Bosques_VelasquezLuna_EliJafet

June 3, 2024

[27]:
```python
import pandas as pd
import numpy as np
```

[2]:
```python
url = "https://archive.ics.uci.edu/ml/machine-learning-databases/statlog/heart/
 ↪heart.dat"

# Columnas en general
column_names = [
    "age", "sex", "chest_pain", "resting_bp", "cholesterol", "fasting_bs",␣
 ↪"rest_ecg",
    "max_heart_rate", "exercise_angina", "oldpeak", "slope", "num_vessels",␣
 ↪"thal", "target"
]

# Columnas categóricas
column_cat = ['chest_pain', 'rest_ecg', 'thal']

# Cargar csv
tabla = pd.read_csv(url, sep=" ", names=column_names)
```

[3]:
```python
tabla
```

[3]:
```
      age  sex  chest_pain  resting_bp  cholesterol  fasting_bs  rest_ecg  \
0    70.0  1.0         4.0       130.0        322.0         0.0       2.0
1    67.0  0.0         3.0       115.0        564.0         0.0       2.0
2    57.0  1.0         2.0       124.0        261.0         0.0       0.0
3    64.0  1.0         4.0       128.0        263.0         0.0       0.0
4    74.0  0.0         2.0       120.0        269.0         0.0       2.0
..    …    …           …           …            …           …         …
265  52.0  1.0         3.0       172.0        199.0         1.0       0.0
266  44.0  1.0         2.0       120.0        263.0         0.0       0.0
267  56.0  0.0         2.0       140.0        294.0         0.0       2.0
268  57.0  1.0         4.0       140.0        192.0         0.0       0.0
269  67.0  1.0         4.0       160.0        286.0         0.0       2.0

     max_heart_rate  exercise_angina  oldpeak  slope  num_vessels  thal  \
0             109.0              0.0      2.4    2.0          3.0   3.0
```

```
1            160.0             0.0     1.6    2.0            0.0   7.0
2            141.0             0.0     0.3    1.0            0.0   7.0
3            105.0             1.0     0.2    2.0            1.0   7.0
4            121.0             1.0     0.2    1.0            1.0   3.0
..             …            …      …      …          …    …
265          162.0             0.0     0.5    1.0            0.0   7.0
266          173.0             0.0     0.0    1.0            0.0   7.0
267          153.0             0.0     1.3    2.0            0.0   3.0
268          148.0             0.0     0.4    2.0            0.0   6.0
269          108.0             1.0     1.5    2.0            3.0   3.0

     target
0         2
1         1
2         2
3         1
4         1
..      …
265       1
266       1
267       1
268       1
269       2

[270 rows x 14 columns]
```

[19]:

```python
# Número de categorías por columna
tabla[column_cat].nunique()
```

[19]:

```
chest_pain    4
rest_ecg      3
thal          3
dtype: int64
```

0.1 Transformar columnas categóricas a binarias con OneHotEncoder

[4]:

```python
from sklearn.preprocessing import OneHotEncoder
```

[8]:

```python
codificador = OneHotEncoder()
codificador.fit(tabla[column_cat])
```

[8]:

```
OneHotEncoder()
```

[9]:

```python
# Columnas categóricas transformadas
column_cat_OneHot = codificador.transform(tabla[column_cat])
column_cat_OneHot.toarray()
```

```
[9]: array([[0., 0., 0., …, 1., 0., 0.],
       [0., 0., 1., …, 0., 0., 1.],
       [0., 1., 0., …, 0., 0., 1.],
       …,
       [0., 1., 0., …, 1., 0., 0.],
       [0., 0., 0., …, 0., 1., 0.],
       [0., 0., 0., …, 1., 0., 0.]])
```

```
[22]: # Usando pandas get_dummies para conservar la estructura de tabla
tabla_encoded = pd.get_dummies(tabla, prefix=column_cat, columns=column_cat,␣
 ↪drop_first=False)
tabla_encoded
```

```
[22]:      age  sex  resting_bp  cholesterol  fasting_bs  max_heart_rate  \
0     70.0  1.0       130.0        322.0         0.0           109.0
1     67.0  0.0       115.0        564.0         0.0           160.0
2     57.0  1.0       124.0        261.0         0.0           141.0
3     64.0  1.0       128.0        263.0         0.0           105.0
4     74.0  0.0       120.0        269.0         0.0           121.0
..     …    …         …           …          …             …
265   52.0  1.0       172.0        199.0         1.0           162.0
266   44.0  1.0       120.0        263.0         0.0           173.0
267   56.0  0.0       140.0        294.0         0.0           153.0
268   57.0  1.0       140.0        192.0         0.0           148.0
269   67.0  1.0       160.0        286.0         0.0           108.0

     exercise_angina  oldpeak  slope  num_vessels  …  chest_pain_1.0  \
0                0.0      2.4    2.0          3.0  …           False
1                0.0      1.6    2.0          0.0  …           False
2                0.0      0.3    1.0          0.0  …           False
3                1.0      0.2    2.0          1.0  …           False
4                1.0      0.2    1.0          1.0  …           False
..               …      …    …          …  …             …
265              0.0      0.5    1.0          0.0  …           False
266              0.0      0.0    1.0          0.0  …           False
267              0.0      1.3    2.0          0.0  …           False
268              0.0      0.4    2.0          0.0  …           False
269              1.0      1.5    2.0          3.0  …           False

     chest_pain_2.0  chest_pain_3.0  chest_pain_4.0  rest_ecg_0.0  \
0             False           False            True         False
1             False            True           False         False
2              True           False           False          True
3             False           False            True          True
4              True           False           False         False
..              …              …              …            …
265           False            True           False          True
```

```
266           True           False           False           True
267           True           False           False          False
268          False           False            True           True
269          False           False            True          False

     rest_ecg_1.0  rest_ecg_2.0  thal_3.0  thal_6.0  thal_7.0
0           False          True      True     False     False
1           False          True     False     False      True
2           False         False     False     False      True
3           False         False     False     False      True
4           False          True      True     False     False
..            …             …        …        …        …
265         False         False     False     False      True
266         False         False     False     False      True
267         False          True      True     False     False
268         False         False     False      True     False
269         False          True      True     False     False

[270 rows x 21 columns]
```

0.2 División de datos

[23]:
```python
from sklearn.model_selection import train_test_split

X = tabla_encoded.drop('target', axis=1)
y = tabla_encoded.target

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,␣
↪random_state=42)
```

1 Bosques

[25]:
```python
from sklearn.model_selection import GridSearchCV
from sklearn.metrics import classification_report
```

1.1 Árbol de Potenciación de Gradiente

[26]:
```python
from sklearn.ensemble import GradientBoostingClassifier
```

[56]:
```python
cuadricula_parametros = [{
    'criterion' : ['friedman_mse', 'squared_error'],
    'max_depth' : range(2,11,5),
    'max_features' : np.linspace(0.1,1,5)
}]
```

[57]:
```
# Optimizar hiperparámetros
buscadorCuadricula = GridSearchCV(GradientBoostingClassifier(),␣
↪cuadricula_parametros,
                                  cv=5, scoring="accuracy")
```

[58]:
```
buscadorCuadricula.fit(X_train,y_train)
```

[58]:
```
GridSearchCV(cv=5, estimator=GradientBoostingClassifier(),
             param_grid=[{'criterion': ['friedman_mse', 'squared_error'],
                          'max_depth': range(2, 11, 5),
                          'max_features': array([0.1  , 0.325, 0.55 , 0.775, 1.
])}],
             scoring='accuracy')
```

[59]:
```
# Mejores hiperparámetros
buscadorCuadricula.best_params_
```

[59]:
```
{'criterion': 'squared_error', 'max_depth': 2, 'max_features': 0.1}
```

[60]:
```
# Entrenar un modelo con los mejores parámetros
gbc = GradientBoostingClassifier(**buscadorCuadricula.best_params_)
gbc.fit(X_train,y_train)
```

[60]:
```
GradientBoostingClassifier(criterion='squared_error', max_depth=2,
                           max_features=0.1)
```

[61]:
```
gbc.estimators_
```

[61]:
```
array([[DecisionTreeRegressor(max_depth=2, max_features=0.1,
                              random_state=RandomState(MT19937) at
0x7E8488D5A140)],
       [DecisionTreeRegressor(max_depth=2, max_features=0.1,
                              random_state=RandomState(MT19937) at
0x7E8488D5A140)],
       [DecisionTreeRegressor(max_depth=2, max_features=0.1,
                              random_state=RandomState(MT19937) at
0x7E8488D5A140)],
       [DecisionTreeRegressor(max_depth=2, max_features=0.1,
                              random_state=RandomState(MT19937) at
0x7E8488D5A140)],
       [DecisionTreeRegressor(max_depth=2, max_features=0.1,
                              random_state=RandomState(MT19937) at
0x7E8488D5A140)],
       [DecisionTreeRegressor(max_depth=2, max_features=0.1,
                              random_state=RandomState(MT19937) at
0x7E8488D5A140)],
       [DecisionTreeRegressor(max_depth=2, max_features=0.1,
```

```
                              random_state=RandomState(MT19937) at
0x7E8488D5A140)],
       [DecisionTreeRegressor(max_depth=2, max_features=0.1,
                              random_state=RandomState(MT19937) at
0x7E8488D5A140)],
       [DecisionTreeRegressor(max_depth=2, max_features=0.1,
                              random_state=RandomState(MT19937) at
0x7E8488D5A140)],
       [DecisionTreeRegressor(max_depth=2, max_features=0.1,
                              random_state=RandomState(MT19937) at
0x7E8488D5A140)],
       [DecisionTreeRegressor(max_depth=2, max_features=0.1,
                              random_state=RandomState(MT19937) at
0x7E8488D5A140)],
       [DecisionTreeRegressor(max_depth=2, max_features=0.1,
                              random_state=RandomState(MT19937) at
0x7E8488D5A140)],
       [DecisionTreeRegressor(max_depth=2, max_features=0.1,
                              random_state=RandomState(MT19937) at
0x7E8488D5A140)],
       [DecisionTreeRegressor(max_depth=2, max_features=0.1,
                              random_state=RandomState(MT19937) at
0x7E8488D5A140)],
       [DecisionTreeRegressor(max_depth=2, max_features=0.1,
                              random_state=RandomState(MT19937) at
0x7E8488D5A140)],
       [DecisionTreeRegressor(max_depth=2, max_features=0.1,
                              random_state=RandomState(MT19937) at
0x7E8488D5A140)],
       [DecisionTreeRegressor(max_depth=2, max_features=0.1,
                              random_state=RandomState(MT19937) at
0x7E8488D5A140)],
       [DecisionTreeRegressor(max_depth=2, max_features=0.1,
                              random_state=RandomState(MT19937) at
0x7E8488D5A140)],
       [DecisionTreeRegressor(max_depth=2, max_features=0.1,
                              random_state=RandomState(MT19937) at
0x7E8488D5A140)],
       [DecisionTreeRegressor(max_depth=2, max_features=0.1,
                              random_state=RandomState(MT19937) at
0x7E8488D5A140)],
       [DecisionTreeRegressor(max_depth=2, max_features=0.1,
                              random_state=RandomState(MT19937) at
0x7E8488D5A140)],
       [DecisionTreeRegressor(max_depth=2, max_features=0.1,
                              random_state=RandomState(MT19937) at
0x7E8488D5A140)],
```

```
       [DecisionTreeRegressor(max_depth=2, max_features=0.1,
                              random_state=RandomState(MT19937) at
0x7E8488D5A140)],
       [DecisionTreeRegressor(max_depth=2, max_features=0.1,
                              random_state=RandomState(MT19937) at
0x7E8488D5A140)],
       [DecisionTreeRegressor(max_depth=2, max_features=0.1,
                              random_state=RandomState(MT19937) at
0x7E8488D5A140)],
       [DecisionTreeRegressor(max_depth=2, max_features=0.1,
                              random_state=RandomState(MT19937) at
0x7E8488D5A140)],
       [DecisionTreeRegressor(max_depth=2, max_features=0.1,
                              random_state=RandomState(MT19937) at
0x7E8488D5A140)],
       [DecisionTreeRegressor(max_depth=2, max_features=0.1,
                              random_state=RandomState(MT19937) at
0x7E8488D5A140)],
       [DecisionTreeRegressor(max_depth=2, max_features=0.1,
                              random_state=RandomState(MT19937) at
0x7E8488D5A140)],
       [DecisionTreeRegressor(max_depth=2, max_features=0.1,
                              random_state=RandomState(MT19937) at
0x7E8488D5A140)],
       [DecisionTreeRegressor(max_depth=2, max_features=0.1,
                              random_state=RandomState(MT19937) at
0x7E8488D5A140)],
       [DecisionTreeRegressor(max_depth=2, max_features=0.1,
                              random_state=RandomState(MT19937) at
0x7E8488D5A140)],
       [DecisionTreeRegressor(max_depth=2, max_features=0.1,
                              random_state=RandomState(MT19937) at
0x7E8488D5A140)],
       [DecisionTreeRegressor(max_depth=2, max_features=0.1,
                              random_state=RandomState(MT19937) at
0x7E8488D5A140)],
       [DecisionTreeRegressor(max_depth=2, max_features=0.1,
                              random_state=RandomState(MT19937) at
0x7E8488D5A140)],
       [DecisionTreeRegressor(max_depth=2, max_features=0.1,
                              random_state=RandomState(MT19937) at
0x7E8488D5A140)],
       [DecisionTreeRegressor(max_depth=2, max_features=0.1,
                              random_state=RandomState(MT19937) at
```

```
0x7E8488D5A140)],
       [DecisionTreeRegressor(max_depth=2, max_features=0.1,
                              random_state=RandomState(MT19937) at
0x7E8488D5A140)],
       [DecisionTreeRegressor(max_depth=2, max_features=0.1,
                              random_state=RandomState(MT19937) at
0x7E8488D5A140)],
       [DecisionTreeRegressor(max_depth=2, max_features=0.1,
                              random_state=RandomState(MT19937) at
0x7E8488D5A140)],
       [DecisionTreeRegressor(max_depth=2, max_features=0.1,
                              random_state=RandomState(MT19937) at
0x7E8488D5A140)],
       [DecisionTreeRegressor(max_depth=2, max_features=0.1,
                              random_state=RandomState(MT19937) at
0x7E8488D5A140)],
       [DecisionTreeRegressor(max_depth=2, max_features=0.1,
                              random_state=RandomState(MT19937) at
0x7E8488D5A140)],
       [DecisionTreeRegressor(max_depth=2, max_features=0.1,
                              random_state=RandomState(MT19937) at
0x7E8488D5A140)],
       [DecisionTreeRegressor(max_depth=2, max_features=0.1,
                              random_state=RandomState(MT19937) at
0x7E8488D5A140)],
       [DecisionTreeRegressor(max_depth=2, max_features=0.1,
                              random_state=RandomState(MT19937) at
0x7E8488D5A140)],
       [DecisionTreeRegressor(max_depth=2, max_features=0.1,
                              random_state=RandomState(MT19937) at
0x7E8488D5A140)],
       [DecisionTreeRegressor(max_depth=2, max_features=0.1,
                              random_state=RandomState(MT19937) at
0x7E8488D5A140)],
       [DecisionTreeRegressor(max_depth=2, max_features=0.1,
                              random_state=RandomState(MT19937) at
0x7E8488D5A140)],
       [DecisionTreeRegressor(max_depth=2, max_features=0.1,
                              random_state=RandomState(MT19937) at
0x7E8488D5A140)],
       [DecisionTreeRegressor(max_depth=2, max_features=0.1,
                              random_state=RandomState(MT19937) at
0x7E8488D5A140)],
       [DecisionTreeRegressor(max_depth=2, max_features=0.1,
```

```
                               random_state=RandomState(MT19937) at
0x7E8488D5A140)],
       [DecisionTreeRegressor(max_depth=2, max_features=0.1,
                               random_state=RandomState(MT19937) at
0x7E8488D5A140)],
       [DecisionTreeRegressor(max_depth=2, max_features=0.1,
                               random_state=RandomState(MT19937) at
0x7E8488D5A140)],
       [DecisionTreeRegressor(max_depth=2, max_features=0.1,
                               random_state=RandomState(MT19937) at
0x7E8488D5A140)],
       [DecisionTreeRegressor(max_depth=2, max_features=0.1,
                               random_state=RandomState(MT19937) at
0x7E8488D5A140)],
       [DecisionTreeRegressor(max_depth=2, max_features=0.1,
                               random_state=RandomState(MT19937) at
0x7E8488D5A140)],
       [DecisionTreeRegressor(max_depth=2, max_features=0.1,
                               random_state=RandomState(MT19937) at
0x7E8488D5A140)],
       [DecisionTreeRegressor(max_depth=2, max_features=0.1,
                               random_state=RandomState(MT19937) at
0x7E8488D5A140)],
       [DecisionTreeRegressor(max_depth=2, max_features=0.1,
                               random_state=RandomState(MT19937) at
0x7E8488D5A140)],
       [DecisionTreeRegressor(max_depth=2, max_features=0.1,
                               random_state=RandomState(MT19937) at
0x7E8488D5A140)],
       [DecisionTreeRegressor(max_depth=2, max_features=0.1,
                               random_state=RandomState(MT19937) at
0x7E8488D5A140)],
       [DecisionTreeRegressor(max_depth=2, max_features=0.1,
                               random_state=RandomState(MT19937) at
0x7E8488D5A140)],
       [DecisionTreeRegressor(max_depth=2, max_features=0.1,
                               random_state=RandomState(MT19937) at
0x7E8488D5A140)],
       [DecisionTreeRegressor(max_depth=2, max_features=0.1,
                               random_state=RandomState(MT19937) at
0x7E8488D5A140)],
       [DecisionTreeRegressor(max_depth=2, max_features=0.1,
                               random_state=RandomState(MT19937) at
0x7E8488D5A140)],
```

```
       [DecisionTreeRegressor(max_depth=2, max_features=0.1,
                              random_state=RandomState(MT19937) at
0x7E8488D5A140)],
       [DecisionTreeRegressor(max_depth=2, max_features=0.1,
                              random_state=RandomState(MT19937) at
0x7E8488D5A140)],
       [DecisionTreeRegressor(max_depth=2, max_features=0.1,
                              random_state=RandomState(MT19937) at
0x7E8488D5A140)],
       [DecisionTreeRegressor(max_depth=2, max_features=0.1,
                              random_state=RandomState(MT19937) at
0x7E8488D5A140)],
       [DecisionTreeRegressor(max_depth=2, max_features=0.1,
                              random_state=RandomState(MT19937) at
0x7E8488D5A140)],
       [DecisionTreeRegressor(max_depth=2, max_features=0.1,
                              random_state=RandomState(MT19937) at
0x7E8488D5A140)],
       [DecisionTreeRegressor(max_depth=2, max_features=0.1,
                              random_state=RandomState(MT19937) at
0x7E8488D5A140)],
       [DecisionTreeRegressor(max_depth=2, max_features=0.1,
                              random_state=RandomState(MT19937) at
0x7E8488D5A140)],
       [DecisionTreeRegressor(max_depth=2, max_features=0.1,
                              random_state=RandomState(MT19937) at
0x7E8488D5A140)],
       [DecisionTreeRegressor(max_depth=2, max_features=0.1,
                              random_state=RandomState(MT19937) at
0x7E8488D5A140)],
       [DecisionTreeRegressor(max_depth=2, max_features=0.1,
                              random_state=RandomState(MT19937) at
0x7E8488D5A140)],
       [DecisionTreeRegressor(max_depth=2, max_features=0.1,
                              random_state=RandomState(MT19937) at
0x7E8488D5A140)],
       [DecisionTreeRegressor(max_depth=2, max_features=0.1,
                              random_state=RandomState(MT19937) at
0x7E8488D5A140)],
       [DecisionTreeRegressor(max_depth=2, max_features=0.1,
                              random_state=RandomState(MT19937) at
0x7E8488D5A140)],
       [DecisionTreeRegressor(max_depth=2, max_features=0.1,
                              random_state=RandomState(MT19937) at
```

```
0x7E8488D5A140)],
       [DecisionTreeRegressor(max_depth=2, max_features=0.1,
                              random_state=RandomState(MT19937) at
0x7E8488D5A140)],
       [DecisionTreeRegressor(max_depth=2, max_features=0.1,
                              random_state=RandomState(MT19937) at
0x7E8488D5A140)],
       [DecisionTreeRegressor(max_depth=2, max_features=0.1,
                              random_state=RandomState(MT19937) at
0x7E8488D5A140)],
       [DecisionTreeRegressor(max_depth=2, max_features=0.1,
                              random_state=RandomState(MT19937) at
0x7E8488D5A140)],
       [DecisionTreeRegressor(max_depth=2, max_features=0.1,
                              random_state=RandomState(MT19937) at
0x7E8488D5A140)],
       [DecisionTreeRegressor(max_depth=2, max_features=0.1,
                              random_state=RandomState(MT19937) at
0x7E8488D5A140)],
       [DecisionTreeRegressor(max_depth=2, max_features=0.1,
                              random_state=RandomState(MT19937) at
0x7E8488D5A140)],
       [DecisionTreeRegressor(max_depth=2, max_features=0.1,
                              random_state=RandomState(MT19937) at
0x7E8488D5A140)],
       [DecisionTreeRegressor(max_depth=2, max_features=0.1,
                              random_state=RandomState(MT19937) at
0x7E8488D5A140)],
       [DecisionTreeRegressor(max_depth=2, max_features=0.1,
                              random_state=RandomState(MT19937) at
0x7E8488D5A140)],
       [DecisionTreeRegressor(max_depth=2, max_features=0.1,
                              random_state=RandomState(MT19937) at
0x7E8488D5A140)],
       [DecisionTreeRegressor(max_depth=2, max_features=0.1,
                              random_state=RandomState(MT19937) at
0x7E8488D5A140)],
       [DecisionTreeRegressor(max_depth=2, max_features=0.1,
                              random_state=RandomState(MT19937) at
0x7E8488D5A140)],
       [DecisionTreeRegressor(max_depth=2, max_features=0.1,
                              random_state=RandomState(MT19937) at
0x7E8488D5A140)],
       [DecisionTreeRegressor(max_depth=2, max_features=0.1,
                              random_state=RandomState(MT19937) at
0x7E8488D5A140)]],
      dtype=object)
```

[62]:
```
# Predecir con el modelo óptimo
y_pred = gbc.predict(X_test)
```

[63]:
```
# Métricas de clasificación
print(classification_report(y_test,y_pred))
```

```
              precision    recall  f1-score   support

           1       0.89      0.97      0.93        33
           2       0.94      0.81      0.87        21

    accuracy                           0.91        54
   macro avg       0.92      0.89      0.90        54
weighted avg       0.91      0.91      0.91        54
```

Debido al tiempo de entrenamiento y búsqueda que toma el GridSearchCV, sólo se escogieron 3 hiperparámetros a optimizar. Se encontró que estos toman los valores:

- criterion='squared_error'
- max_depth=2
- max_features=0.1

El modelo con estos parámetros obtuvo un Accuracy de 91%, e indica una preferencia por arboles menos profundos (profundidad de 2) lo que ayudaría a evitar el sobreajuste o ser menos sensible al ruido. Del mismo modo un 10% en max_features indicaría una mayor aleatoreidad al momento de construir los árboles ayudando a producir un modelo más general.

1.2 Bosque Aleatorio

[65]:
```
from sklearn.ensemble import RandomForestClassifier
```

[66]:
```
cuadricula_parametros = [{
    'criterion' : ['gini', 'entropy', 'log_loss'],
    'max_samples' : np.linspace(0.1,1,5),
    'max_features' : np.linspace(0.1,1,5)
}]
```

[67]:
```
# Optimizar hiperparámetros
buscadorCuadricula = GridSearchCV(RandomForestClassifier(),␣
 ↪cuadricula_parametros,
                                  cv=5, scoring="accuracy")
buscadorCuadricula.fit(X_train,y_train)
```

[67]:
```
GridSearchCV(cv=5, estimator=RandomForestClassifier(),
             param_grid=[{'criterion': ['gini', 'entropy', 'log_loss'],
                          'max_features': array([0.1  , 0.325, 0.55 , 0.775, 1.
]),
```

```
                          'max_samples': array([0.1  , 0.325, 0.55 , 0.775, 1.
])}],
             scoring='accuracy')
```

[68]:
```
# Mejores hiperparámetros
buscadorCuadricula.best_params_
```

[68]:
```
{'criterion': 'gini', 'max_features': 0.775, 'max_samples': 0.1}
```

[69]:
```
# Entrenar un modelo con los mejores parámetros
bosque = RandomForestClassifier(**buscadorCuadricula.best_params_)
bosque.fit(X_train,y_train)
```

[69]:
```
RandomForestClassifier(max_features=0.775, max_samples=0.1)
```

[70]:
```
bosque.estimators_
```

[70]:
```
[DecisionTreeClassifier(max_features=0.775, random_state=1233183326),
 DecisionTreeClassifier(max_features=0.775, random_state=1818962143),
 DecisionTreeClassifier(max_features=0.775, random_state=788998151),
 DecisionTreeClassifier(max_features=0.775, random_state=1043834719),
 DecisionTreeClassifier(max_features=0.775, random_state=223631265),
 DecisionTreeClassifier(max_features=0.775, random_state=984774144),
 DecisionTreeClassifier(max_features=0.775, random_state=1406376100),
 DecisionTreeClassifier(max_features=0.775, random_state=1059516044),
 DecisionTreeClassifier(max_features=0.775, random_state=280608137),
 DecisionTreeClassifier(max_features=0.775, random_state=2105594227),
 DecisionTreeClassifier(max_features=0.775, random_state=1139690940),
 DecisionTreeClassifier(max_features=0.775, random_state=1032805442),
 DecisionTreeClassifier(max_features=0.775, random_state=49282198),
 DecisionTreeClassifier(max_features=0.775, random_state=175533462),
 DecisionTreeClassifier(max_features=0.775, random_state=1435483587),
 DecisionTreeClassifier(max_features=0.775, random_state=80052807),
 DecisionTreeClassifier(max_features=0.775, random_state=1480515958),
 DecisionTreeClassifier(max_features=0.775, random_state=1082781354),
 DecisionTreeClassifier(max_features=0.775, random_state=993240468),
 DecisionTreeClassifier(max_features=0.775, random_state=1501589853),
 DecisionTreeClassifier(max_features=0.775, random_state=361466079),
 DecisionTreeClassifier(max_features=0.775, random_state=1015057216),
 DecisionTreeClassifier(max_features=0.775, random_state=1542418632),
 DecisionTreeClassifier(max_features=0.775, random_state=1091260929),
 DecisionTreeClassifier(max_features=0.775, random_state=1355963083),
 DecisionTreeClassifier(max_features=0.775, random_state=1548295410),
 DecisionTreeClassifier(max_features=0.775, random_state=452561781),
 DecisionTreeClassifier(max_features=0.775, random_state=1267519345),
 DecisionTreeClassifier(max_features=0.775, random_state=1886893069),
 DecisionTreeClassifier(max_features=0.775, random_state=130868101),
```

```
DecisionTreeClassifier(max_features=0.775, random_state=996935099),
DecisionTreeClassifier(max_features=0.775, random_state=393348935),
DecisionTreeClassifier(max_features=0.775, random_state=761336560),
DecisionTreeClassifier(max_features=0.775, random_state=1401751047),
DecisionTreeClassifier(max_features=0.775, random_state=765498767),
DecisionTreeClassifier(max_features=0.775, random_state=1444366259),
DecisionTreeClassifier(max_features=0.775, random_state=1627671881),
DecisionTreeClassifier(max_features=0.775, random_state=2126450226),
DecisionTreeClassifier(max_features=0.775, random_state=773555963),
DecisionTreeClassifier(max_features=0.775, random_state=540344592),
DecisionTreeClassifier(max_features=0.775, random_state=206634580),
DecisionTreeClassifier(max_features=0.775, random_state=692116172),
DecisionTreeClassifier(max_features=0.775, random_state=2083400313),
DecisionTreeClassifier(max_features=0.775, random_state=663438234),
DecisionTreeClassifier(max_features=0.775, random_state=181227131),
DecisionTreeClassifier(max_features=0.775, random_state=1778225551),
DecisionTreeClassifier(max_features=0.775, random_state=1700853388),
DecisionTreeClassifier(max_features=0.775, random_state=1414299727),
DecisionTreeClassifier(max_features=0.775, random_state=1786013016),
DecisionTreeClassifier(max_features=0.775, random_state=59138408),
DecisionTreeClassifier(max_features=0.775, random_state=465278249),
DecisionTreeClassifier(max_features=0.775, random_state=1853737257),
DecisionTreeClassifier(max_features=0.775, random_state=1858115361),
DecisionTreeClassifier(max_features=0.775, random_state=1278145804),
DecisionTreeClassifier(max_features=0.775, random_state=2133462305),
DecisionTreeClassifier(max_features=0.775, random_state=1114986734),
DecisionTreeClassifier(max_features=0.775, random_state=1561094529),
DecisionTreeClassifier(max_features=0.775, random_state=334808184),
DecisionTreeClassifier(max_features=0.775, random_state=1080376414),
DecisionTreeClassifier(max_features=0.775, random_state=880648983),
DecisionTreeClassifier(max_features=0.775, random_state=1457386169),
DecisionTreeClassifier(max_features=0.775, random_state=1619003608),
DecisionTreeClassifier(max_features=0.775, random_state=806944074),
DecisionTreeClassifier(max_features=0.775, random_state=1023837127),
DecisionTreeClassifier(max_features=0.775, random_state=261438171),
DecisionTreeClassifier(max_features=0.775, random_state=1161350487),
DecisionTreeClassifier(max_features=0.775, random_state=786436484),
DecisionTreeClassifier(max_features=0.775, random_state=644093377),
DecisionTreeClassifier(max_features=0.775, random_state=850735829),
DecisionTreeClassifier(max_features=0.775, random_state=974047461),
DecisionTreeClassifier(max_features=0.775, random_state=884692807),
DecisionTreeClassifier(max_features=0.775, random_state=56417657),
DecisionTreeClassifier(max_features=0.775, random_state=944613313),
DecisionTreeClassifier(max_features=0.775, random_state=2091035775),
DecisionTreeClassifier(max_features=0.775, random_state=1652244605),
DecisionTreeClassifier(max_features=0.775, random_state=1633437036),
DecisionTreeClassifier(max_features=0.775, random_state=129888640),
```

```
 DecisionTreeClassifier(max_features=0.775, random_state=1139419876),
 DecisionTreeClassifier(max_features=0.775, random_state=1474410805),
 DecisionTreeClassifier(max_features=0.775, random_state=1312433182),
 DecisionTreeClassifier(max_features=0.775, random_state=2113085620),
 DecisionTreeClassifier(max_features=0.775, random_state=1958640297),
 DecisionTreeClassifier(max_features=0.775, random_state=794990923),
 DecisionTreeClassifier(max_features=0.775, random_state=1008218788),
 DecisionTreeClassifier(max_features=0.775, random_state=763310197),
 DecisionTreeClassifier(max_features=0.775, random_state=1623567709),
 DecisionTreeClassifier(max_features=0.775, random_state=2033826087),
 DecisionTreeClassifier(max_features=0.775, random_state=307061608),
 DecisionTreeClassifier(max_features=0.775, random_state=1181836900),
 DecisionTreeClassifier(max_features=0.775, random_state=1875344981),
 DecisionTreeClassifier(max_features=0.775, random_state=632547612),
 DecisionTreeClassifier(max_features=0.775, random_state=66891184),
 DecisionTreeClassifier(max_features=0.775, random_state=1376038088),
 DecisionTreeClassifier(max_features=0.775, random_state=1897354354),
 DecisionTreeClassifier(max_features=0.775, random_state=1106050160),
 DecisionTreeClassifier(max_features=0.775, random_state=15799959),
 DecisionTreeClassifier(max_features=0.775, random_state=1755844843),
 DecisionTreeClassifier(max_features=0.775, random_state=1880697334),
 DecisionTreeClassifier(max_features=0.775, random_state=53727088),
 DecisionTreeClassifier(max_features=0.775, random_state=447687543)]
```

[71]:

```
# Predecir con el modelo óptimo
y_pred = bosque.predict(X_test)
```

[72]:

```
# Métricas de clasificación
print(classification_report(y_test,y_pred))
```

```
              precision    recall  f1-score   support

           1       0.84      0.94      0.89        33
           2       0.88      0.71      0.79        21

    accuracy                           0.85        54
   macro avg       0.86      0.83      0.84        54
weighted avg       0.86      0.85      0.85        54
```

En este caso se encontraron los siguientes valores:

- criterion='gini'
- max_samples=0.1
- max_features=0.775

El modelo con estos parámetros obtuvo un Accuracy de 85%. Con una mayor predicción de la categoría 1 que igualmente tiene mayor soporte. A diferencia del arbol de potenciación de gradiente, el bosque aleatorio toma el 77.5% de las características, lo que implica menos variabilidad en las características usadas; aunque se obtuvo el 10% de máximas muestras, lo que permite explorar la contribución de más grupos de muestras separados.

Para incrementar el accuracy, sería conveniente explorar otros hiperparámetros.