Checkpoint 2: Árbol de decisión

Objetivo:

Como objetivo principal para este checkpoint, debemos poder entrenar un modelo de árbol de decisión y realizar una predicción con datos de prueba.

Busqueda de hyperparámetros:

Previo a realizar un análisis de de hyperparámetros, transformamos las variables cualitativas en variables cuantitativas utilizando el método de oneHotEncoding y filtrando las columnas que no eran relevantes para nuestro target. Luego, realizamos una búsqueda de hyperparametros con Random Search para poder optimizar al máximo la predicción del árbol enfocandonos principalmente en el f-score, ya que con esto estaríamos ajustando tanto los valores de precisión como también de recall. Además, realizamos una división del dataset en 80% de los datos para entrenamiento y un 20% de los datos para prueba.

Observación: *Optamos por el método de Random Search y no por el GridSearch ya que este último demoraba mucho en relación al anterior y no apreciamos cambios significativos en la métrica que queríamos optimizar*

Para poder correr el Random Search, utilizamos 20 iteraciones con 5 folds con los siguientes juegos de parámetros:

```
#Conjunto de parámetros que quiero usar
params_grid = {'criterion':['gini','entropy'],
               'min_samples_leaf':list(range(1,5)),
               'min_samples_split': list(range(2,10)),
               'ccp_alpha':np.linspace(0.001,0.05,n),
               'max_depth':list(range(1,20))}
```

Con estos valores obtenemos una métrica de f-score entre 0.79 y 0.85. A continuación, dejamos una de las mejores combinaciones:

```
# {'min_samples_split': 15, 'min_samples_leaf': 3, 'max_depth': 15, 'criterion': 'gini',
'ccp_alpha': 0.0001}
# 0.8537731251732715
```

Entrenamiento:

Una vez obtenidos los mejores hyperparámetros, realizamos el entrenamiento del árbol y realizamos una predicción con los datos de prueba. En nuestra experiencia, los datos de prueba performaron de forma similar que con los datos de entrenamiento, por lo que concluimos que el modelo realiza una buena predicción y no quedó sobreentrenado ya que nuestras métricas no pasaban los valores aprox de f-score 0.9.

Dejamos a continuación, uno de las iteraciones de nuestro modelo:
Métrica en datos de entrenamiento

```
# {'min_samples_split': 5, 'min_samples_leaf': 1, 'max_depth': 7, 'criterion': 'entropy',
'ccp_alpha': 0.001}
# 0.8302306919606145
```

Métrica en datos de test

```
# Accuracy: 0.8293302729731945
# Recall: 0.8155102040816327
# Precision: 0.83991928703548
# f1 score: 0.8275347912524852
```

Matriz de confusión