Cervantes Rubí Brandon

Inteligencia Artificial 2022-2, G.3, 01/05/2022
Practica No.11 Pronostico (Árbol de decisión).

Objetivo.

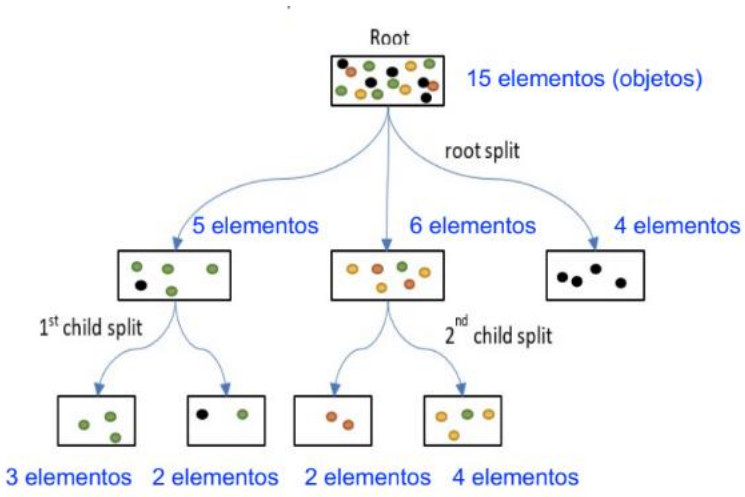
Pronosticar el área del tumor de pacientes con cáncer de mama a través de un árbol de decisión.

Características.

Estudios clínicos a partir de imágenes digitalizadas de pacientes con cáncer de mama de Wisconsin (WDBC, Wisconsin Diagnostic Breast Cancer).

Los árboles de decisión son algoritmos los cuales toman una serie de variables independientes como entrada y las analizan una por una en cada uno de los diferentes niveles del árbol partiendo de una raíz para obtener el determinado valor de una variable dependiente.

En este aspecto tendremos 2 diferentes tipos de árboles de decisión, primero los árboles de clasificación los cuales trabajan con etiquetas como resultados (A, B, C), (0/1), etc. y después están los árboles de regresión, dichos arboles están diseñados para entregar como resultado un valor continuo.



Desarrollo.

Como primer paso tenemos la importación de las librerías necesarias para trabajar con el conjunto de datos de los pacientes a segmentar. `pandas` para la manipulación de los datos, `numpy` para el manejo de matrices, `matplotlib` y `seaborn` para la visualización de estos datos mediante gráficos de diferente tipo dependiendo del análisis.

Después tenemos que hacer la lectura de nuestros datos los cuales están relacionados a las características de los tumores presentados en cada uno de las pacientes relacionadas como su área, textura y la etiqueta la cual nos indica si se trata de un tumor maligno o benigno que nos ayudara más adelante para el proceso de entrenamiento del modelo.

Cervantes Rubí Brandon

Inteligencia Artificial 2022-2, G.3, 01/05/2022
Practica No.11 Pronostico (Árbol de decisión).

Tenemos que leer los datos con ayuda de pandas el cual nos genera un `dataframe` con 569 registros y 12 columnas asociadas a las características de los tumores presentados.

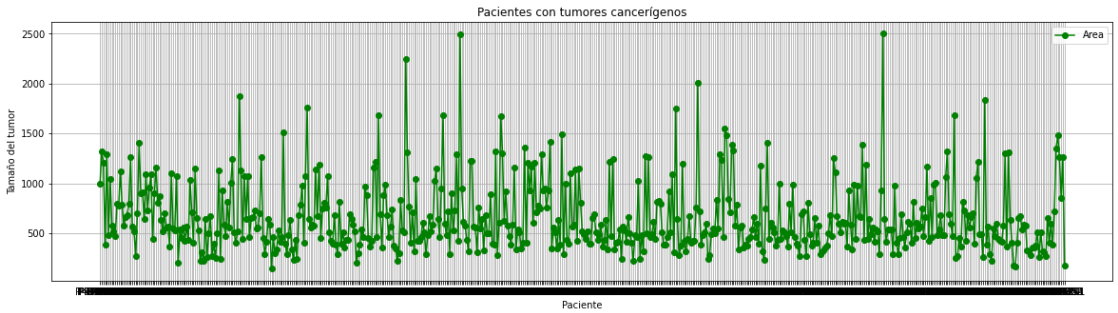
A continuación, se muestra una parte del conjunto de datos leído

	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

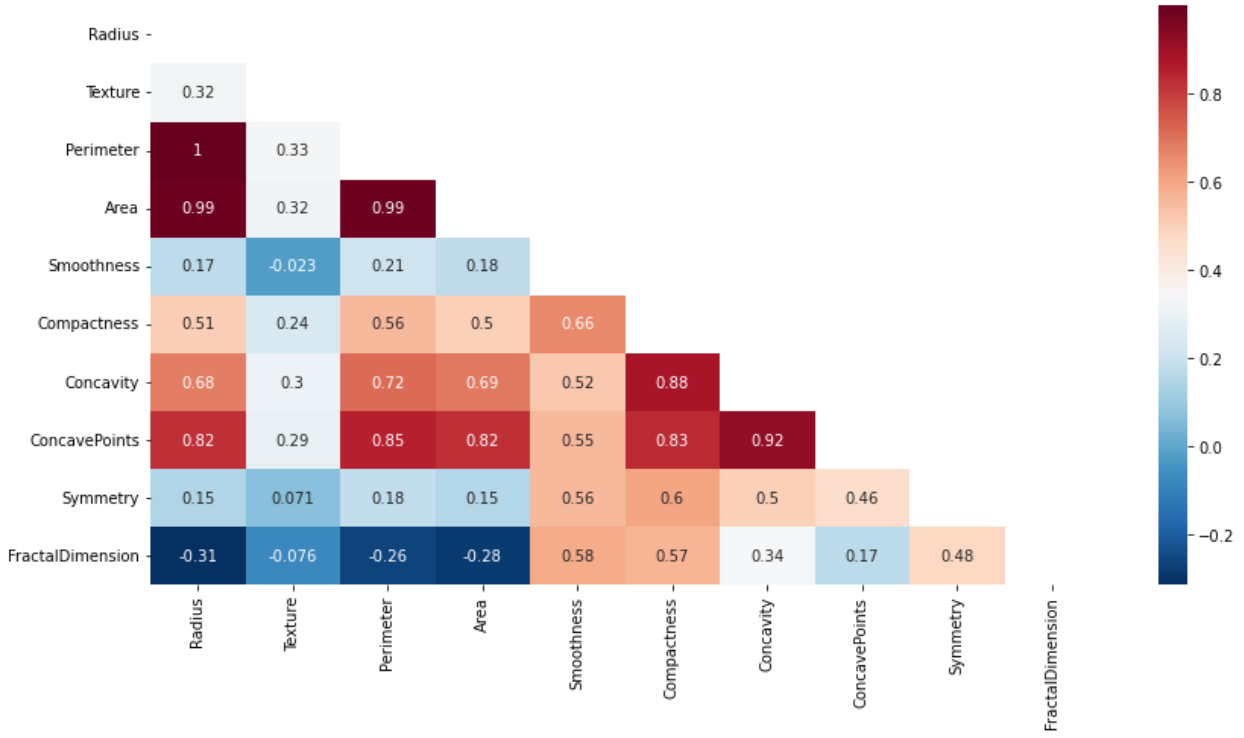
Podemos además hacer una breve descripción de las características mas importantes de los datos leídos estadísticamente hablando. Con el método `describe` del dataframe podremos obtener el conteo de valores no nulos por cada registro asociado a cada variable, además de la media, la desviación estándar, valor mínimo, valor máximo y los cuantiles (25%, 50%, y 75%) de estos valores podemos tomar al 50% como la mediana del conjunto de datos.

	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

Seguido de esto se graficó la relación que existe entre cada uno de los pacientes y el tamaño de sus tumores teniendo rangos desde por debajo de las 500 unidades y como máximo llegando a las 2500.



Considerando las 12 variables en el modelo se decidió hacer un análisis para la selección de características, realizando un análisis de correlación con la matriz de correlaciones se trazando un mapa de calor para hacer el resultado más legible pues este mapa nos muestra con un índice de -1 a 1 el nivel de correlación que tiene cada variable con las demás. A continuación, se muestra el mapa de calor obtenido resaltando las relaciones fuertes entre área y perímetro, radio y área, Concavity con la mayoría de las variables, Concave points de igual manera con la mayoría de las variables por lo cual se decidió que se eliminarían del modelo.



Al final del análisis de correlación las variables independientes que se seleccionaron fueron las siguientes.

1.- Texture.

2.- Area.

3.- Smoothness.

4.- Compactness.

5.- Symmetry.

6.- FractalDimension.

Con las variables independientes del modelo seleccionadas se procedió con la aplicación del algoritmo, Para ello fue necesaria la clase `DecisionTreeRegressor` de `sklearn`, también se ocuparon `mean_squared_error`, `mean_absolute_error`, `r2_score` para el calculo de los errores del modelo y por último `model_selection` para la partición del conjunto de entrenamiento y conjunto de pruebas.

Se hizo un nuevo arreglo de `numpy` con las variables seleccionadas en el anterior análisis de correlación al cual se le llamo X, además se creo otro arreglo llamado Y que contenía los valores de la variable a predecir `Area`. Con estos dos arreglos se utilizo `model_selection` para separar los 2 conjuntos en 4, dos correspondientes a las `X` y `Y` de entrenamiento que representan el 80% del total los datos (455 registros) y otros 2 correspondientes a las `X` y `Y` del conjunto de pruebas que representan el otro 20% (114 registros) del conjunto de datos., por último para utilizar esta librería fue necesario establecer una semilla de aleatoriedad y el atributo `shuffle` en True para que mezcle los registros evitando tomarlos tal y como están.

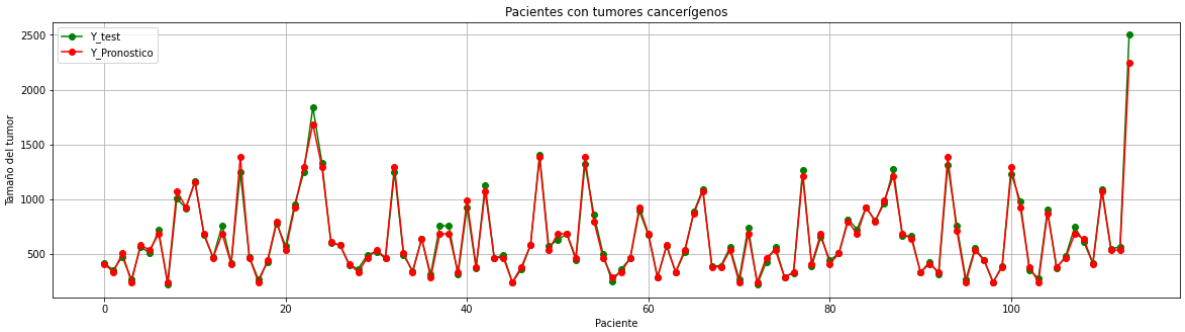
```
X_train, X_test, Y_train, Y_test = model_selection.train_test_split(X, Y,
                                                                    test_size = 0.2,
                                                                    random_state = 1234,
                                                                    shuffle = True
```

Con los conjuntos de datos establecidos, para generar el árbol es necesario utilizar la clase `DecisionTreeRegressor` para crear un objeto a partir de ella sobre la que es posible definir ciertos parámetros del árbol. En la práctica se trabajó con `max_deep` podemos establecer la profundidad del árbol, con `min_samples_split` se define un mínimo de hojas que se tienen que generar y con `min_samples_leaf` podemos controlar el numero de registros que definan una hoja de tal forma que los nodos hoja que no cumplan con este mínimo no se consideran para las reglas que se generen al final. Además, es necesario plantar una semilla de aleatoriedad y `criterion` la cual define la función que se usara para medir la calidad en las divisiones que presenten los nodos intermedios, para los árboles de esta práctica se usó el error absoluto (`absolute_error`).

Una vez que se crea el objeto con los parámetros establecidos, se utiliza el método `fit` pasando como parámetros los conjuntos de datos de entrenamiento que corresponden al 80% de los registros. Para generar las métricas de rendimiento del modelo primero es necesario utilizar el método `predict` pasando como parámetros el conjunto de datos de prueba que corresponde a las variables independientes del modelo `X_test` para generar sus respectivos valores de área predichos los cuales se evaluaran con el conjunto de datos que tiene los valores reales para estos registros, es decir `Y_test`.

```
#MODIFICANDO LOS HIPERPARAMETROS
PronosticoAD = DecisionTreeRegressor(max_depth=5, min_samples_split=4,
min_samples_leaf=2, random_state=0, criterion = "absolute_error")
PronosticoAD.fit(X_train, Y_train)
Y_Pronostico = PronosticoAD.predict(X_test)
Valores = pd.DataFrame(Y_test, Y_Pronostico)
```

A modo de comparación se generó una gráfica donde se pueden ver los valores predichos para el área del tumor en color rojo y los valores reales en verde.



Como podemos observar parece que el modelo tiene un buen desempeño pues prácticamente los valores pronosticados están sobrepuestos con los valores reales teniendo excepciones sobre todo para aquellos registros donde el área pronosticada tiene a ser más grande.

Con la función `r2_score` podemos obtener el valor del `score` del modelo teniendo los siguientes resultados para mínimo 4 elementos en las hojas y mínimo 2 nodos hojas. Con todas las variables que se seleccionaron a partir del análisis de correlación del modelo se obtuvo un 99.53% de score y una altura total de 15 niveles en el árbol el cual es muy alto perdiendo esta característica para generalizar lo mayor posible.

Profundidad	5	6	7	8	9
Score (%)	98.44	98.86	98.85	99.18	99.21

De estos valores podemos observar que si disminuimos la altura del árbol también lo hace el score, en este caso el disminuir el score nos podría ayudar para evitar caer en un sobreajuste ya que con un porcentaje de score en 99% puede que no se generalice lo suficiente.

Con el atributo `feature_importances_` podemos generar una lista con la importancia de cada una de las variables que se consideraron, a partir de esta lista se creo un dataframe ordenado por valor de importancia en el que podemos observar variaciones dependiendo de la configuración.

Para las tres configuraciones presentadas se generaron diferentes arboles partiendo de diferentes variables las cuales se detallan, pare generar las gráficas de los árboles y el conjunto de reglas fue necesario trabajar con los módulos `export_graphviz`, `plot_tree` y `export_text` de `sklearn.tree`.

Cervantes Rubí Brandon

Inteligencia Artificial 2022-2, G.3, 01/05/2022
Practica No.11 Pronostico (Árbol de decisión).

Árbol con todas las variables del modelo.

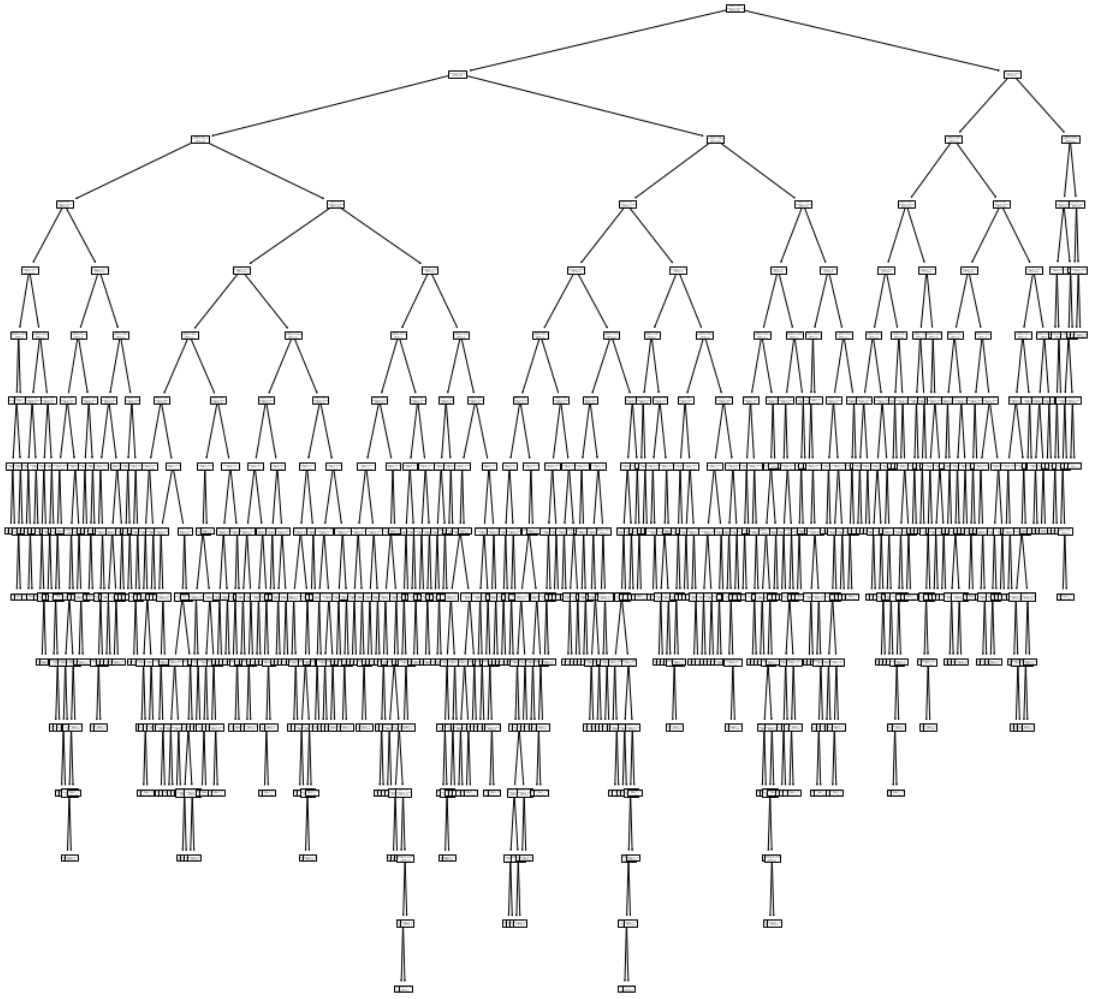
Nodo del que se parte.

```
Radius <= 16.625
squared_error = 121144.947
samples = 455
value = 655.692
```

	Variable	Importancia
0	Radius	0.996370
4	Compactness	0.001719
2	Perimeter	0.001345
1	Texture	0.000187
3	Smoothness	0.000129
7	Symmetry	0.000082
5	Concavity	0.000066
6	ConcavePoints	0.000060
8	FractalDimension	0.000043

El árbol generado presenta 15 niveles y el nodo raíz parte del perímetro con un score del 99.24%.

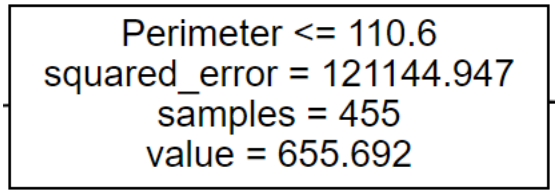
El radio es la variable que presenta un mayor nivel de entropía o que mayor importancia toma para el modelo, por el contrario, la menos importante es Dimensión Fractal.



Cervantes Rubí Brandon

Árbol con las variables seleccionadas del análisis de correlación.

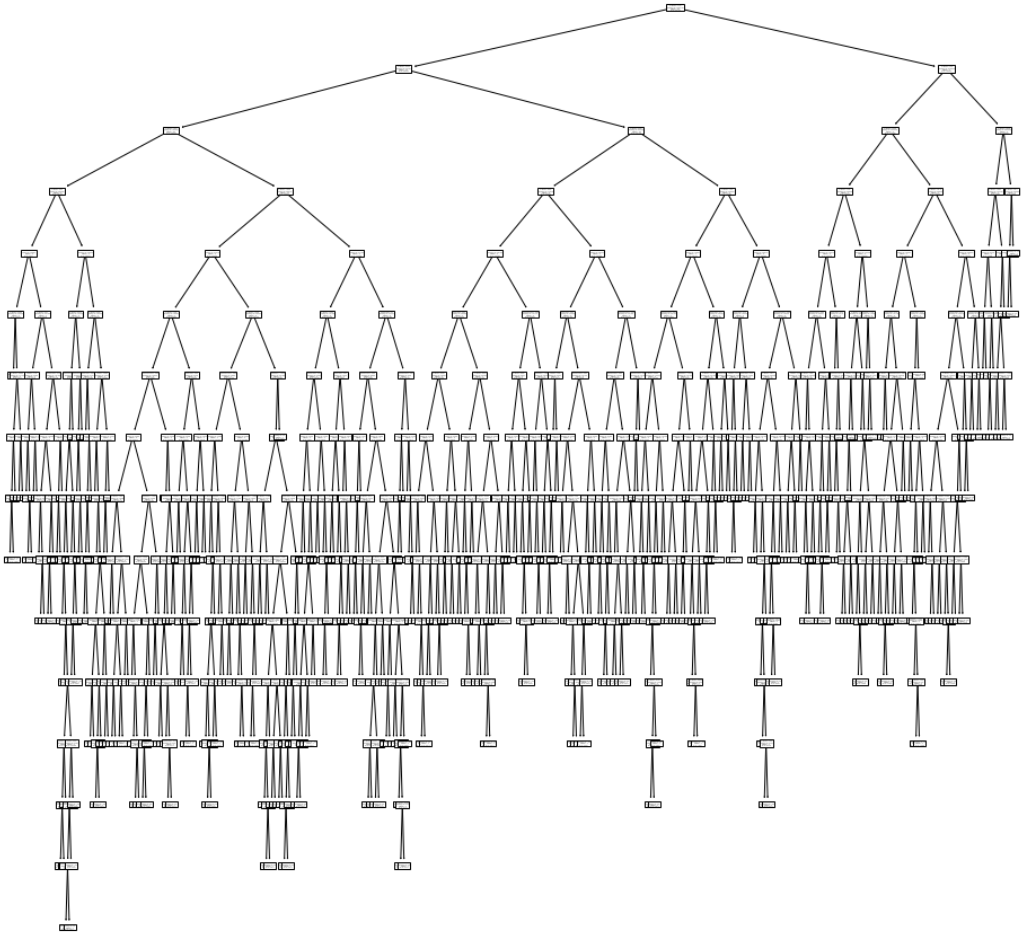
Nodo del que se parte.



	Variable	Importancia
1	Perimeter	0.995231
3	Compactness	0.001731
5	FractalDimension	0.001650
2	Smoothness	0.001000
0	Texture	0.000312
4	Symmetry	0.000076

El árbol generado presenta 15 niveles y el nodo raíz parte del perímetro con un score del 99.54%.

El perímetro es la variable que presenta un mayor nivel de entropía mayor importancia o toma para el modelo, por el contrario, la menos importante es simetría.

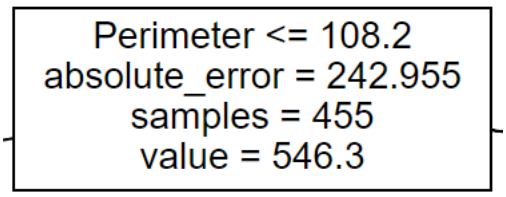


Cervantes Rubí Brandon

Inteligencia Artificial 2022-2, G.3, 01/05/2022
Practica No.11 Pronostico (Árbol de decisión).

Árbol con las variables seleccionadas del análisis de correlación y una profundidad de 5, por lo menos 2 nodos hoja y 4 registros por hoja.

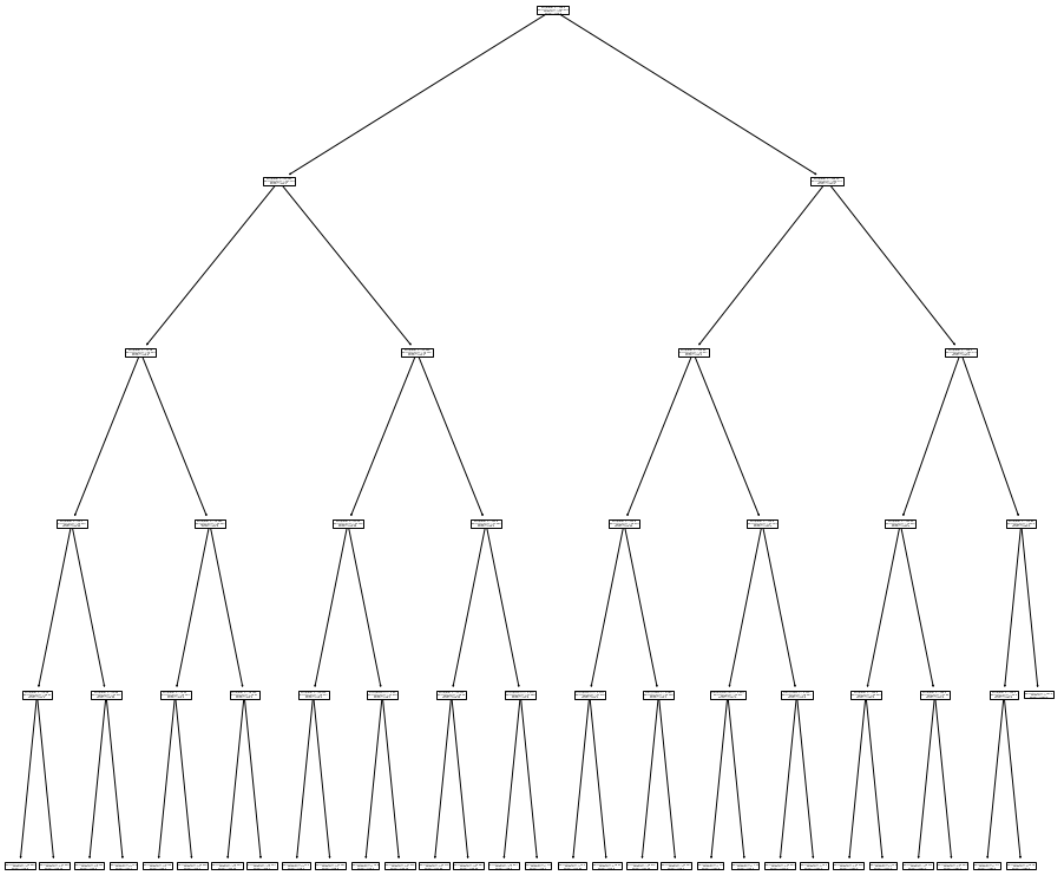
Nodo del que se parte.



	Variable	Importancia
1	Perimeter	0.992116
3	Compactness	0.004108
2	Smoothness	0.002594
5	FractalDimension	0.001182
0	Texture	0.000000
4	Symmetry	0.000000

El árbol generado presenta 5 niveles y el nodo raíz parte del perímetro con un score del 98.44%.

El perímetro es la variable que presenta un mayor nivel de entropía mayor importancia o toma para el modelo, por el contrario, la menos importante es simetría.



Con la función `export_test` podemos generar una lista de reglas las cuales definen la estructura de divisiones que va a tomar el árbol de decisión para hacer los nuevos pronósticos, si la imprimimos podremos visualizar cada uno de los limites que dictan la división en los nodos de la siguiente manera.

```
|--- Perimeter <= 110.60
|   |--- Perimeter <= 84.01
|   |   |--- Perimeter <= 70.18
|   |   |   |--- Perimeter <= 60.40
|   |   |   |   |--- Perimeter <= 53.68
|   |   |   |   |   |--- value: [179.92]
|   |   |   |   |--- Perimeter >  53.68
|   |   |   |   |   |--- value: [246.75]
|   |   |   |--- Perimeter >  60.40
|   |   |   |   |--- Perimeter <= 64.89
|   |   |   |   |   |--- value: [294.61]
|   |   |   |   |--- Perimeter >  64.89
|   |   |   |   |   |--- value: [338.63]
|   |   |--- Perimeter >  70.18
```

Y por último es posible estimar nuevos valores para el área del tumor dando como entrada un nuevo registro con la estructura de un dataframe al método `predict` de nuestro modelo de la siguiente manera, para este registro se obtuvo un área de 1001 con el modelo que considera 5 niveles de profundidad y las variables de la selección de características.

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

Conclusiones.

En esta práctica se trabajaron los árboles de regresión para estimar valores continuos correspondientes al área de un tumor, para ello fue necesario realizar una selección de características basado en las correlaciones de las variables. Una vez generado el árbol tenemos que cuidar las métricas de desempeño de modo que no sean muy bajas para el contexto en el que se este trabajando pero que tampoco sean muy altas para evitar caer en un sobreajuste, en este último caso es posible cambiar el criterio de separación en los nodos o hacer una poda modificando los hiperparametros del árbol para establecer un mínimo de registros por cada hoja y con esto generalizar un poco más el árbol sobre el cual se vayan a trabajar nuevas estimaciones de los registros.

Por ultimo los arboles pueden ser una buena alternativa al algoritmo de regresión lineal múltiple, en este caso se incremento el score en 1% con respecto a practicas anteriores.

Fuentes.

pandas documentation — pandas 1.4.1 documentation. (2022). Pandas. Recuperado 2022, de https://pandas.pydata.org/docs/index.html#

sklearn.tree.DecisionTreeRegressor. (2022). Scikit-Learn. Recuperado 2022, de https://scikit-learn.org/stable/modules/generated/sklearn.tree.DecisionTreeRegressor.html

Dataset. https://archive.ics.uci.edu/ml/datasets/Breast+Cancer+Wisconsin+(Diagnostic)