Momento de Retroalimentación: Módulo 2 Análisis y Reporte sobre el desempeño del modelo. (Portafolio Análisis)

Introducción:

La base de datos es sobre el precios de telefonos y celulares con varias características como la marca, el tamaño, la memoria, el RAM entre otras, lo que se buscara es predecir "la gama" del dispositivo que está dividido en "Alta:2", "Media:1" y "Economico:0".

El Modelo: **El modelo que use fue el Gradient Boosting**

Principales Hiper Parámetros del Gradient Boosting Classifier

n_estimators:
Descripción: Número de árboles en el modelo.
Impacto: Más árboles suelen mejorar la precisión pero pueden causar sobreajuste y hacer el modelo más lento.

learning_rate:
Descripción: Tasa de ajuste en cada iteración.
Impacto: Una tasa baja requiere más árboles para el mismo rendimiento, mejorando la generalización. Una tasa alta ajusta más rápido pero puede ser menos estable.

max_depth:
Descripción: Profundidad máxima de los árboles.
Impacto: Profundidades mayores capturan patrones complejos pero pueden causar sobreajuste. Menores profundidades pueden llevar a un ajuste insuficiente (underfitting).

min_samples_split:
Descripción: Muestras mínimas requeridas para dividir un nodo.
Impacto: Un valor alto evita divisiones en nodos pequeños, ayudando a prevenir el sobreajuste.

min_samples_leaf:
Descripción: Muestras mínimas en un nodo hoja.
Impacto: Asegura que los nodos hoja tengan suficientes muestras, reduciendo el riesgo de sobreajuste.

subsample:
Descripción: Proporción de muestras usadas para ajustar cada árbol.
Impacto: Menor proporción reduce el sobreajuste y mejora la robustez del modelo.

max_features:
Descripción: Número máximo de características consideradas para dividir un nodo.
Impacto: Limitar características ayuda a reducir el sobreajuste y mejora la generalización.

Para la selección de los hiperparametros se usó GridSearch donde se pusieron varios hiperparametros para probar y aunque el modelo que mejor accuracy se registró tenia 5% más que el modelo final cuando se hacian los demas análisis resultaba tener un nivel muy alto de overfitting, por lo que al final decidí usar unos valores que no tenían tanta accuracy reducían mucho el overfitting, por lo que el análisis se hará cono este modelo.

Diagnóstico y explicación el grado de bias o sesgo: bajo medio alto
El sesgo en un modelo de machine learning mide cuán lejos están las predicciones del modelo de los valores reales en promedio. Un modelo con alto sesgo es demasiado simple y no captura bien los patrones en los datos, lo que puede llevar a errores en las predicciones tanto en los datos de entrenamiento como en los de prueba (underfitting). Por otro lado, un modelo con bajo sesgo es más flexible y puede ajustarse mejor a los datos, pero si es demasiado complejo, puede ajustarse demasiado a los datos de entrenamiento y fallar en los datos nuevos (overfitting).

```
Exactitud en el conjunto de entrenamiento: 0.75
Exactitud en el conjunto de prueba: 0.71
Sesgo medio (El modelo está capturando la relación, pero puede mejorar)
```

Diagnóstico y explicación el grado de varianza: bajo medio alto
La varianza en un modelo mide cuánto cambian las predicciones del modelo para diferentes conjuntos de datos de entrenamiento. Una varianza alta indica que el modelo es muy sensible a pequeñas fluctuaciones en los datos, lo que puede llevar a sobreajuste (overfitting).

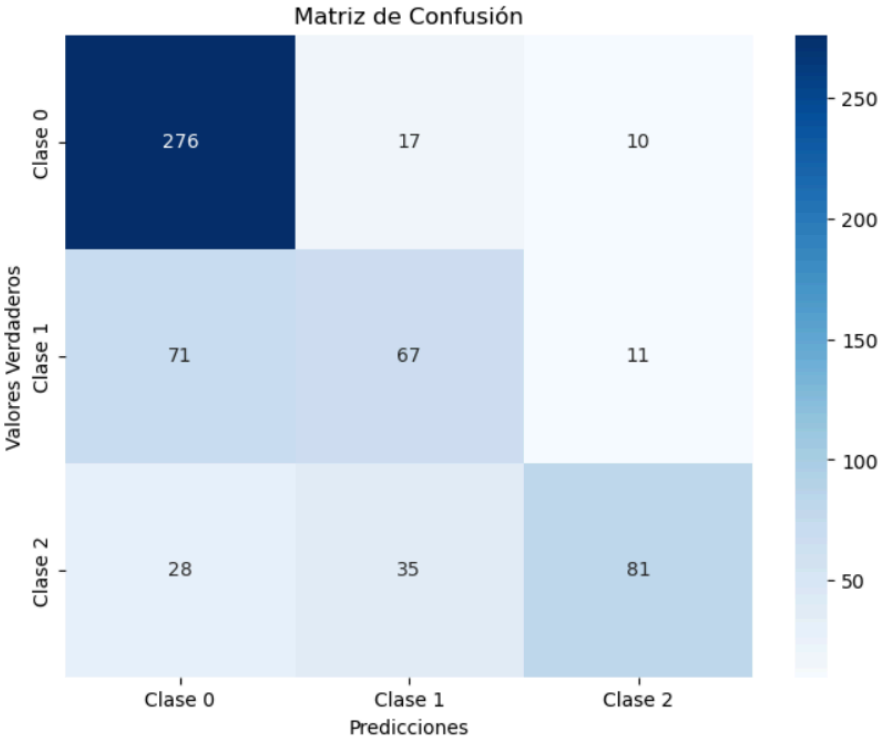
```
Error Cuadrático Medio (MSE): 0.4798657718120805
Grado de varianza: Alto
```

Métricas Importantes:
Precisión (Accuracy): Mide el rendimiento global.
Precisión (Precision): Evalúa la exactitud de las predicciones positivas.
Recuerdo (Recall): Mide la capacidad de detección de la clase positiva.
F1-Score: Balancea precisión y recuerdo para un rendimiento equilibrado

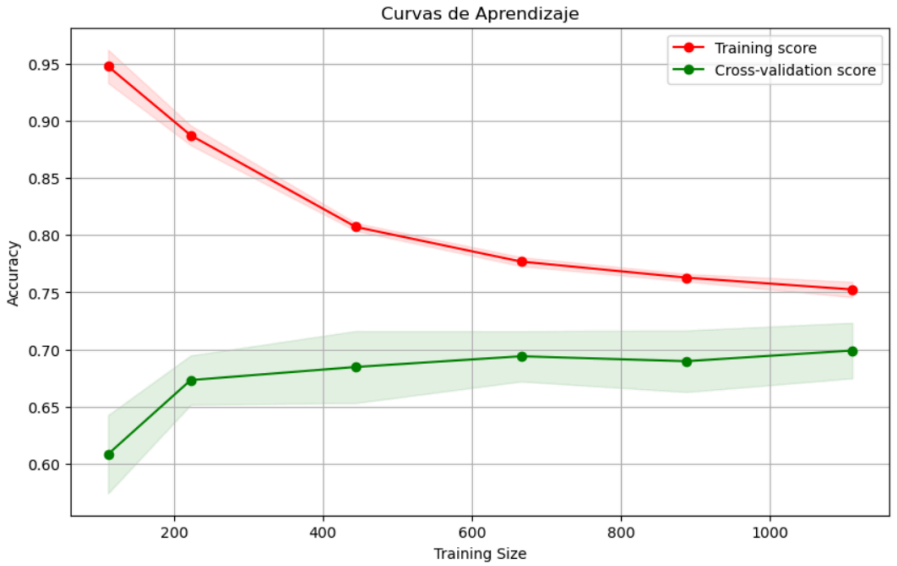
Análisis y comparación de varias métricas en cuanto al Train y el Test

```
Accuracy en entrenamiento: 0.75
Accuracy en prueba: 0.71
Diferencia en accuracy: 0.04
F1-Score en entrenamiento: 0.74
F1-Score en prueba: 0.70
Diferencia en F1-Score: 0.04
Precision en entrenamiento: 0.75
Precision en prueba: 0.71
Diferencia en Precision: 0.04
Recall en entrenamiento: 0.75
Recall en prueba: 0.71
Diferencia en Recall: 0.04
Nivel de ajuste del modelo: Fitting
```

Matriz de Confusión



Comparación de las curvas de aprendizaje:



Conclusiones:

En general el modelo no fue malo y después de evaluar varios métodos el de Gradient Boosting fue el mejor, creo que el tema de el overfitting y el error que tenia el modelo en predecir la clase correcta se debe a que, como se puede observar, la mayoría de los valores se encuentran en un rango de precios más cercano a la clase 0 y a la clase 1, tomando en cuenta el gran sesgo de los precios ya que un 75% de los valores se encuentra por debajo del rango que conforma la clase 1, la poca variabilidad en los componentes que había y la gran cantidad de datos que componen un celular que no estaban incluidos en la base de datos , además de que sabemos que hay marcas que pueden inflar su precio solo por el simple hecho de tener un “prestigio” , siento que explicaría mucho el porque sin importar cómo ajustaba los hiper parámetros el modelo no presentaba una mejora significativa.

Es por esto que veo muy entendible que en la matriz de confusión la mayor parte de los errores hayan sido debido al modelo confundiendo las 2 primeras clases. Mientras que en las curvas de aprendizaje se puede observar como set de entrenamiento empieza con un overfitting muy alto, siendo estos valores de “buena” accuracy los que encontraba el GridSearch, pero gracias a los hiperparametros que implemente ese overfitting disminuyo rápidamente llegando a acercarse mucho al cross-validation score.