Universidad Nacional
Autónoma de México

Inteligencia Artificial

Árboles de Decisión (Pronóstico)

Grupo 3

Nombre:
Barreiro Valdez Alejandro

Práctica 11

Profesor: Dr. Guillermo Gilberto Molero Castillo

5 de mayo de 2022

Introducción

Se utilizarán árboles de decisión para generar un modelo que permita predecir el área de un tumor a partir de estudios clínicos. Además, se probarán diferentes configuraciones para la creación de árboles cambiando diferentes parámetros para generar diferentes modelos. De cada uno de los modelos se dará un análisis que permita saber si el modelo es apto para usarlo. Se utilizarán conceptos vistos en otras prácticas para la selección de variables y la generación de un conjunto de datos de entrenamiento y de prueba.

Objetivos

Pronosticar el área del tumor de pacientes con cancer de mama a través de un árbol de decisión utilizando estudios clínicos a partir de imágenes digitalizadas de pacientes con cáncer de mama de Wisconsin (WDBC, Wisconsin Diagnostic Breast Cancer).

Desarrollo

Se importan las bibliotecas necesarias para la manipulación de datos.

```
import pandas as pd                 # Para la manipulación y análisis de datos
import numpy as np                  # Para crear vectores y matrices n dimensionales
import matplotlib.pyplot as plt     # Para la generación de gráficas a partir de los datos
import seaborn as sns               # Para la visualización de datos basado en matplotlib
%matplotlib inline
```

Se importan los datos del cáncer de mama para utilizarlos.

```
BCancer = pd.read_csv('WDBCOriginal.csv')
BCancer
```

	IDNumber	Diagnosis	Radius	Texture	Perimeter	Area	Smoothness	Compactness	Concavity	ConcavePoints	Symmetry	FractalDimension
0	P-842302	M	17.99	10.38	122.80	1001.0	0.11840	0.27760	0.30010	0.14710	0.2419	0.07871
1	P-842517	M	20.57	17.77	132.90	1326.0	0.08474	0.07864	0.08690	0.07017	0.1812	0.05667
2	P-84300903	M	19.69	21.25	130.00	1203.0	0.10960	0.15990	0.19740	0.12790	0.2069	0.05999
3	P-84348301	M	11.42	20.38	77.58	386.1	0.14250	0.28390	0.24140	0.10520	0.2597	0.09744
4	P-84358402	M	20.29	14.34	135.10	1297.0	0.10030	0.13280	0.19800	0.10430	0.1809	0.05883
...	...	...	...	...	...	...	...	...	...	...	...	...
564	P-926424	M	21.56	22.39	142.00	1479.0	0.11100	0.11590	0.24390	0.13890	0.1726	0.05623
565	P-926682	M	20.13	28.25	131.20	1261.0	0.09780	0.10340	0.14400	0.09791	0.1752	0.05533
566	P-926954	M	16.60	28.08	108.30	858.1	0.08455	0.10230	0.09251	0.05302	0.1590	0.05648
567	P-927241	M	20.60	29.33	140.10	1265.0	0.11780	0.27700	0.35140	0.15200	0.2397	0.07016
568	P-92751	B	7.76	24.54	47.92	181.0	0.05263	0.04362	0.00000	0.00000	0.1587	0.05884

569 rows × 12 columns

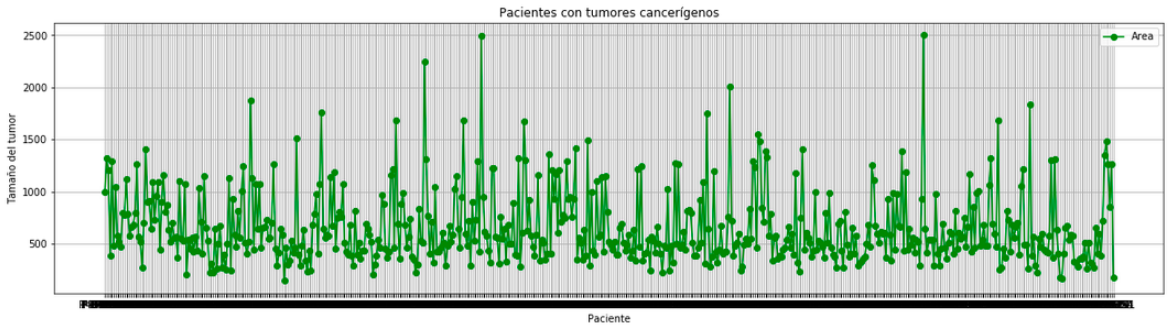
Se muestran datos que analizan la media, los datos totales, el mínimo y máximo entre otras para entender cada una de las variables que tiene el conjunto de datos de los estudios clínicos.

```
BCancer.describe()
```

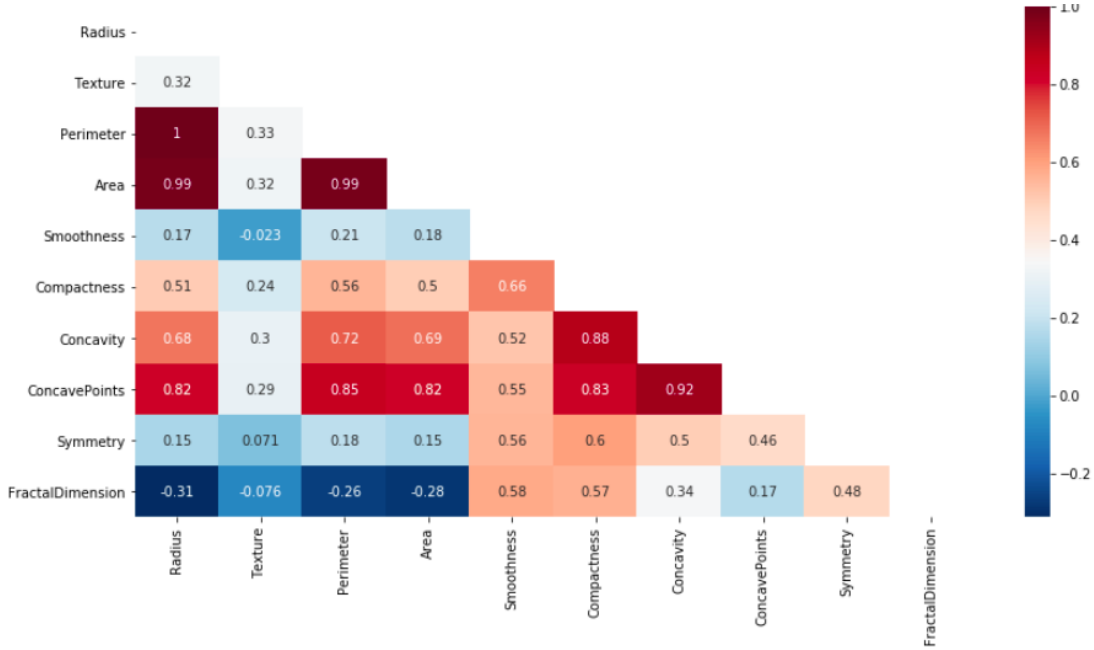
	Radius	Texture	Perimeter	Area	Smoothness	Compactness	Concavity	ConcavePoints	Symmetry	FractalDimension
count	569.000000	569.000000	569.000000	569.000000	569.000000	569.000000	569.000000	569.000000	569.000000	569.000000
mean	14.127292	19.289649	91.969033	654.889104	0.096360	0.104341	0.088799	0.048919	0.181162	0.062798
std	3.524049	4.301036	24.298981	351.914129	0.014064	0.052813	0.079720	0.038803	0.027414	0.007060
min	6.981000	9.710000	43.790000	143.500000	0.052630	0.019380	0.000000	0.000000	0.106000	0.049960
25%	11.700000	16.170000	75.170000	420.300000	0.086370	0.064920	0.029560	0.020310	0.161900	0.057700
50%	13.370000	18.840000	86.240000	551.100000	0.095870	0.092630	0.061540	0.033500	0.179200	0.061540
75%	15.780000	21.800000	104.100000	782.700000	0.105300	0.130400	0.130700	0.074000	0.195700	0.066120
max	28.110000	39.280000	188.500000	2501.000000	0.163400	0.345400	0.426800	0.201200	0.304000	0.097440

Se realizó una gráfica del área de tumor por paciente para visualizar los datos.

```
plt.figure(figsize=(20, 5))
plt.plot(BCancer['IDNumber'], BCancer['Area'], color='green', marker='o', label='Area')
plt.xlabel('Paciente')
plt.ylabel('Tamaño del tumor')
plt.title('Pacientes con tumores cancerígenos')
plt.grid(True)
plt.legend()
plt.show()
```



Se realiza la selección de variables utilizando un mapa de calor y las variables seleccionadas son: textura, área, smoothness, compactness, symmetry, FractalDimension y perímetro.



Se generó un árbol profundo a partir de los datos de entrenamiento utilizando muestras aleatorias de una semilla. Se utilizó la biblioteca de sklearn.

```
PronosticoAD = DecisionTreeRegressor(random_state=0)
PronosticoAD.fit(X_train, Y_train)

#PronosticoAD = DecisionTreeRegressor(max_depth=8, min_samples_split=4, min_samples_leaf=2, random_state=0)
#PronosticoAD.fit(X_train, Y_train)
```

```
DecisionTreeRegressor(ccp_alpha=0.0, criterion='mse', 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='deprecated',
                      random_state=0, splitter='best')
```

Se genera el pronóstico

Se genera un dataframe con los pronósticos que genera el modelo a partir de los datos de prueba. Se juntan los valores de prueba y pronosticados para calcular cuál es el *score*.

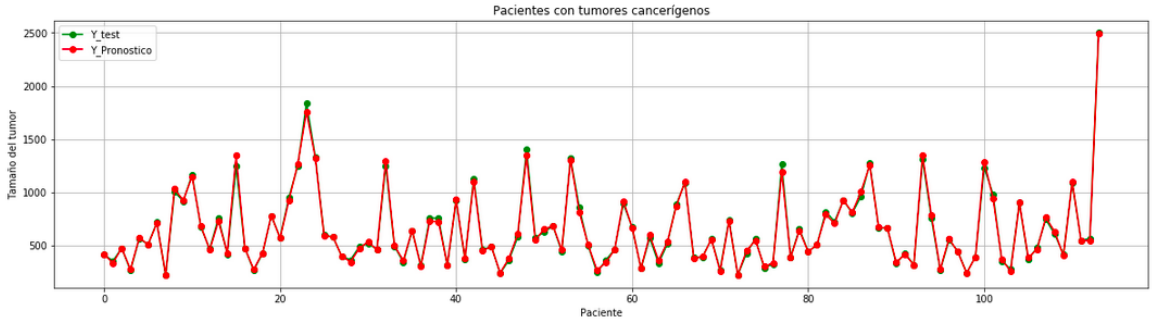
```
Valores = pd.DataFrame(Y_test, Y_Pronostico)
Valores
```

	0
420.5	416.2
337.7	357.6
477.1	476.7
278.6	269.4
575.5	568.9
...	...
408.2	419.8
1102.0	1094.0
546.4	551.7
552.4	565.4
2499.0	2501.0

114 rows × 1 columns

Se graficaron los datos de prueba y los pronósticos para ver qué tan parecidos son los datos del pronóstico del modelo.

```
plt.figure(figsize=(20, 5))
plt.plot(Y_test, color='green', marker='o', label='Y_test')
plt.plot(Y_Pronostico, color='red', marker='o', label='Y_Pronostico')
plt.xlabel('Paciente')
plt.ylabel('Tamaño del tumor')
plt.title('Pacientes con tumores cancerígenos')
plt.grid(True)
plt.legend()
plt.show()
```



Además, se calculó el *score* de los datos pronosticados y los datos de prueba. EL valor fue de .996. Este es un valor muy alto pero cae en sobre ajuste ya que son muy parecidos hasta los valores más lejanos.

```
r2_score(Y_test, Y_Pronostico)
```

```
0.9962328122277272
```

Se obtuvieron los parámetros del modelo para confirmar que existe un sobreajuste.

```
print('Criterio: \n', PronosticoAD.criterion)
print('Importancia variables: \n', PronosticoAD.feature_importances_)
print("MAE: %.4f" % mean_absolute_error(Y_test, Y_Pronostico))
print("MSE: %.4f" % mean_squared_error(Y_test, Y_Pronostico))
print("RMSE: %.4f" % mean_squared_error(Y_test, Y_Pronostico, squared=False))
print('Score: %.4f' % r2_score(Y_test, Y_Pronostico))
```

```
Criterio:
 mse
Importancia variables:
 [8.38144804e-04 9.93535454e-01 4.31438423e-04 1.27530585e-03
 4.65659921e-04 3.45399653e-03]
MAE: 15.2079
MSE: 502.9766
RMSE: 22.4271
Score: 0.9962
```

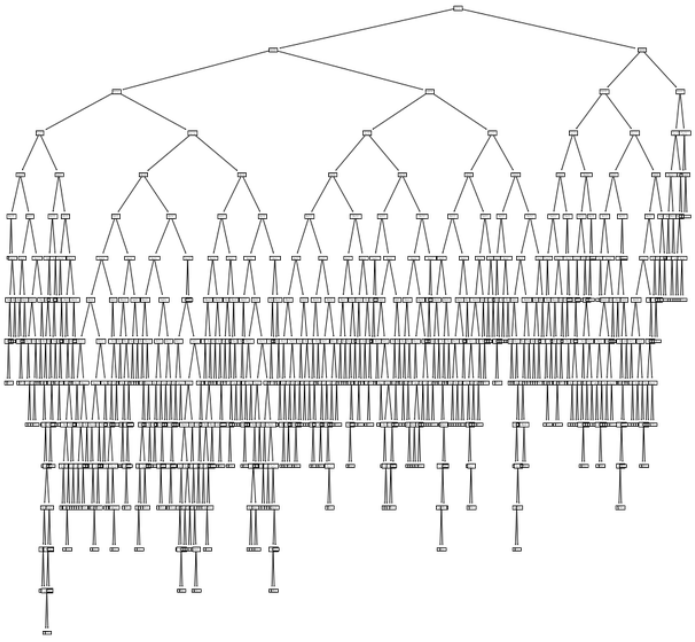
Se generó una tabla de las variables más relevantes para el árbol que se generó. La tabla es la siguiente donde el perímetro el nodo raíz.

```
Importancia = pd.DataFrame({'Variable': list(BCancer[['Texture', 'Perimeter', 'Smoothness',
                                                      'Compactness', 'Symmetry', 'FractalDimension']]),
                            'Importancia': PronosticoAD.feature_importances_}).sort_values('Importancia', ascending=
Importancia
```

	Variable	Importancia
1	Perimeter	0.993535
5	FractalDimension	0.003454
3	Compactness	0.001275
0	Texture	0.000838
4	Symmetry	0.000466
2	Smoothness	0.000431

Se instala graphviz para generar el árbol de decisión que se generó y que se puede utilizar para generar un pronóstico. El árbol generado fue el siguiente pero se necesita general un árbol que genere un modelo más generalizado. El árbol tiene 15 niveles por lo que convendría bajar este número de niveles.

```
from sklearn.tree import plot_tree
plt.figure(figsize=(16,16))
plot_tree(PronosticoAD, feature_names = ['Texture', 'Perimeter', 'Smoothness',
                                         'Compactness', 'Symmetry', 'FractalDimension'])
plt.show()
```



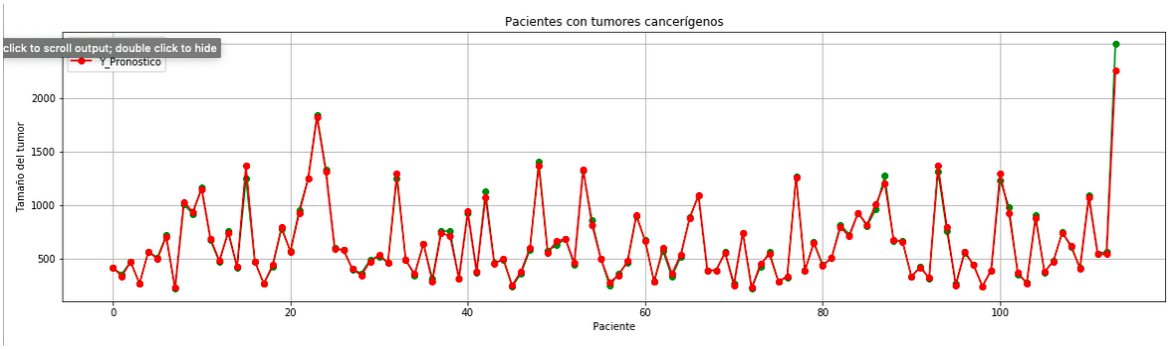
También se pueden generar reglas que ayudan a generar las predicciones. Las reglas son las siguientes.

```
: from sklearn.tree import export_text
  Reporte = export_text(PronosticoAD, feature_names = ['Texture', 'Perimeter', 'Smoothness',
                                                       'Compactness', 'Symmetry', 'FractalDimension'])
  print(Reporte)
```

```
|--- Perimeter <= 110.60
|   |--- Perimeter <= 84.01
|   |   |--- Perimeter <= 70.18
|   |   |   |--- Perimeter <= 60.40
|   |   |   |   |--- Perimeter <= 53.68
|   |   |   |   |   |--- Perimeter <= 45.85
|   |   |   |   |   |   |--- value: [143.50]
|   |   |   |   |   |--- Perimeter >  45.85
|   |   |   |   |   |   |--- Perimeter <= 50.02
|   |   |   |   |   |   |   |--- Symmetry <= 0.20
|   |   |   |   |   |   |   |   |--- Texture <= 25.02
|   |   |   |   |   |   |   |   |   |--- value: [181.00]
|   |   |   |   |   |   |   |   |--- Texture >  25.02
|   |   |   |   |   |   |   |   |   |--- value: [178.80]
|   |   |   |   |   |   |   |--- Symmetry >  0.20
|   |   |   |   |   |   |   |   |--- value: [170.40]
|   |   |   |   |   |--- Perimeter >  50.02
|   |   |   |   |   |   |--- Perimeter <= 52.49
|   |   |   |   |   |   |   |--- value: [201.90]
```

Hay salidas de solo un elemento en las hojas por lo que se sabe que hay un sobreajuste.

Se generó un nuevo modelo con profundidad máxima de ocho. Se redujo el sobreajuste del modelo y a la vez casi no se redujo el *score* que se tiene.




```
r2_score(Y_test, Y_Pronostico)
```

```
0.9918503430404906
```

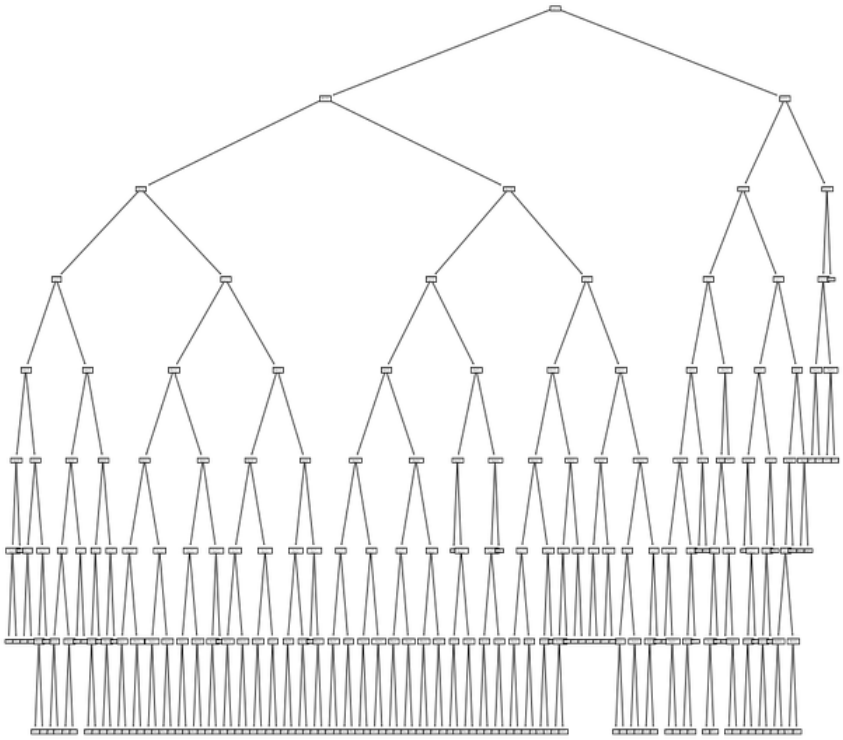
Los parámetros del modelo se calcularon como en el modelo pasado. Los parámetros como la importancia de las variables, MAE, MSE y RMSE fueron diferentes a los del modelo pasado pero el score se mantuvo.

```
print('Criterio: \n', PronosticoAD.criterion)
print('Importancia variables: \n', PronosticoAD.feature_importances_)
print("MAE: %.4f" % mean_absolute_error(Y_test, Y_Pronostico))
print("MSE: %.4f" % mean_squared_error(Y_test, Y_Pronostico))
print("RMSE: %.4f" % mean_squared_error(Y_test, Y_Pronostico, squared=False))
print('Score: %.4f' % r2_score(Y_test, Y_Pronostico))
```

```
Criterio:
 mse
Importancia variables:
 [1.71125637e-04 9.95918034e-01 9.41712055e-04 1.06787154e-03
 1.51010763e-04 1.75024574e-03]
MAE: 18.1236
MSE: 1088.1025
RMSE: 32.9864
Score: 0.9919
```

El árbol del nuevo modelo que se generó se puede observar en la siguiente imagen. Este modelo ya no cae en sobreajuste ya que se redujo la unicidad de los nodos hoja.

```
from sklearn.tree import plot_tree
plt.figure(figsize=(16,16))
plot_tree(PronosticoAD, feature_names = ['Texture', 'Perimeter', 'Smoothness',
                                         'Compactness', 'Symmetry', 'FractalDimension'])
plt.show()
```



Por último, al igual que en prácticas pasadas se puede generar un pronóstico a partir del modelo utilizando el método *predict*. Se generó una predicción del área del tumor a partir de datos específicos.

```
AreaTumorID1 = pd.DataFrame({'Texture': [10.38],
                             'Perimeter': [120.8],
                             'Smoothness': [0.11840],
                             'Compactness': [0.27760],
                             'Symmetry': [0.2419],
                             'FractalDimension': [0.07871]})
PronosticoAD.predict(AreaTumorID1)
```

```
array([991.83333333])
```

Conclusiones

En esta práctica se logró crear un modelo utilizando un árbol de decisión. Este modelo predice el área de un tumor a partir de estudios clínicos. Se crearon dos

árboles de decisión, uno con una semilla aleatoria y el procedimiento que utiliza la biblioteca. Sin embargo, se pudo observar en el árbol que existían hojas que solo contenían un dato. Esto es una señal de que existe un sobreajuste en el modelo que se generó. Para esto se generó un segundo modelo que casi no cambió en su score y que tenía 8 niveles de profundidad. Otras maneras de obtener diferentes configuraciones para el modelo es cambiar las variables a utilizar. Con esta práctica se pudo observar la manera en que se puede tener un sobreajuste y las desventajas de esta condición. El sobreajuste afecta el modelo ya que se tiene un modelo muy específico a los datos que se tienen. Si se introducen datos externos puede que existan sesgos a la tendencia de los datos que se tienen. Por último, se pudo visualizar de buena manera el árbol que se genera graficando las decisiones que se deben tomar y poniendo en texto estas decisiones.