General Electric Aerospace

Reporte Final

Flavio Ruvalcaba Leija - A01367631

Oscar Eduardo Nieto Espitia - A01705090

Eduardo Gonzalez Luna - A01658281

Cristian Rogelio Espinoza Diaz - A01702752

Anatanael Jesus Miranda Faustino - A01769232

29/11/2023

1. Introducción

General Electric Aviation es una división de la empresa multinacional General Electric (GE) que se dedica a la fabricación y desarrollo de productos y servicios relacionados con la industria aeroespacial. Fundada en 1917, General Electric Aviation es uno de los principales actores en el campo de la aviación y ha desempeñado un papel fundamental en la evolución de la tecnología de la aviación.

La empresa se especializa en la producción de motores de avión, sistemas de propulsión, componentes y servicios relacionados con la aviación comercial y militar. General Electric Aviation también se dedica a la investigación y desarrollo de tecnologías avanzadas, como motores más eficientes y amigables con el medio ambiente, sistemas de control de vuelo y componentes de aeronaves innovadores.

La compañía tiene una presencia global y trabaja en colaboración con numerosas aerolíneas, fabricantes de aviones y organizaciones gubernamentales para mejorar la eficiencia y la seguridad en la aviación. General Electric Aviation es conocida por su compromiso con la innovación y la mejora continua en la industria de la aviación, lo que la convierte en una empresa con alta relevancia en el mundo de la tecnología aeroespacial.

Actualmente General Electric Aerospace está teniendo problemas para optimizar el algoritmo que usan para tener estabilidad en el punto de misión de crucero. Tener estabilidad en el punto de misión de crucero puede proporcionar varios beneficios: seguridad en el vuelo, eficiencia de combustible, durabilidad de la aeronave. General Electric quiere generar conocimiento para identificar las variables más significativas que ayuden a mejorar la estabilidad de vuelo.

2. Estado del arte

El equipo realizó una investigación sobre algoritmos que podríamos implementar para el desarrollo del modelo. En esta investigación buscamos información sobre algoritmos que podríamos implementar para el reto.

Posteriormente, realizamos iteraciones con logistic regression, XGBoost y una red neuronal. El algoritmo que siempre destacó por tener mejor rendimiento en todas las iteraciones fue XGBoost.

A su vez, encontramos una estrategia para la búsqueda del parámetro óptimo para la técnica de regularización L2, esto mediante el uso del algoritmo Grid Search CV.

Por último, establecimos como métricas de evaluación a accuracy, precision, recall y f1-score. Asimismo utilizamos la técnica de k-fold cross-validation como herramienta de validación para los modelos con Logistic Regression y XGBoost. De igual forma, el socio formador nos compartió una carpeta de box con un set de 100 archivos que el socio formador nos brindó para reforzar la validación de nuestros modelos.

A continuación, se brindará información más detallada sobre cada uno de estos aspectos mencionados.

3. Selección de un modelo óptimo

Para el modelo final utilizamos el algoritmo XGBoost (Extreme Gradient Boosting) es un método de aprendizaje automático supervisado utilizado para clasificación y regresión. Es una implementación de código abierto popular y eficiente del algoritmo de árboles aumentados de gradientes. XGBoost utiliza la potenciación de gradientes, que es un algoritmo de aprendizaje que combina múltiples modelos débiles (árboles de decisión) para crear un modelo más fuerte y preciso.

Este algoritmo se destaca por su capacidad para manejar grandes conjuntos de datos, su eficiencia en términos de tiempo de entrenamiento y predicción, y su capacidad para manejar características numéricas y categóricas.

4. Refinamiento del modelo

Un aspecto relevante en este proyecto fue la implementación de una técnica de regularización conocida como 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.

Para establecer el valor óptimo para dicho término de penalización utilizamos el algoritmo conocido como Grid Search CV. Para determinar dicho parámetro, el algoritmo realiza una búsqueda de fuerza bruta probando todas los valores que colocamos en un array y comparaba el accuracy del modelo que obtenía con dicho valor para que al final imprimiera el valor de penalización que obtuvo la accuracy más alta.

5. Métricas de evaluación

Para evaluar la calidad y eficiencia del modelo, se utilizaron cuatro métricas clave:

- **Accuracy:** Indica el porcentaje total de predicciones correctas. Proporciona una visión general del rendimiento en contextos amplios donde interesa la proporción de aciertos.

- **Precisión:** Mide la capacidad del modelo para hacer predicciones positivas correctas sobre el total de predicciones positivas. Es crucial cuando se deben minimizar los falsos positivos, como en aplicaciones médicas o de seguridad.

- **Recall (Sensibilidad):** Evalúa la habilidad de detectar casos positivos reales sobre el total de casos positivos disponibles. Es fundamental cuando la detección de positivos es crítica, como en pruebas para enfermedades graves.

- **F1-Score:** Combina la precisión y la sensibilidad en una métrica, ofreciendo un equilibrio entre identificar verdaderos positivos y evitar falsos positivos. Es útil cuando los costos de errores en clases son distintos.

  En conjunto, estas métricas entregan una evaluación integral del desempeño del modelo desde múltiples perspectivas: la accuracy entrega una visión global, la precisión se enfoca en la confiabilidad de los positivos, el recall en la detección de positivos verdaderos, y el valor f1-score combina ambas métricas para proporcionar un equilibrio en situaciones donde minimizar los falsos positivos es importante.

6. Validación del modelo

La validación cruzada k-fold es una técnica comúnmente utilizada en el aprendizaje automático para evaluar el rendimiento de un modelo y reducir el riesgo de sobreajuste. Permite utilizar eficientemente un conjunto de datos limitado al dividirlo en k subconjuntos (llamados "folds" o pliegues).

El proceso consiste en:

1. División del conjunto de datos: El conjunto de datos se divide en k subconjuntos más pequeños del mismo tamaño. Cada uno de estos subconjuntos se llama "fold".

2. Iteración: Se realiza un bucle a lo largo de k iteraciones. En cada iteración, un fold se selecciona como conjunto de prueba y los k-1 folds restantes se utilizan como conjunto de entrenamiento.

3. Entrenamiento y evaluación: El modelo se entrena en el conjunto de entrenamiento y se evalúa en el conjunto de prueba. Se registra la métrica de rendimiento (por ejemplo, precisión, error cuadrático medio, etc.) para esa iteración.

4. Promedio de resultados: Después de completar las k iteraciones, se promedian las métricas de rendimiento obtenidas en cada iteración para obtener una estimación más robusta del rendimiento del modelo.

Otro aspecto relevante para nuestra fase de validación fue que para los 2 modelos que tuvieron mejores resultados, lo ejecutamos en otra fase de validación pero con 100 archivos csv que nos proporcionaron con el propósito de tener otros datos para evaluar el rendimiento de dichos modelos ante datos nunca utilizados en las fase de construcción y ajuste de los modelos.

Para mayor información sobre esto puede checar el siguiente documento: Evaluation .

7. Método

La metodología de trabajo que el equipo implementó en el proyecto fue Crisp-DM (Cross Industry Standard Process for Data Mining). Es un marco de trabajo que proporciona una estructura para guiar el proceso de minería de datos. Este marco se ha convertido en un estándar en la industria y es utilizado ampliamente para planificar y ejecutar proyectos de minería de datos. CRISP-DM consta de seis fases distintas:

- Comprensión del Negocio (Business Understanding): En esta fase, se busca comprender los objetivos y requisitos del negocio desde una perspectiva de minería de datos. Esto implica definir el problema, los objetivos del proyecto y los criterios de éxito desde la perspectiva del negocio. También se identifican los recursos disponibles y las limitaciones del proyecto.

- Comprensión de los Datos (Data Understanding): Aquí se realiza una exploración inicial de los datos disponibles. Esto incluye la recopilación de datos, la exploración de la calidad de los datos, la identificación de variables relevantes y la comprensión de la distribución de los datos. El objetivo es obtener una visión detallada y completa de los datos que se utilizarán en el análisis.

- Preparación de los Datos (Data Preparation): En esta fase, se lleva a cabo la preparación de los datos para el análisis. Esto puede implicar la limpieza de datos, la selección de variables, la transformación de datos y la generación de nuevas características. El objetivo es tener un conjunto de datos listo para alimentar a los algoritmos de minería de datos.

- Modelado (Modeling): En esta fase, se seleccionan y aplican diversas técnicas de modelado para construir un modelo predictivo o descriptivo. Esto puede incluir el uso de algoritmos de aprendizaje automático, reglas de asociación, clustering, entre otros. Se ajustan los modelos a los datos de entrenamiento y se evalúa su rendimiento.

- Evaluación (Evaluation): Se evalúan los modelos construidos en términos de los objetivos del negocio. Se utilizan métricas de rendimiento para determinar

la eficacia del modelo. En caso necesario, se ajustan o refinan los modelos para mejorar su rendimiento.

- Despliegue (Deployment): Si el modelo es considerado satisfactorio según los criterios de evaluación, se despliega en un entorno operativo. Esto puede implicar la integración del modelo en sistemas existentes, la automatización de procesos o la implementación de herramientas para su uso continuo.

8. Obtención de datos e información relevante

En una primera instancia, se nos informó sobre la sensibilidad de los datos proporcionados por GE, los cuales eran de suma importancia y bajo ninguna circunstancia debían ser expuestos públicamente. Para garantizar la confidencialidad de esta información, procedimos a firmar un acuerdo de no divulgación que aseguraba a GE que no filtraríamos ningún dato proporcionado.

El proyecto de análisis de datos con GE ha sido una empresa que ha requerido una cuidadosa consideración de la sensibilidad de la información, seguido de un riguroso proceso para garantizar la confidencialidad de los datos proporcionados. La firma de un acuerdo de no divulgación subraya el compromiso con la seguridad de la información y la responsabilidad en su manejo.

Para obtener los datos del socio formador, fue imperativo realizarlo a través de una computadora proporcionada por GE. Acceder a esta computadora requería una configuración que tomó aproximadamente 4 semanas. Durante este proceso, obtuvimos acceso a "my tech assistant", la aplicación que nos proporciona acceso a diversas herramientas esenciales para el proyecto, incluyendo anaconda y box sync.

Una vez descargadas estas aplicaciones, los datos estaban almacenados en la carpeta compartida de box sync. Sin embargo, para utilizarlos con los scripts que habíamos desarrollado, era necesario descargar los 2000 archivos localmente.

Durante nuestro análisis, identificamos que las variables "col5_float" y "col13_float" presentaban una significativa multicolinealidad, siendo un valor comúnmente inferior a 5 un indicativo de ausencia de dicho fenómeno.

Además, decidimos eliminar la variable "col17_float" debido a su elevado índice de correlación con "col15_float", alcanzando una correlación de 0.96. Esta alta correlación sugiere que ambas variables son prácticamente idénticas después de filtrar las variables, justificando la decisión de eliminar una de ellas.

Un hallazgo adicional fue identificar las variables más influyentes en la predicción de la fase de "Stable Cruise" después de entrenar y evaluar nuestro modelo. Utilizamos el F-score para evaluar la importancia de las variables, revelando que ciertas variables tuvieron un impacto significativo en el rendimiento del modelo.

Estos descubrimientos resaltan la relevancia de ciertas variables en la predicción de la fase de "Stable Cruise", proporcionando perspectivas valiosas sobre qué características del vuelo son críticas para la estabilidad. En resumen, el proceso involucró medidas rigurosas para salvaguardar la confidencialidad de los datos, una

cuidadosa configuración para acceder a ellos y análisis detallados que condujeron a decisiones fundamentadas sobre las variables a considerar en nuestro modelo predictivo.

9. Construcción de modelos preliminares

Como se mencionó anteriormente, el equipo realizó una investigación sobre posibles algoritmos. Esta investigación fue sobre 3 algoritmos, Logistic Regression y Random Forest.

Se llegó a la conclusión de usar Logistic Regression como modelo benchmark, es decir, será el punto de comparación con otras arquitecturas que implementemos. Esto debido a su rapidez en ejecución y a su compatibilidad con técnicas de regularización que pensamos implementar.

A su vez, después de realizar la limpieza de datos mencionada anteriormente, cada modelo que el equipo construyó fue con las siguientes variables independientes:

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

Y como variable dependiente:

- stableCruise_boolean

Un aspecto importante que se encontró fue el desbalance que había de datos de las clases 0 y 1, el primer acercamiento que se tuvo fue la técnica de oversampling (sobremuestreo) por medio de SMOTE (Synthetic Minority Over-sampling Technique).

SMOTE es una técnica de sobremuestreo utilizada en problemas de desequilibrio de clases en conjuntos de datos. Su objetivo principal es abordar el problema de la falta de representación de la clase minoritaria al generar instancias sintéticas de esa clase.

En el contexto de clasificación binaria, hay dos clases: la mayoría (clase negativa) y la minoría (clase positiva). En muchos conjuntos de datos del mundo real, la clase minoritaria suele tener menos instancias, lo que puede llevar a que el modelo de machine learning tenga dificultades para aprender patrones asociados con esa clase.

De manera general, SMOTE funciona de la siguiente manera:

1. Identificación de Instancias de la Clase Minoritaria:

   SMOTE selecciona instancias individuales de la clase minoritaria en el conjunto de datos original.

2. Selección de Vecinos:

Para cada instancia seleccionada, SMOTE elige k vecinos más cercanos (k es un parámetro definido por el usuario).

3. Generación de Instancias Sintéticas:

   SMOTE crea nuevas instancias sintéticas interpolando características de las instancias seleccionadas y sus vecinos. La idea es crear una combinación ponderada de características de las instancias existentes para generar nuevas instancias que estén en la región de la clase minoritaria.

4. Incorporación de Instancias Sintéticas al Conjunto de Datos:

   Las instancias sintéticas generadas se agregan al conjunto de datos original.

El propósito de este proceso es aumentar la cantidad de instancias de la clase minoritaria, mejorando así la capacidad del modelo para aprender patrones asociados con esa clase. Es importante destacar que SMOTE se debe aplicar únicamente al conjunto de entrenamiento y antes de cualquier división en conjuntos de entrenamiento y prueba para evitar la contaminación de información.

Con esta técnica el equipo ejecutó 3 arquitecturas, Logistic Regression, XGBoost y una red neuronal. Cada arquitectura fue ejecutada con y sin la implementación de la técnica de regularización L2. Como se mencionó en las primeras secciones del documento, se ejecutó un algoritmo para determinar el valor óptimo del parámetro de penalización de L2, el cual nos arrojó que el óptimo de 0.001.

Al observar las métricas de evaluaciones de cada arquitectura, se pudo observar el patrón de que las arquitecturas con la implementación de la técnica L2 tenían un rendimiento ligeramente superior a las arquitecturas sin L2.

Sin embargo, el tiempo de ejecución de los modelos con oversampling era demasiado costoso, llegando a tardar alrededor de 90 minutos. Por consecuencia, el equipo decidió hacer una última iteración pero utilizando la otra técnica para el tratamiento del desbalance de las clases, el cual es undersampling (submuestreo).

En este caso específico, el enfoque que implementamos fue igualar el número de datos en ambas clases, reduciendo la cantidad de instancias de la clase mayoritaria para que coincidan con la cantidad de instancias de la clase minoritaria. Este enfoque se conoce comúnmente como "undersampling equitativo".

A grandes rasgos la implementación que se realizó en nuestro código de undersampling fue el siguiente:

1. Identificación de la Clase Mayoritaria y Minoritaria:

   Identificamos las clases mayoritaria (clase negativa, 0) y minoritaria (clase positiva, 1).

2. Determinación del Tamaño de la Muestra de la Clase Mayoritaria:

Determinamos el número de instancias que se desea mantener para la clase mayoritaria, que establecimos que fuera igual al número de instancias de la clase minoritaria.

3. Selección Aleatoria de Instancias de la Clase Mayoritaria:

   Seleccionamos aleatoriamente un subconjunto de instancias de la clase mayoritaria de modo que su tamaño coincida con el tamaño de la clase minoritaria.

4. Construcción del Conjunto de Datos Equilibrado:

   Las instancias seleccionadas se combinan con todas las instancias de la clase minoritaria para formar el conjunto de datos equilibrado.

El objetivo del undersampling equitativo es mejorar el equilibrio entre las clases para que el modelo no esté sesgado hacia la clase mayoritaria y pueda aprender patrones de ambas clases de manera más equitativa.

En esta iteración, decidimos solo ejecutar una vez cada arquitectura. Dichas ejecuciones fueron con la implementación de L2. Debido al cambio en la técnica, decidimos ejecutar el script para determinar el valor óptimo del parámetro de L2. Para este caso obtuvimos que sería de 0.1.

10. Comparación de los modelos preliminares

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

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 undersampling (submuestreo).

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	94.42%	94.20%	94.17%	37,016	32,972	99%	91%

con L2							
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.

11. Construcción del modelo final

De todas las iteraciones de los diversos modelos que desarrollamos tenemos 2 modelos que tuvieron un rendimiento que sobresaltaron con las diversas métricas de desempeño los cuales se mencionan a continuación:

11.1 Modelo 1. XGBoost con L2 y oversampling.

El primer modelo que estaremos evaluando utilizó la técnica de oversampling. El oversampling es una técnica utilizada en el ámbito del aprendizaje automático, la cual consiste en aumentar la frecuencia de la clase minoritaria, generando instancias sintéticas o replicando instancias existentes. Hay varias formas de implementar el oversampling, la forma en la que implementamos fue por medio de:

- **SMOTE (Synthetic Minority Over-sampling Technique):** es una técnica más avanzada que crea instancias sintéticas de la clase minoritaria mediante la interpolación entre instancias existentes. En lugar de duplicar instancias,

Smote origina nuevas instancias generando puntos en el espacio de características entre instancias de la clase minoritaria vecinas.

```
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 1. Resultados de la ejecución del modelo.

Este modelo posee los siguientes valores acorde a la matriz de confusión:

- **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.

El modelo muestra un buen rendimiento en la clase 0 (negativa) según el informe de clasificación y la matriz de confusión. La precisión para la clase 0 es alta (1.00), y el recall es del 91%, lo que indica que el modelo es eficaz para identificar la mayoría de los casos negativos reales. La matriz de confusión muestra un número significativamente mayor de true negatives (TN) que false negatives (FN), lo que respalda la conclusión de un buen rendimiento para la clase 0.

Sin embargo, para la clase 1 (positiva), el rendimiento es menos satisfactorio. Aunque el recall es del 97%, lo que indica que el modelo identifica la mayoría de los casos positivos reales, la precisión es baja (0.03), lo que significa que la mayoría de las predicciones positivas son falsas. La matriz de confusión muestra que la cantidad de false positives (FP) es mayor que la de true positives (TP). Esto confirma que el modelo tiende a clasificar erróneamente muchos casos como positivos en esta clase.

A pesar de estas diferencias en el rendimiento entre las clases, la precisión general del modelo en el conjunto de prueba es alta (0.906). Esto se debe a la gran proporción de casos negativos en tus datos, lo que puede llevar a una alta precisión incluso si el modelo tiene dificultades para predecir la clase minoritaria.

En resumen, el modelo es efectivo para predecir la clase mayoritaria (clase 0), pero muestra limitaciones en la predicción de la clase minoritaria (clase 1), con un alto número de falsos positivos.

11.2 Modelo 2. XGBoost con L2 y undersampling.

El segundo modelo que estaremos evaluando utilizó la técnica de undersampling. El undersampling 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.

Nuestra implementación de esta técnica fue generar un dataset con la misma cantidad de ejemplos de la clase mayoritaria a la clase minoritaria.

```
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 2. Resultados de la ejecución del modelo.

Este modelo posee los siguientes valores acorde a la matriz de confusión:

- **Verdaderos Positivos (True Positives, TP):** 37,016 predicciones correctas de la clase 1.
- **Falsos Negativos (False Negatives, FN)**: 438 predicciones incorrectas de la clase 0.
- **Falsos Positivos (False Positives, FP):** 3,865 predicciones incorrectas de la clase 1.

- **Verdaderos Negativos (True Negatives, TN):** 32,972 predicciones correctas de la clase 0.

En este conjunto de resultados, el modelo también muestra un rendimiento bastante bueno en general, pero con algunas diferencias en comparación con el caso anterior.

La precisión para la clase 0 es alta (0.99), lo que indica que la gran mayoría de las predicciones negativas son correctas. El recall es del 90%, lo que sugiere que el modelo identifica la mayoría de los casos negativos reales. La matriz de confusión muestra un número mayor de true negatives (TN) que de false negatives (FN), lo cual es positivo y respalda la conclusión de un buen rendimiento para la clase 0.

La precisión para la clase 1 es de 0.91, lo que indica que la mayoría de las predicciones positivas son correctas. Esta es una mejora significativa en comparación con el caso anterior. El recall es del 99%, lo que sugiere que el modelo identifica casi todos los casos positivos reales. También es una mejora sustancial en comparación con el caso anterior. La matriz de confusión muestra un número mayor de true positives (TP) que de false positives (FP), lo cual es positivo y respalda la mejora en el rendimiento para la clase 1.

La precisión general del modelo en el conjunto de prueba es alta (0.942), lo que indica un buen rendimiento en la clasificación general de ambos tipos de clases. Este modelo muestra una mejora en el rendimiento en comparación con el anterior, especialmente en la predicción de la clase 1. La precisión y el recall para ambas clases son notables, y la matriz de confusión refleja un equilibrio entre true positives y true negatives.
En resumen, este modelo demuestra una capacidad sólida para clasificar tanto la clase mayoritaria como la minoritaria, con una mejora significativa en la precisión y recall de la clase 1 en comparación con el modelo anterior.

Comparación inicial de los resultados de los modelos.

Métricas	XGBoost oversampling L2	XGBoost undersampling L2	Modelo Benchmark
Recall clase 1	97%	99%	68%
True Positives	36,093	37,016	25,340
Precisión clase 1	3%	91%	1%
Precisión clase 0	100%	99%	100%
Accuracy	≈ 90%	≈ 94%	≈ 72%

A simple vista, la elección del modelo final sería el modelo de XGBoost con la implementación de undersampling y L2. Debido a que fue el modelo que tuvo

mejores resultados en todos los aspectos desde el accuracy en las fases de entrenamiento, pruebas y validación hasta la identificación de la clase 1 con métricas como recall y true positives.

No obstante, para evitar elecciones con posible sesgo dado que se pueden dar en la implementación de técnicas para el manejo de identificar la eficiencia del modelo de manera equitativa, se usó 100 archivos csv proporcionados por el socio formador. Estos datos se nos fueron proporcionados con este objetivo de que todos los equipos validarán el rendimiento de los modelos debido a que son datos nunca vistos en los primeros 2718 csv que nos brindó para el entrenamiento y creación de los modelos.

Comparación de los resultados con 100 csv de los modelos.

Métricas	XG Boost oversampling L2	XG Boost undersampling L2
Recall	77%	84%
True Positives	15,138	16,543
Precisión clase 1	3%	3%
Precisión clase 0	100%	100%
Accuracy	96.56%	96.21%

En estos nuevos resultados se puede observar que el modelo con undersampling posee una accuracy ligeramente menor al modelo con oversampling, cualquiera de las dos accuracy cumple con nuestro objetivo de minería de datos. Sin embargo, al considerar que el objetivo de negocio consiste en la identificación de stable cruise, es decir, la detección de la clase 1.

El modelo XGBoost con la implementación de L2 y undersampling posee mayor rendimiento para la detección de la clase 1. Esta afirmación se puede corroborar por medio de las métricas que seleccionamos previamente donde el recall de la clase 1 es mayor por 13%, el número de true positives es mayor por 1,405.

En consecuencia, nuestra elección será el modelo con el uso de undersampling puesto que demostró un rendimiento ligeramente mayor al modelo de oversampling.

El modelo final posee las siguientes características:

Algoritmo: XGBoost

Técnica sobre el desequilibrio de las clases: Undersampling (Submuestreo)

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.

12. Resultados

Después de entrenar y evaluar nuestro modelo, hemos identificado las variables más influyentes en la predicción de la fase de "Stable Cruise". La importancia de las variables se evaluó utilizando el F-score, y los resultados revelaron que las siguientes variables tuvieron el mayor impacto en el rendimiento del modelo.

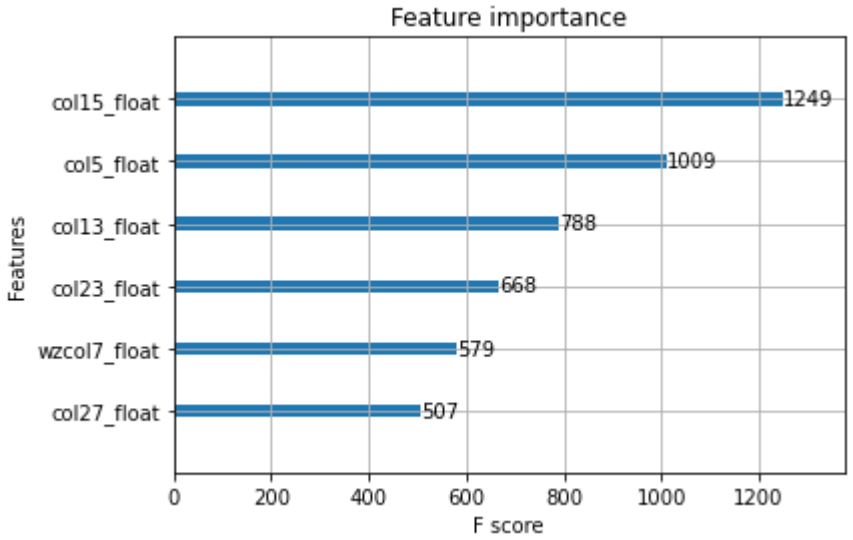

Fig. 1 Importancia de cada variable en F-score (XGBoost Undersampling L2)

De esta gráfica podemos concluir lo siguiente:

col15_float (F-score: 1249): Esta característica tiene el F-score más alto, lo que indica que tiene una influencia significativa en las predicciones del modelo. Podría representar una variable crucial para distinguir entre las dos clases.

col5_float (F-score: 1009): Aunque tiene un F-score más bajo que col15_float, sigue siendo una característica importante y contribuye de manera significativa a las predicciones. La diferencia en los F-scores puede indicar que col15_float tiene un impacto ligeramente mayor en la clasificación.

col13_float (F-score: 788): Esta característica también es importante, pero tiene un F-score más bajo en comparación con las anteriores. Aún así, sigue siendo valiosa para el modelo.

col23_float (F-score: 668): Similar a col13_float, esta característica contribuye al modelo, pero su impacto es menor en comparación con las dos primeras.

wzcol7_float (F-score: 579): Tiene un F-score más bajo, lo que sugiere que su influencia en las predicciones es menor que las características anteriores.

col27_float (F-score: 507): Es la característica con el F-score más bajo en tu lista, lo que indica que su contribución al modelo es la más débil entre las características mencionadas.

```
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. 2 Resultados del modelo final (XGBoost con undersampling y L2)

Accuracy del modelo:

- Entrenamiento: El modelo alcanza una precisión del 94.4% en el conjunto de entrenamiento, lo que indica que es capaz de predecir correctamente la clase de aproximadamente el 94.4% de las instancias en el conjunto de entrenamiento.

- Prueba: La precisión en el conjunto de prueba es del 94.2%, lo que sugiere que el modelo generaliza bien a nuevos datos que no ha visto durante el entrenamiento.
- Validación: La precisión en el conjunto de validación es del 94.2%, lo cual es consistente con el rendimiento en el conjunto de prueba.

Confusion Matrix (Matriz de Confusión):

La matriz de confusión muestra cómo se clasificaron las instancias en el conjunto de prueba:

- **Verdaderos Positivos (True Positives, TP):** 37,016 predicciones correctas de la clase 1.
- **Falsos Negativos (False Negatives, FN)**: 3865 predicciones incorrectas de la clase 0.
- **Falsos Positivos (False Positives, FP):** 32,972 predicciones incorrectas de la clase 1.
- **Verdaderos Negativos (True Negatives, TN):** 438 predicciones correctas de la clase 0.

Classification Report (Informe de Clasificación) para el Conjunto de Prueba:

- Precisión: La precisión indica la proporción de instancias positivas correctamente clasificadas entre todas las instancias clasificadas como positivas. En este caso, la precisión para la clase 0.0 es del 99%, y para la clase 1.0 es del 91%.
- Recall (Sensibilidad): La sensibilidad indica la proporción de instancias positivas correctamente clasificadas con respecto al total de instancias positivas reales. En este caso, el recall para la clase 0.0 es del 90%, y para la clase 1.0 es del 99%.
- F1-score: El F1-score es la media armónica de la precisión y el recall. Es una métrica que equilibra ambos aspectos. El F1-score para la clase 0.0 es del 94%, y para la clase 1.0 es del 95%.

Referencias

- 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.
- *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.*
- 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/