Módulo 2: Análisis y reporte sobre el desempeño del modelo

Daniel Saldaña Rodríguez
A00829752

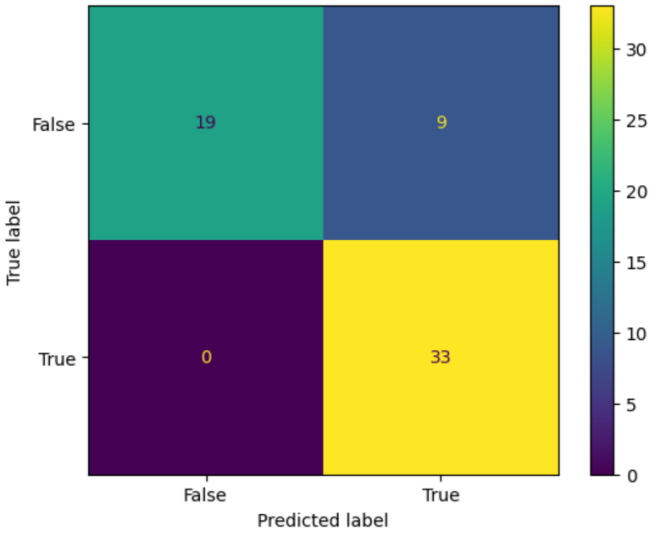
A continuación se llevará a cabo un análisis del rendimiento del modelo de aprendizaje implementado con herramientas externas.

Para esta implementación de un método de aprendizaje de datos se seleccionó una base de datos proveniente de Kaggle, esta base cuenta con 13 variables que son usadas para predecir si el paciente tiene un riesgo elevado de sufrir un ataque al corazón. Hay solo dos posibles resultados, 0 (No hay un riego elevado), y 1 (Hay un riego elevado), por lo que este es un problema de clasificación binaria. Entre estas variables podemos encontrar la edad, sexo, niveles de colesterol, tipo de dolor de pecho y nivel de azucar entre otras. La base de datos puede ser encontrada en esta liga:
https://www.kaggle.com/datasets/rashikrahmanpritom/heart-attack-analysis-prediction-dataset

La separación se llevó a cabo utilizando la función train_test_split de scikit learn. Se utilizó un ratio de 80-20 entre los datos de entrenamiento y los de prueba. Adicionalmente se utilizó el parámetro ‘stratify’ para asegurarse que ambas partes tuvieran proporciones similares de datos respecto a la variable de respuesta.

El modelo que brindó los mejores resultados fue el de Gradient Boosting, con las siguientes métricas:

- Accuracy: 0.8525
- Precision: 0.7857
- Recall: 1.0
- F1: 0.88



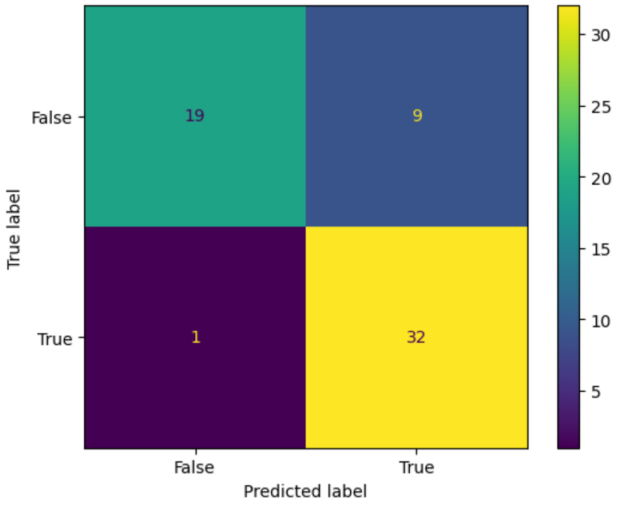
También se utilizó la función cross_val_score para poner a prueba el modelo más a fondo. La accuracy promedio fue de 78.5459%.

Gradient Boosting es un modelo que es vulnerable al overfitting, para detectar este fenómeno podemos comparar la accuracy de training con la de testing. Al hacer esto podemos ver que hay una gran diferencia:

- Accuracy train: 0.9256
- Accuracy test: 0.8525

Para mejorar la situación se hizo un ajuste al modelo, se añadió el parámetro 'min_samples_split', el cual designa el número mínimo de muestras necesarias para dividir un nodo interno. Este valor se denominó como 100. Con este cambio se consiguieron estos resultados:

- Accuracy train: 0.8554
- Accuracy test: 0.8361
- Precision: 0.7805
- Recall: 0.9697
- F1: 0.8649



Podemos ver que ahora los valores de accuracy usando los datos de training y testing son mucho más cercanos por lo que hay menos overfitting, aunque ahora el modelo comete un error más por lo que las métricas bajan. Sin embargo, al usar cross_val_score una vez más, vemos que este cambio fue definitivamente positivo, ya que el promedio de accuracy sube a 82.2449%