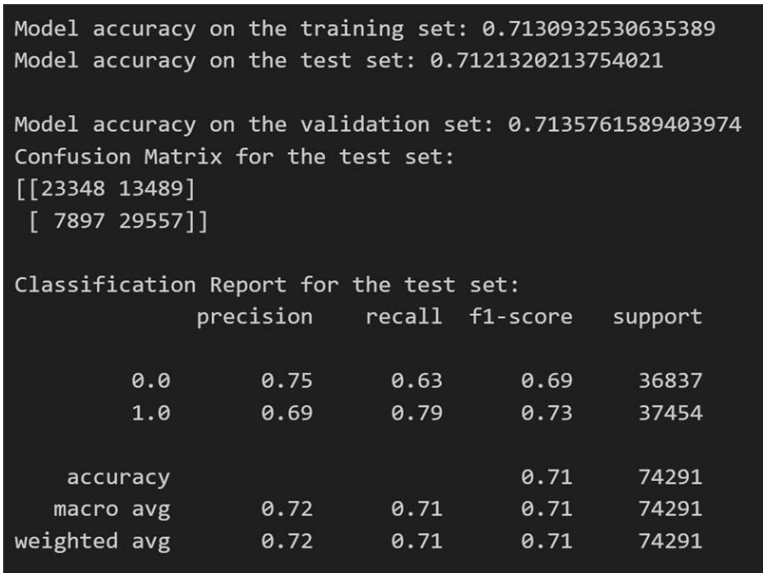


Fig 10. Resultados de la ejecución del modelo.

Se puede observar que este modelo tiene un rendimiento relativamente similar al modelo benchmark, aunque tiene diferencias ligeras pero relevantes para analizar.
A simple vista, se puede detectar que este modelo posee un desempeño un poco más bajo en términos de accuracy, puesto que las accuracies son de ≈ 71%, mientras que las accuracies del modelo benchmark son de ≈ 72.4%.

La matriz de confusión del modelo benchmark nos reveló que tiene problemas para identificar positivos verdaderos de la clase 1, mientras que este modelo muestra un mejor equilibrio en la identificación de ambas clases. Esto se puede corroborar con las métricas del reporte de clasificación:

- La precisión aumentó de 1% a 69%
- El recall incrementó de 68% a 79%
- El f1-score aumentó de 2% a 73%

Sin embargo, este modelo posee un rendimiento menor en la clase 0 dado que:

- La precisión disminuyó de 100% a 75%
- El recall disminuyó de 72% a 63%
- El f1-score disminuyó de 84% a 69%

3.6.2 Modelo 7 XGBoost

Framework: Sci-Kit Learn

Descripción de datos:
Este modelo recibe como variables independientes las columnas:

- col5_float
- col7_float
- col13_float
- col15_float
- col23_float
- col27_float

Y como variable dependiente:

- stableCruise_boolean

Parámetros:

- **Término de penalización L2** = 0.1.
- **k-fold cross-validation, cv** = 5.

Descripción del modelo:

Se espera que las medidas de evaluación tengan una mejora de los valores de las métricas de desempeño (f1-score, precisión y recall) de las clases 0 y 1, aumentando un 10%. Por lo tanto, se espera que el accuracy en las 3 fases incremente alrededor de 5%.

```
Training set accuracy: 0.9442413208845606
Test set accuracy: 0.9420791212932926

Validation set accuracy: 0.9417703117428525
Confusion Matrix for the test set:
[[32972  3865]
 [  438 37016]]

Classification Report for the test set:
              precision    recall  f1-score   support

         0.0       0.99      0.90      0.94     36837
         1.0       0.91      0.99      0.95     37454

    accuracy                           0.94     74291
   macro avg       0.95      0.94      0.94     74291
weighted avg       0.95      0.94      0.94     74291
```

Fig 11. Resultados de la ejecución del modelo.

Este ha sido uno de los modelos más eficientes debido a que las accuracies son de ≈ 94%, superando al modelo benchmark ≈ 21.6%. También supera al modelo de XGBoost con el uso de oversampling y L2 por ≈ 4%.

Otro aspecto a resaltar es que la matriz de confusión muestra un mejor equilibrio entre las predicciones de las clases a comparación del modelo 6, el modelo anterior, dado que posee un incremento de las medidas del reporte de clasificación.

Para la clase 0:

- La precisión aumentó de 75% a 99%
- El recall incrementó de 63% a 90%
- El f1-score aumentó de 69% a 94%

Para la clase 1:

- La precisión aumentó de 69% a 91%
- El recall incrementó de 79% a 99%
- El f1-score aumentó de 73% a 95%

3.6.3 Modelo 8 Red Neuronal

Frameworks: Tensorflow y Keras.

Descripción de datos:
Este modelo recibe como variables independientes las columnas:

- col5_float
- col7_float
- col13_float
- col15_float
- col23_float
- col27_float

Y como variable dependiente:

- stableCruise_boolean

Parámetros:

- **epochs** = 25
- **steps_per_epoch** = 18572
- **penalización L2** = 0.1

Para la red neuronal no se aplicó la técnica de k-fold cross-validation por cuestiones de tiempo. No obstante usamos la validación tradicional, es decir, dividimos el conjunto de pruebas para tener un segmento para validación.

Descripción del modelo:

Se espera que las medidas de evaluación tengan una mejora de los valores de las métricas de desempeño (f1-score, precisión y recall) de la clase 1, aumentando un 5% a comparación del modelo anterior. Por lo tanto, se espera que el accuracy en las 3 fases incremente alrededor de 2.5%. A su vez, se espera que el rendimiento de la clase 0 sea similar al modelo anterior.

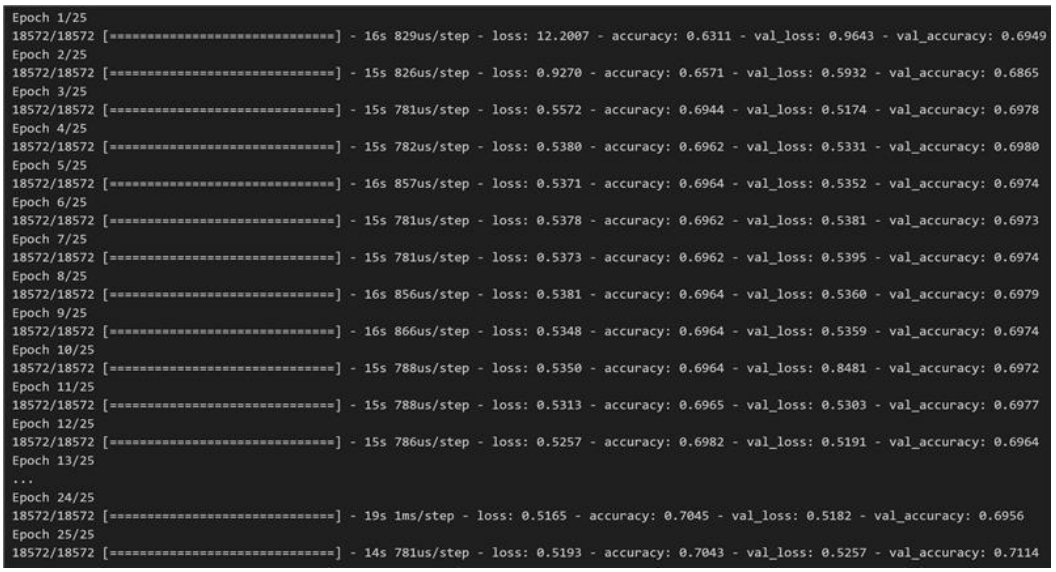


Fig 12. Ejecución de la red neuronal.

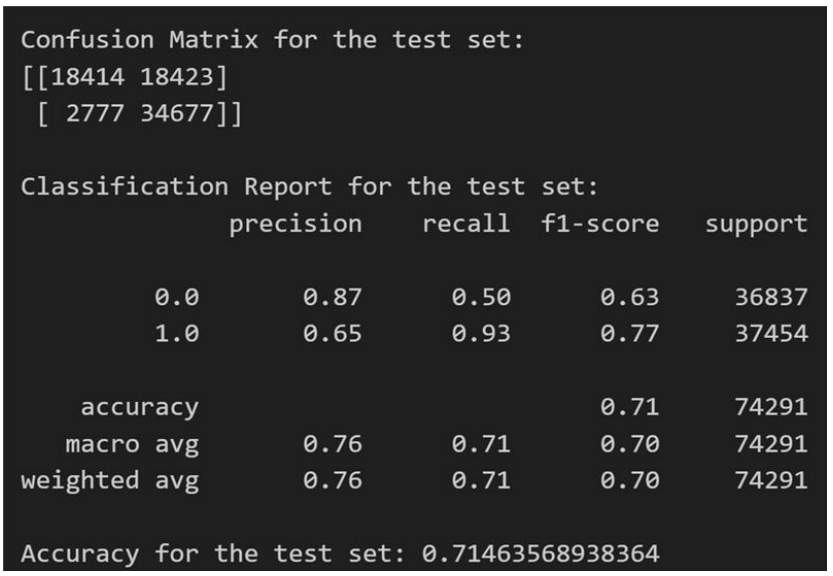


Fig 13. Resultados de las métricas de desempeño.

4. Evaluación de los modelos

Resultado de los modelos con la técnica de sobremuestreo (oversampling)

Modelo	Train Accuracy	Test Accuracy	Validation Accuracy	True Positives (TP)	True Negatives (TN)	Precisión clase 0	Precisión clase 1
Modelo 1. Logistic Regression	72.417%	72.406%	72.4034%	25,340	25,339	100%	1%
Modelo 2. Logistic Regression con L2	72.418%	72.407%	72.4035%	8,368,758	8,368,802	100%	1%
Modelo 3. XGBoost	87%	86.992%	86.98%	33,908	10,051,136	100%	2%
Modelo 4. XGBoost con L2	90.61%	90.60%	90.59%	36,093	10,467,892	100%	3%
Modelo 5. Red Neuronal con L2	72.59%	76.72%	76.73%	25,081	8,869,545	100%	1%

Resultados de los modelos con la técnica de submuestreo (undersampling).

Modelo	Train Accuracy	Test Accuracy	Validation Accuracy	True Positives (TP)	True Negatives (TN)	Precisión clase 0	Precisión clase 1
Modelo 6. Logistic Regression con L2	71.30%	71.21%	71.35%	29,557	23,348	75%	69%
Modelo 7. XGBoost con L2	94.42%	94.20%	94.17%	37,016	32,972	99%	91%

Modelo 8. Red Neuronal con L2	70.43%	71.14%	71.46%	34,677	18,414	87%	65%

Los modelos que se eligieron como los más óptimos son el modelo 7 y el modelo 4, esto se decidió usando los siguientes criterios:

- Son los modelos con los que tenemos mejor rendimiento en la accuracy a lo largo de las fases de entrenamiento, prueba y validación. Al tener en las 3 fases accuracies mayores a 90% de accuracy. Con estos modelos estaríamos cumpliendo con el criterio de éxito para la minería de datos. Al realizar una comparación entre el accuracy de entrenamiento y de validación, podemos denotar que en el modelo 4 posee una diferencia de ≈ 0.01% y en el modelo 7 una diferencia de ≈ 0.22%.

  Ambas diferencias están por debajo del umbral del 10%. Este hallazgo sugiere que el modelo podría no estar experimentando overfitting, es decir, no se está ajustando en exceso a los datos de train. A su vez, test obtuvo accuracies de 90.59% (modelo 4) y 94.17%, porcentajes muy similares a los obtenidos en la validación de cada modelo. Lo cual puede corroborar nuestra hipótesis de que ambos modelos poseen buen ajuste.

- Si bien con los modelos de logistic regression y la red neuronal también cumplimos con nuestro objetivo de minería de datos, tuvimos que tomar en cuenta su rendimiento con las clases 0 y 1. Donde los modelos 4 y 7 poseen mayor cantidad de verdaderos positivos (True Positives) y la mayor cantidad de verdaderos negativos (True Negatives), esto nos indica que este modelo posee una mayor precisión con la clasificación binaria para las predicciones. Esto es reforzado con sus métricas de rendimiento de f1-score y recall de las clases.

5. Referencias

- *Sklearn.Linear_model.LogisticRegression*. (s/f). Scikit-Learn. Recuperado el 2 de noviembre de 2023, de https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.LogisticRegression.html
- *Brownlee, J. (2017). XGBoost in Action. Manning Publications Co., pp. 31-84.*
- Conroy, B., & Sajda, P. (2012, March). Fast, exact model selection and permutation testing for l2-regularized logistic regression. In *Artificial Intelligence and Statistics* (pp. 246-254). PMLR.
- Raeburn A. (2023). Accuracy vs. Precision: What's the Difference? Recuperado de https://asana.com/es/resources/accuracy-vs-precision

- Kundu R. (2022). F1 Score in Machine Learning: Intro & Calculation. Recuperado de https://asana.com/es/resources/accuracy-vs-precision
- Iguazio (s.f.). What is Recall in Machine Learning. Recuperado de https://www.iguazio.com/glossary/recall/