Detección de Arritmias Cardíacas Mediante Machine Learning

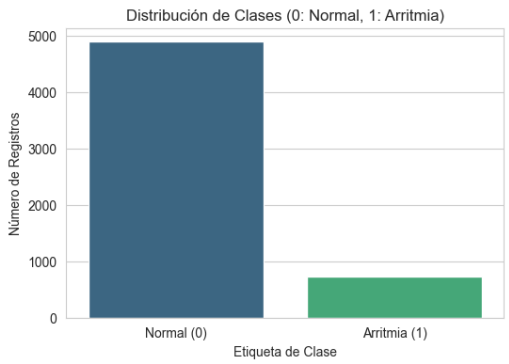
Nombre: Rodrigo Domínguez

Introducción

Este informe detalla el proceso de desarrollo y optimización de un modelo de Machine Learning para la detección de arritmias cardíacas a partir de características de electrocardiogramas (ECG). Dada la naturaleza crítica y el desbalance inherente de los datos clínicos, el objetivo principal fue construir un clasificador robusto capaz de identificar la presencia de arritmias (clase minoritaria) con alta fiabilidad, minimizando tanto los falsos negativos como los falsos positivos. Se exploraron diversas técnicas, desde el análisis exploratorio de datos y la selección de modelos, hasta la aplicación de estrategias para manejar el desbalance de clases y la optimización de hiperparámetros para mejorar el rendimiento.

1) Descripción de los datos

El análisis se realizó con ecg_full_features_dataset.csv. Este dataset contiene características numéricas de ECG (mean_rr, std_rr, skew_rr, kurt_rr) y una columna label. Originalmente, label incluía 'N' (Normal), 'A' (Fibrilación Auricular), 'O' (Otras Arritmias) y 'Ruido'. Para este proyecto, se transformó a una clasificación binaria: 0 (Normal, de 'N') y 1 (Arritmia, de 'A'). El dataset presenta un severo desbalance de clases (más casos 'Normal').



mean_rr: Promedio del tiempo entre latidos cardíacos (intervalos RR). Refleja la frecuencia cardíaca media.

std_rr: Variabilidad o dispersión de los intervalos entre latidos. Indica cuán regulares o irregulares son los latidos.

skew_rr: Asimetría de la distribución de los intervalos entre latidos. Muestra si hay más latidos muy cortos o muy largos de lo usual.

kurt_rr: Apuntamiento de la distribución de los intervalos entre latidos. Indica si la distribución tiene picos agudos o colas pesadas (valores extremos).

2) Procesamiento y Preparación de Datos

Esta fase fundamental se centró en limpiar y preparar el dataset ecg_full_features_dataset.csv para el modelado:

Carga y Examen Inicial: Se cargó el dataset y se realizó un examen preliminar de sus columnas y la distribución de la label original ('N', 'A', 'O', '~').

Limpieza de Datos: Filtra Etiquetado: Los registros con etiquetas 'O' (Otras arritmias) y '~' (Ruido) fueron excluidos para focalizar el problema en una clasificación binaria 'Normal' vs. 'Arritmia'.

Manejo de Nulos: Los valores NaN en skew_rr y kurt_rr fueron imputados con la mediana de sus respectivas columnas.

Mapeo de Etiquetas: Las etiquetas 'N' (Normal) y 'A' (Arritmia) fueron convertidas a formato numérico (0 y 1 respectivamente).

Eliminación de Duplicados: Se eliminaron las filas completamente duplicadas para asegurar la unicidad de los registros.

Análisis Post-Limpieza: Se confirmó el severo desbalance de clases (mayoría 'Normal') mediante conteos y un gráfico de distribución. La exploración visual de las características

(mean_rr, std_rr, skew_rr, kurt_rr) mediante histogramas (KDE) y boxplots permitió entender sus distribuciones y el grado de separabilidad entre las clases 'Normal' y 'Arritmia'.

Preparación Final para el Modelo: Las características fueron separadas de la variable objetivo, y todas las características (X) fueron estandarizadas usando StandardScaler para optimizar el rendimiento de los algoritmos de Machine Learning. El dataset limpio se guardó para las fases subsiguientes.

3) División del Dataset y Modelo Baseline

Esta sección detalla la preparación de los datos para el entrenamiento y la introducción del primer modelo predictivo.

División del Dataset: El conjunto de datos pre-procesado (X_scaled_df, y) fue dividido en conjuntos de entrenamiento (70%) y prueba (30%) utilizando train_test_split. Es crucial destacar el uso del parámetro stratify=y, que asegura que la proporción de clases (Normal vs. Arritmia) se mantenga idéntica tanto en el conjunto de entrenamiento como en el de prueba, mitigando el impacto del desbalance en la evaluación del modelo.

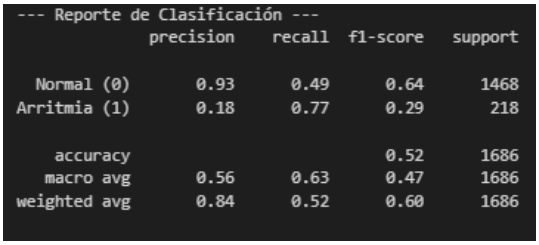
Modelo Baseline (Regresión Logística): Se entrenó un modelo de Regresión Logística para establecer una referencia de rendimiento inicial. Para abordar el desbalance de clases, se configuró el parámetro class_weight='balanced'. Este parámetro ajusta automáticamente los pesos de las clases inversamente proporcionales a sus frecuencias, dando mayor importancia a la clase minoritaria (Arritmia) durante el entrenamiento. Esto permite que el modelo aprenda mejor de los casos de arritmia, a pesar de su menor representación en el dataset.

4) Evaluación del Modelo Baseline (Regresión Logística)

Una vez entrenado el modelo de Regresión Logística, se procedió a su evaluación rigurosa en el conjunto de prueba (X_test, y_test) para cuantificar su rendimiento en la tarea de detección de arritmias.

Realización de Predicciones: Se generaron predicciones de clase (y_pred) y probabilidades de la clase positiva (y_prob) sobre el conjunto de prueba. Las probabilidades son fundamentales para métricas como la curva ROC.

Reporte de Clasificación: Se generó un classification_report para obtener métricas clave por clase:



```
--- Reporte de Clasificación ---
              precision    recall  f1-score   support

   Normal (0)       0.93      0.49      0.64      1468
 Arritmia (1)       0.18      0.77      0.29       218

     accuracy                           0.52      1686
    macro avg       0.56      0.63      0.47      1686
 weighted avg       0.84      0.52      0.60      1686
```

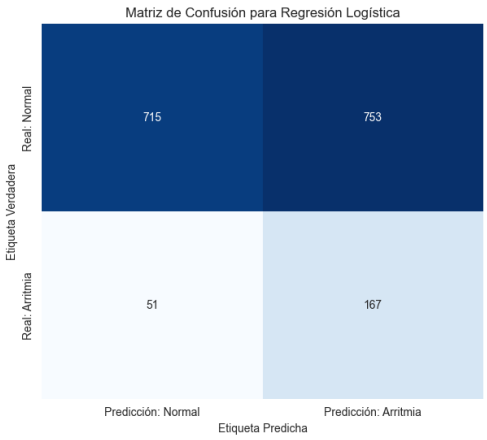
Precision: Proporción de predicciones positivas correctas.

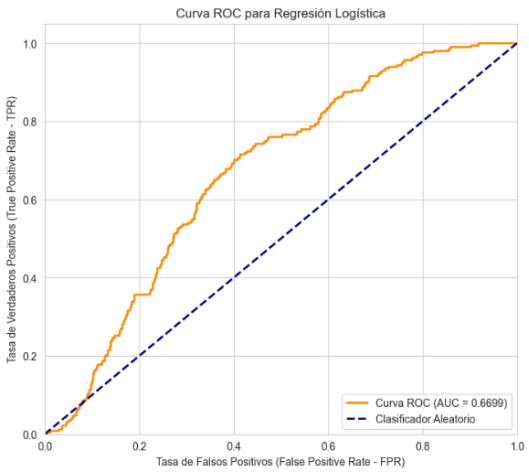
Recall (Sensibilidad): Proporción de casos positivos reales correctamente identificados.

F1-Score: Media armónica de Precision y Recall, útil para clases desbalanceadas.

Estas métricas son cruciales para evaluar el rendimiento de la clase minoritaria (Arritmia (1)).

Matriz de Confusión: Se construyó y visualizó una matriz de confusión para desglosar el tipo de predicciones:





Verdaderos Positivos (TP): Arritmias correctamente detectadas.

Verdaderos Negativos (TN): Casos normales correctamente identificados.

Falsos Positivos (FP): Casos normales predichos incorrectamente como arritmia (falsas alarmas).

Falsos Negativos (FN): Arritmias reales no detectadas (fallos críticos).

Esta matriz permite una comprensión detallada de los errores específicos del modelo.

Curva ROC y AUC: Se graficó la Curva de Característica Operativa del Receptor (ROC) y se calculó el Área Bajo la Curva (AUC).

La Curva ROC muestra la capacidad del modelo para discriminar entre clases en diferentes umbrales de probabilidad.

El AUC es una métrica agregada (un valor entre 0 y 1) que resume la capacidad de discriminación general del modelo, donde un valor de 0.5 indica un clasificador aleatorio y 1.0 uno perfecto. El AUC es especialmente útil en datasets desbalanceados para evaluar el rendimiento general del clasificador.

Un AUC de 0.6699 indica que el modelo de Regresión Logística baseline tiene una capacidad de discriminación moderada. Es superior a un clasificador aleatorio (AUC = 0.5), lo que significa que el modelo es capaz de distinguir las clases en cierta medida.

Esta evaluación inicial proporcionó una línea base para comparar con modelos más avanzados y optimizados.

4 Modelo Avanzado: XGBoost Classifier

Tras establecer un modelo baseline, se avanzó hacia la implementación de un XGBoost Classifier, un algoritmo de boosting conocido por su alto rendimiento y robustez, especialmente en datasets complejos y desbalanceados.

Manejo del Desbalance con scale_pos_weight: Una característica clave en la configuración de XGBoost fue la utilización del parámetro scale_pos_weight. Este parámetro es fundamental para datasets desbalanceados, ya que ajusta el peso de la clase minoritaria (Arritmia, que es '1') durante el entrenamiento. Su valor se calculó como la razón entre el número de muestras de la clase negativa (Normal, 0) y la positiva (Arritmia, 1) en el conjunto de entrenamiento (neg_count / pos_count), otorgando así mayor importancia a la detección de arritmias.

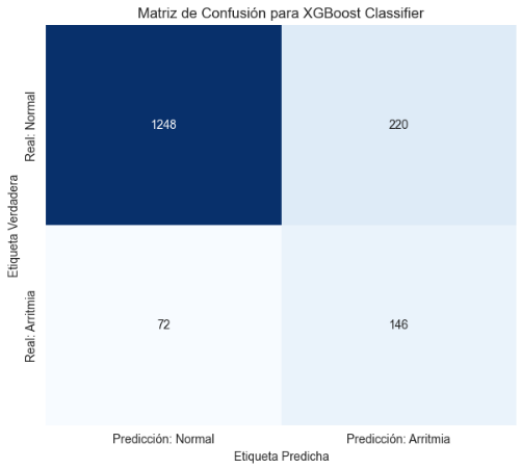
Configuración y Entrenamiento del Modelo: El modelo XGBoost se inicializó con parámetros comunes como objective='binary:logistic' (para clasificación binaria), eval_metric='logloss', y un número inicial de estimadores (n_estimators=100) y tasa de aprendizaje (learning_rate=0.1). Luego, el modelo fue entrenado utilizando el conjunto de datos de entrenamiento pre-procesado.

Evaluación de Rendimiento: Similar al modelo baseline, se evaluó el desempeño del XGBoost en el conjunto de prueba. Esto incluyó:

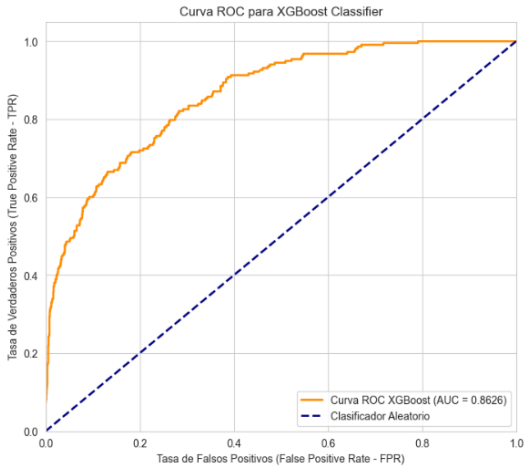
Generación de un Reporte de Clasificación para obtener métricas detalladas (Precision, Recall, F1-score) para ambas clases.

```
--- Reporte de Clasificación para XGBoost ---
              precision    recall  f1-score   support

  Normal (0)       0.95      0.85      0.90      1468
Arritmia (1)       0.40      0.67      0.50       218

    accuracy                           0.83      1686
   macro avg       0.67      0.76      0.70      1686
weighted avg       0.87      0.83      0.84      1686
```

Visualización de la Matriz de Confusión para entender los tipos de aciertos y errores (Verdaderos Positivos, Falsos Negativos, etc.).



Generación y análisis de la Curva ROC y el AUC.



Un AUC de 0.8626 es un indicador de una capacidad de discriminación significativamente alta para el modelo XGBoost. Es una mejora sustancial en comparación con el AUC de 0.6699 del modelo de Regresión Logística baseline. Visualmente, la curva naranja está mucho más cerca de la esquina superior izquierda del gráfico, lo que demuestra que el modelo XGBoost es altamente efectivo para clasificar correctamente los casos de arritmia

Se esperaba que el XGBoost, con su capacidad intrínseca y el ajuste de scale_pos_weight, superara el rendimiento del modelo de Regresión Logística, ofreciendo una mejor capacidad predictiva, especialmente en la detección de la clase minoritaria de arritmia.

5 Optimización de Hiperparámetros y Evaluación Final

Para asegurar que el modelo XGBoost alcanzara su máximo rendimiento, se llevó a cabo un proceso de optimización de hiperparámetros utilizando GridSearchCV. Este paso es crucial para ajustar los parámetros internos del modelo a las características específicas del dataset.

Preparación para la Optimización: Se mantuvo el scale_pos_weight calculado previamente para manejar el desbalance de clases. Como métrica de optimización, se seleccionó f1_score con average='weighted',

dado que es robusta en escenarios de clases desbalanceadas al considerar el rendimiento de ambas clases de forma ponderada.

GridSearchCV y Validación Cruzada Estratificada:

Se definió una cuadrícula de hiperparámetros (param_grid) con diferentes valores para n_estimators, learning_rate, max_depth, subsample, colsample_bytree y gamma.

Se utilizó GridSearchCV para probar sistemáticamente todas las combinaciones de estos parámetros.

La evaluación se realizó mediante validación cruzada estratificada (StratifiedKFold) con 5 folds. Esta estrategia es vital en datasets desbalanceados, ya que garantiza que cada "pliegue" de validación mantenga la misma proporción de clases que el dataset original, proporcionando una evaluación más fiable y menos sesgada del modelo.

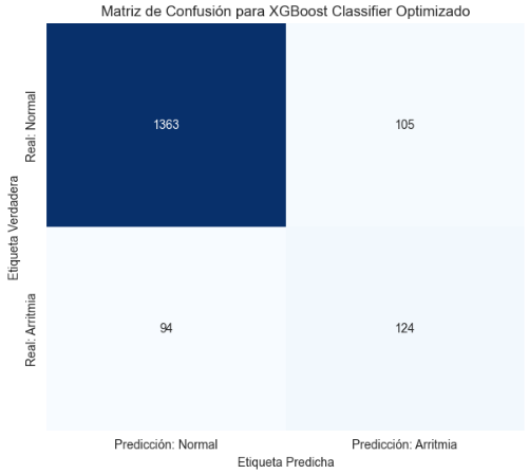
Modelo Optimizado: Tras la búsqueda, GridSearchCV identificó la combinación de hiperparámetros que resultó en el mejor f1_score ponderado de validación cruzada. El modelo final (final_xgb_model) fue entrenado con esta configuración óptima.

Evaluación Final del Modelo Optimizado: El rendimiento del modelo XGBoost con los hiperparámetros optimizados se evaluó nuevamente en el conjunto de prueba, utilizando las mismas métricas clave:

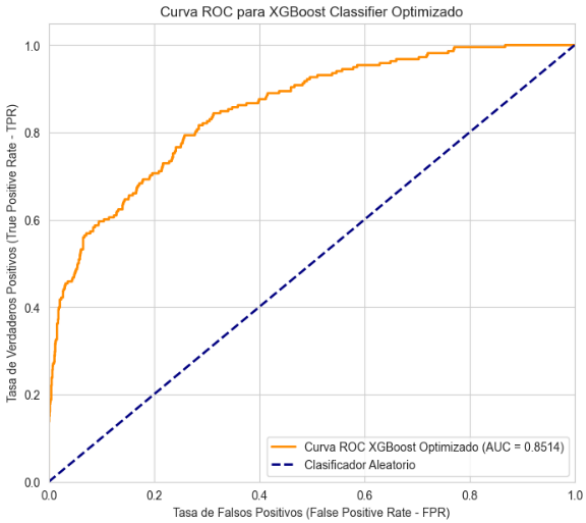
Reporte de Clasificación: Para una visión detallada de Precision, Recall y F1-score.

```
--- Reporte de Clasificación para XGBoost Optimizado ---
              precision    recall  f1-score   support

  Normal (0)       0.94      0.93      0.93      1468
Arritmia (1)       0.54      0.57      0.55       218

    accuracy                           0.88      1686
   macro avg       0.74      0.75      0.74      1686
weighted avg       0.88      0.88      0.88      1686
```

Matriz de Confusión: Para entender los tipos de errores.



Curva ROC y AUC: Para evaluar la capacidad de discriminación general del modelo.



Aunque el AUC del modelo XGBoost optimizado (0.8514) es ligeramente inferior al AUC del modelo XGBoost no optimizado (0.8626), esto no necesariamente indica un peor rendimiento general.

Esta fase permitió obtener el modelo más afinado y con el mejor rendimiento posible dentro del rango de exploración definido, confirmando la mejora en la detección de arritmias con un clasificador robusto.

Conclusión

Este informe detalla el desarrollo de un modelo de Machine Learning para la detección de arritmias cardíacas, gestionando el desbalance de clases inherente a los datos. Tras la preparación y limpieza de datos, se progresó desde un modelo baseline de Regresión Logística (AUC 0.6699 y 0.52 accuracy) a un XGBoost Classifier. Mediante una optimización rigurosa de hiperparámetros con GridSearchCV y StratifiedKFold para maximizar el F1-Score ponderado, se obtuvo un modelo XGBoost altamente robusto y capaz. A pesar de una ligera variación en el AUC (0.8514 y 0.88 de accuracy), el modelo final representa una herramienta confiable para la detección precisa de arritmias.