# Análisis de Datos y Aprendizaje Máquina con Tensorflow 2.0: Clasificación

2019/09/30

## Árboles de decision ID3

- Objetivo: Conocer los arboles ID3 para clasificación y como visualizarlos
- Los árboles de decisión (DT) son un método de aprendizaje supervisado relacionado con la entropía, se utiliza para la clasificación y la regresión. El algoritmo hace particiones en las características de los datos de forma que los va clasificando
- Los árboles de decisión son muy interpretables, lo que puede ser muy útil con algunos conjuntos de datos, pues indican que variable difiere de otra en cuanto a la cantidad de datos que se particionan

```
In [1]: import sklearn
        import pandas as pd
        import seaborn as sns
        import matplotlib.pyplot as plt
```

### Análisis exploratorio

#### Etiquetas de clase a valor numérico

- Diagnosis (M = malignant, B = benign)

```
In [2]: %matplotlib inline

        df = pd.read_csv("data-breast.csv",index_col=0)

        df.head(10)
```

```
Out[2]:          diagnosis  radius_mean  texture_mean  perimeter_mean  area_mean  \
        id
        842302           M        17.99         10.38          122.80     1001.0
        842517           M        20.57         17.77          132.90     1326.0
        84300903         M        19.69         21.25          130.00     1203.0
        84348301         M        11.42         20.38           77.58      386.1
        84358402         M        20.29         14.34          135.10     1297.0
        843786           M        12.45         15.70           82.57      477.1
        844359           M        18.25         19.98          119.60     1040.0
```

| | | | | | |
|---|---|---|---|---|---|
| 84458202 | M | 13.71 | 20.83 | 90.20 | 577.9 |
| 844981 | M | 13.00 | 21.82 | 87.50 | 519.8 |
| 84501001 | M | 12.46 | 24.04 | 83.97 | 475.9 |

| id | smoothness_mean | compactness_mean | concavity_mean | \ |
|---|---|---|---|---|
| 842302 | 0.11840 | 0.27760 | 0.30010 | |
| 842517 | 0.08474 | 0.07864 | 0.08690 | |
| 84300903 | 0.10960 | 0.15990 | 0.19740 | |
| 84348301 | 0.14250 | 0.28390 | 0.24140 | |
| 84358402 | 0.10030 | 0.13280 | 0.19800 | |
| 843786 | 0.12780 | 0.17000 | 0.15780 | |
| 844359 | 0.09463 | 0.10900 | 0.11270 | |
| 84458202 | 0.11890 | 0.16450 | 0.09366 | |
| 844981 | 0.12730 | 0.19320 | 0.18590 | |
| 84501001 | 0.11860 | 0.23960 | 0.22730 | |

| id | concave points_mean | symmetry_mean | ... | texture_worst | \ |
|---|---|---|---|---|---|
| | | | ... | | |
| 842302 | 0.14710 | 0.2419 | ... | 17.33 | |
| 842517 | 0.07017 | 0.1812 | ... | 23.41 | |
| 84300903 | 0.12790 | 0.2069 | ... | 25.53 | |
| 84348301 | 0.10520 | 0.2597 | ... | 26.50 | |
| 84358402 | 0.10430 | 0.1809 | ... | 16.67 | |
| 843786 | 0.08089 | 0.2087 | ... | 23.75 | |
| 844359 | 0.07400 | 0.1794 | ... | 27.66 | |
| 84458202 | 0.05985 | 0.2196 | ... | 28.14 | |
| 844981 | 0.09353 | 0.2350 | ... | 30.73 | |
| 84501001 | 0.08543 | 0.2030 | ... | 40.68 | |

| id | perimeter_worst | area_worst | smoothness_worst | compactness_worst | \ |
|---|---|---|---|---|---|
| 842302 | 184.60 | 2019.0 | 0.1622 | 0.6656 | |
| 842517 | 158.80 | 1956.0 | 0.1238 | 0.1866 | |
| 84300903 | 152.50 | 1709.0 | 0.1444 | 0.4245 | |
| 84348301 | 98.87 | 567.7 | 0.2098 | 0.8663 | |
| 84358402 | 152.20 | 1575.0 | 0.1374 | 0.2050 | |
| 843786 | 103.40 | 741.6 | 0.1791 | 0.5249 | |
| 844359 | 153.20 | 1606.0 | 0.1442 | 0.2576 | |
| 84458202 | 110.60 | 897.0 | 0.1654 | 0.3682 | |
| 844981 | 106.20 | 739.3 | 0.1703 | 0.5401 | |
| 84501001 | 97.65 | 711.4 | 0.1853 | 1.0580 | |

| id | concavity_worst | concave points_worst | symmetry_worst | \ |
|---|---|---|---|---|
| 842302 | 0.7119 | 0.2654 | 0.4601 | |
| 842517 | 0.2416 | 0.1860 | 0.2750 | |
| 84300903 | 0.4504 | 0.2430 | 0.3613 | |
| 84348301 | 0.6869 | 0.2575 | 0.6638 | |

```
84358402            0.4000                0.1625          0.2364
843786              0.5355                0.1741          0.3985
844359              0.3784                0.1932          0.3063
84458202            0.2678                0.1556          0.3196
844981              0.5390                0.2060          0.4378
84501001            1.1050                0.2210          0.4366

          fractal_dimension_worst  Unnamed: 32
id
842302                    0.11890          NaN
842517                    0.08902          NaN
84300903                  0.08758          NaN
84348301                  0.17300          NaN
84358402                  0.07678          NaN
843786                    0.12440          NaN
844359                    0.08368          NaN
84458202                  0.11510          NaN
844981                    0.10720          NaN
84501001                  0.20750          NaN

[10 rows x 32 columns]
```

In [3]:
```
df.iloc[:,1:].describe()
```

Out[3]:
```
       radius_mean  texture_mean  perimeter_mean    area_mean  \
count   569.000000    569.000000      569.000000   569.000000
mean     14.127292     19.289649       91.969033   654.889104
std       3.524049      4.301036       24.298981   351.914129
min       6.981000      9.710000       43.790000   143.500000
25%      11.700000     16.170000       75.170000   420.300000
50%      13.370000     18.840000       86.240000   551.100000
75%      15.780000     21.800000      104.100000   782.700000
max      28.110000     39.280000      188.500000  2501.000000

       smoothness_mean  compactness_mean  concavity_mean  concave points_mean  \
count       569.000000        569.000000      569.000000           569.000000
mean          0.096360          0.104341        0.088799             0.048919
std           0.014064          0.052813        0.079720             0.038803
min           0.052630          0.019380        0.000000             0.000000
25%           0.086370          0.064920        0.029560             0.020310
50%           0.095870          0.092630        0.061540             0.033500
75%           0.105300          0.130400        0.130700             0.074000
max           0.163400          0.345400        0.426800             0.201200

       symmetry_mean  fractal_dimension_mean  ...  texture_worst  \
count     569.000000              569.000000  ...     569.000000
mean        0.181162                0.062798  ...      25.677223
std         0.027414                0.007060  ...       6.146258
min         0.106000                0.049960  ...      12.020000
25%         0.161900                0.057700  ...      21.080000
```

```
50%         0.179200                0.061540  ...       25.410000
75%         0.195700                0.066120  ...       29.720000
max         0.304000                0.097440  ...       49.540000

       perimeter_worst  area_worst  smoothness_worst  compactness_worst  \
count       569.000000  569.000000        569.000000         569.000000
mean        107.261213  880.583128          0.132369           0.254265
std          33.602542  569.356993          0.022832           0.157336
min          50.410000  185.200000          0.071170           0.027290
25%          84.110000  515.300000          0.116600           0.147200
50%          97.660000  686.500000          0.131300           0.211900
75%         125.400000 1084.000000          0.146000           0.339100
max         251.200000 4254.000000          0.222600           1.058000

       concavity_worst  concave points_worst  symmetry_worst  \
count       569.000000            569.000000      569.000000
mean          0.272188              0.114606        0.290076
std           0.208624              0.065732        0.061867
min           0.000000              0.000000        0.156500
25%           0.114500              0.064930        0.250400
50%           0.226700              0.099930        0.282200
75%           0.382900              0.161400        0.317900
max           1.252000              0.291000        0.663800

       fractal_dimension_worst  Unnamed: 32
count               569.000000          0.0
mean                  0.083946          NaN
std                   0.018061          NaN
min                   0.055040          NaN
25%                   0.071460          NaN
50%                   0.080040          NaN
75%                   0.092080          NaN
max                   0.207500          NaN

[8 rows x 31 columns]
```

In [4]:
```
df = df.replace({'B':0, 'M':1})
df
```

Out[4]:
```
          diagnosis  radius_mean  texture_mean  perimeter_mean  area_mean  \
id
842302            1        17.99         10.38          122.80     1001.0
842517            1        20.57         17.77          132.90     1326.0
84300903          1        19.69         21.25          130.00     1203.0
84348301          1        11.42         20.38           77.58      386.1
84358402          1        20.29         14.34          135.10     1297.0
...             ...          ...           ...             ...        ...
926424            1        21.56         22.39          142.00     1479.0
926682            1        20.13         28.25          131.20     1261.0
926954            1        16.60         28.08          108.30      858.1
```

```
927241          1       20.60        29.33          140.10     1265.0
92751           0        7.76        24.54           47.92      181.0

          smoothness_mean  compactness_mean  concavity_mean  \
id
842302            0.11840           0.27760         0.30010
842517            0.08474           0.07864         0.08690
84300903          0.10960           0.15990         0.19740
84348301          0.14250           0.28390         0.24140
84358402          0.10030           0.13280         0.19800
...                   ...               ...             ...
926424            0.11100           0.11590         0.24390
926682            0.09780           0.10340         0.14400
926954            0.08455           0.10230         0.09251
927241            0.11780           0.27700         0.35140
92751             0.05263           0.04362         0.00000

          concave points_mean  symmetry_mean  ...  texture_worst  \
id                                            ...
842302                0.14710         0.2419  ...          17.33
842517                0.07017         0.1812  ...          23.41
84300903              0.12790         0.2069  ...          25.53
84348301              0.10520         0.2597  ...          26.50
84358402              0.10430         0.1809  ...          16.67
...                       ...            ...  ...            ...
926424                0.13890         0.1726  ...          26.40
926682                0.09791         0.1752  ...          38.25
926954                0.05302         0.1590  ...          34.12
927241                0.15200         0.2397  ...          39.42
92751                 0.00000         0.1587  ...          30.37

          perimeter_worst  area_worst  smoothness_worst  compactness_worst  \
id
842302             184.60      2019.0           0.16220            0.66560
842517             158.80      1956.0           0.12380            0.18660
84300903           152.50      1709.0           0.14440            0.42450
84348301            98.87       567.7           0.20980            0.86630
84358402           152.20      1575.0           0.13740            0.20500
...                   ...         ...               ...                ...
926424             166.10      2027.0           0.14100            0.21130
926682             155.00      1731.0           0.11660            0.19220
926954             126.70      1124.0           0.11390            0.30940
927241             184.60      1821.0           0.16500            0.86810
92751               59.16       268.6           0.08996            0.06444

          concavity_worst  concave points_worst  symmetry_worst  \
id
842302             0.7119                0.2654          0.4601
842517             0.2416                0.1860          0.2750
```

```
84300903             0.4504                 0.2430          0.3613
84348301             0.6869                 0.2575          0.6638
84358402             0.4000                 0.1625          0.2364
...                     ...                    ...             ...
926424               0.4107                 0.2216          0.2060
926682               0.3215                 0.1628          0.2572
926954               0.3403                 0.1418          0.2218
927241               0.9387                 0.2650          0.4087
92751                0.0000                 0.0000          0.2871

          fractal_dimension_worst  Unnamed: 32
id
842302                    0.11890          NaN
842517                    0.08902          NaN
84300903                  0.08758          NaN
84348301                  0.17300          NaN
84358402                  0.07678          NaN
...                           ...          ...
926424                    0.07115          NaN
926682                    0.06637          NaN
926954                    0.07820          NaN
927241                    0.12400          NaN
92751                     0.07039          NaN

[569 rows x 32 columns]
```

## Preparar datos para entrenamiento

In [5]:
```python
from sklearn.model_selection import train_test_split

X = df.drop('diagnosis',axis=1)
X = X.drop('Unnamed: 32',axis=1)
y = df['diagnosis']
# dividir datos
train, test, train_labels, test_labels = train_test_split(X, y,
                                         test_size = 0.33, random_state = 42)
```

In [6]:
```python
train.head()
```

Out[6]:
```
        radius_mean  texture_mean  perimeter_mean  area_mean  smoothness_mean  \
id
87164         15.46         11.89          102.50      736.9          0.12570
905190        12.85         21.37           82.63      514.5          0.07551
857637        19.21         18.57          125.50     1152.0          0.10530
914580        12.47         17.31           80.45      480.1          0.08928
892604        12.46         19.89           80.43      471.3          0.08451

        compactness_mean  concavity_mean  concave points_mean  symmetry_mean  \
id
87164            0.15550         0.20320              0.10970         0.1966
```

```
905190           0.08316          0.06126              0.01867          0.1580
857637           0.12670          0.13230              0.08994          0.1917
914580           0.07630          0.03609              0.02369          0.1526
892604           0.10140          0.06830              0.03099          0.1781

        fractal_dimension_mean  ...  radius_worst  texture_worst  \
id                              ...
87164                  0.07069  ...         18.79          17.04
905190                 0.06114  ...         14.40          27.01
857637                 0.05961  ...         26.14          28.14
914580                 0.06046  ...         14.06          24.34
892604                 0.06249  ...         13.46          23.07

        perimeter_worst  area_worst  smoothness_worst  compactness_worst  \
id
87164            125.00      1102.0           0.15310             0.3583
905190            91.63       645.8           0.09402             0.1936
857637           170.10      2145.0           0.16240             0.3511
914580            92.82       607.3           0.12760             0.2506
892604            88.13       551.3           0.10500             0.2158

        concavity_worst  concave points_worst  symmetry_worst  \
id
87164            0.5830               0.18270          0.3216
905190           0.1838               0.05601          0.2488
857637           0.3879               0.20910          0.3537
914580           0.2028               0.10530          0.3035
892604           0.1904               0.07625          0.2685

        fractal_dimension_worst
id
87164                   0.10100
905190                  0.08151
857637                  0.08294
914580                  0.07661
892604                  0.07764

[5 rows x 30 columns]
```

## Visualizar características con matplotlib

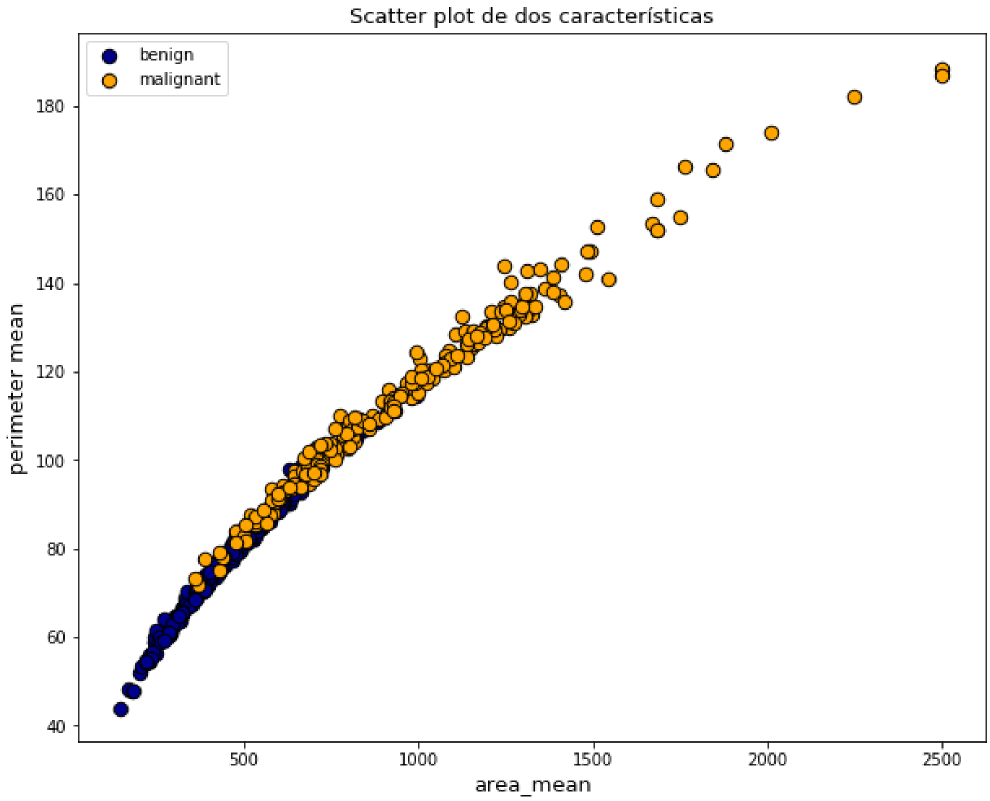
- Obtener etiquetas

In [7]:
```python
import numpy as np

target_ids = np.unique(df.values[:,0])
```

In [8]:
```python
df['area_mean'].values.shape, df['perimeter_mean'].values.shape
```

Out[8]:
```
((569,), (569,))
```

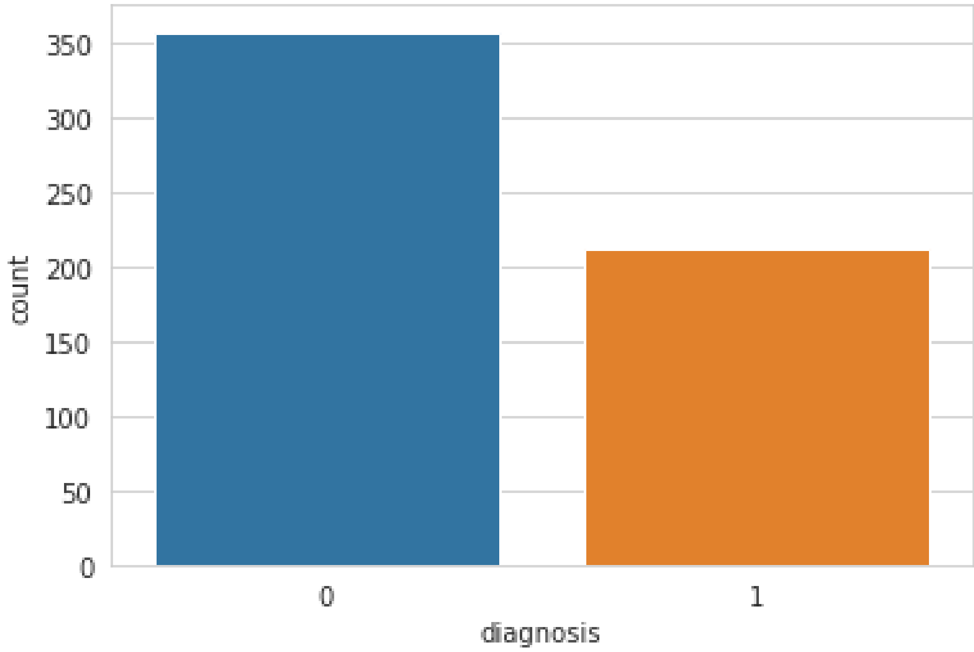
```
In [9]: X_plot = np.concatenate(([df['area_mean'].values], [df['perimeter_mean'].values]), axis=0

In [10]: y = df['diagnosis'].values

In [11]: X_plot.shape

Out[11]: (2, 569)

In [12]: plt.figure(figsize=(10,8))
         colors = ['darkblue','orange']
         target_names = ['benign','malignant']


         for i, c, label in zip(target_ids, colors, target_names):
             plt.scatter(X_plot[0,i == y], X_plot[1,i == y], c = c,  edgecolors='black', s=285,la
         plt.legend()
         plt.title("Scatter plot de dos características",fontsize=13)
         plt.xlabel("area_mean",fontsize=13)
         plt.ylabel("perimeter mean",fontsize=13)
         plt.show()
```




- Equilibrio de etiquetas

```
In [13]: sns.set_style('whitegrid')
         sns.countplot(x='diagnosis',data=df)
```

```
Out[13]: <matplotlib.axes._subplots.AxesSubplot at 0x7f8a2c9c7790>
```




### Evaluación del modelo

- Se obtienen las predicciones, informe de clasificación y matriz de confusión.

```
In [14]: from sklearn.tree import DecisionTreeClassifier
```

```
In [15]: dtc = DecisionTreeClassifier()
         dtc.fit(train, train_labels)
```

```
Out[15]: DecisionTreeClassifier(class_weight=None, criterion='gini', max_depth=None,
                                max_features=None, max_leaf_nodes=None,
                                min_impurity_decrease=0.0, min_impurity_split=None,
                                min_samples_leaf=1, min_samples_split=2,
                                min_weight_fraction_leaf=0.0, presort=False,
                                random_state=None, splitter='best')
```

```
In [16]: predictions = dtc.predict(test)
         print(predictions)
```

```
[0 1 1 0 0 1 1 1 1 1 1 1 0 1 0 1 0 0 0 1 0 0 1 0 0 0 0 0 0 1 0 0 1 0 0 0 1
 0 1 0 0 1 0 0 1 0 1 1 0 0 1 1 0 0 0 0 0 1 1 0 0 1 1 0 0 0 1 1 0 0 1 1 0 1
```

```
 0 0 0 1 0 1 1 0 0 1 1 1 1 1 0 0 0 0 0 0 0 0 1 1 0 1 1 0 1 1 0 0 0 1 1 0 1
 0 0 1 0 1 0 0 0 1 0 0 0 1 0 1 1 0 0 1 1 1 0 0 0 1 1 0 1 1 0 1 0 0 1 0 1 1
 1 0 1 0 0 0 1 1 1 0 0 0 0 0 0 0 1 0 0 0 0 1 0 0 0 1 0 0 1 0 0 0 0 0 0 1 1
 1 0 0]
```

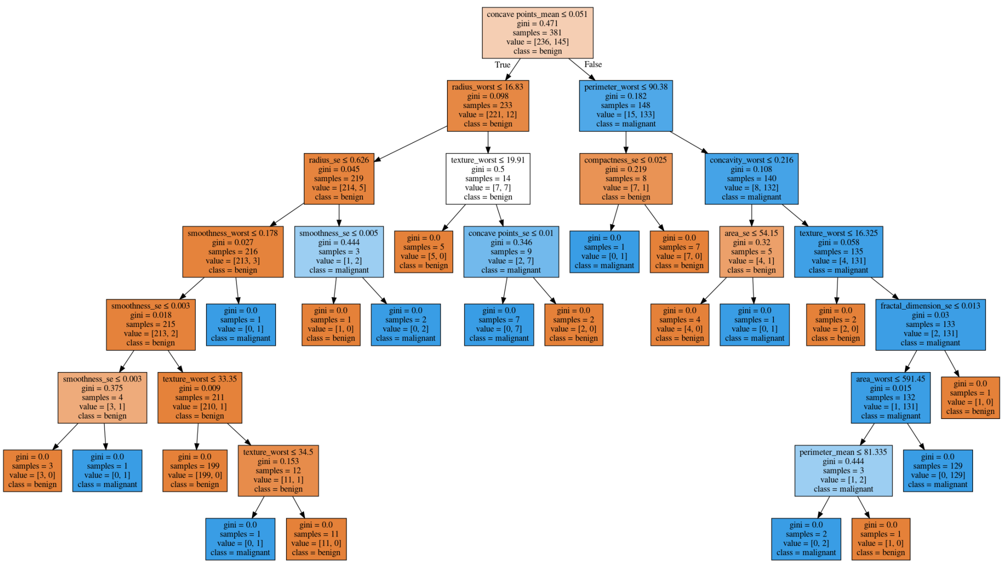
In [17]:
```python
from sklearn.metrics import accuracy_score

print(accuracy_score(test_labels, predictions))
```

```
0.9202127659574468
```

In [18]:
```python
from sklearn.externals.six import StringIO
from IPython.display import Image
from sklearn.tree import export_graphviz
import pydotplus


dot_data = StringIO()
export_graphviz(dtc, out_file=dot_data,
                filled=True, rounded=False,
                special_characters=True,
                feature_names =  X.columns, class_names=target_names)
graph = pydotplus.graph_from_dot_data(dot_data.getvalue())
Image(graph.create_png())
```

```
/home/emam/.conda/envs/tf2/lib/python3.7/site-packages/sklearn/externals/six.py:31: DeprecationWa
  "(https://pypi.org/project/six/).", DeprecationWarning)
```

Out[18]:



- Guardar árbol en .png

In [19]:
```
graph.write_png('cancer.png')
```

Out[19]:
```
True
```

In [20]:
```
from sklearn.metrics import classification_report,confusion_matrix

print("Predicciones:\n")
print(predictions)
print("\nReporte de clasificación:\n")
print(classification_report(predictions,test_labels))
```

```
Predicciones:

[0 1 1 0 0 1 1 1 1 1 1 1 0 1 0 1 0 0 0 1 0 0 1 0 0 0 0 0 0 1 0 0 1 0 0 0 1
 0 1 0 0 1 0 0 1 0 1 1 0 0 1 1 0 0 0 0 0 1 1 0 0 1 1 0 0 0 1 1 0 0 1 1 0 1
 0 0 0 1 0 1 1 0 0 1 1 1 1 1 0 0 0 0 0 0 0 0 1 1 0 1 1 0 1 1 0 0 0 1 1 0 1
 0 0 1 0 1 0 0 0 1 0 0 0 1 0 1 1 0 0 1 1 1 0 0 0 1 1 0 1 1 0 1 0 0 1 0 1 1
 1 0 1 0 0 0 1 1 1 0 0 0 0 0 0 0 1 0 0 0 0 1 0 0 0 1 0 0 1 0 0 0 0 0 0 1 1
 1 0 0]

Reporte de clasificación:

              precision    recall  f1-score   support

           0       0.89      0.98      0.94       110
           1       0.97      0.83      0.90        78

    accuracy                           0.92       188
   macro avg       0.93      0.91      0.92       188
weighted avg       0.92      0.92      0.92       188
```

In [21]:
```
print("Confusion matrix")
conf_mat=confusion_matrix(predictions,test_labels)
print(conf_mat)
```

```
Confusion matrix
[[108   2]
 [ 13  65]]
```

In [22]:
```
from sklearn.metrics import accuracy_score

print(accuracy_score(test_labels, predictions))
```

```
0.9202127659574468
```

- Probar ID3 con otro dataset