Universidad Nacional
Autónoma de México

Inteligencia Artificial

Árboles de Decisión (Clasificación)

Grupo 3

Nombre:
Barreiro Valdez Alejandro

Práctica 12

Profesor: Dr. Guillermo Gilberto Molero Castillo

5 de mayo de 2022

Introducción

Se utilizarán árboles de decisión para generar un modelo que permita clasificar tumores malignos y benignos a partir de estudios clínicos. Además, se probarán diferentes configuraciones para la creación de árboles de regresión cambiando algunos parámetros para generar diferentes árboles de decisiones. Se analizará cada uno de los árboles para saber cuál es mejor. Se utilizarán conceptos vistos en prácticas pasadas 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
```

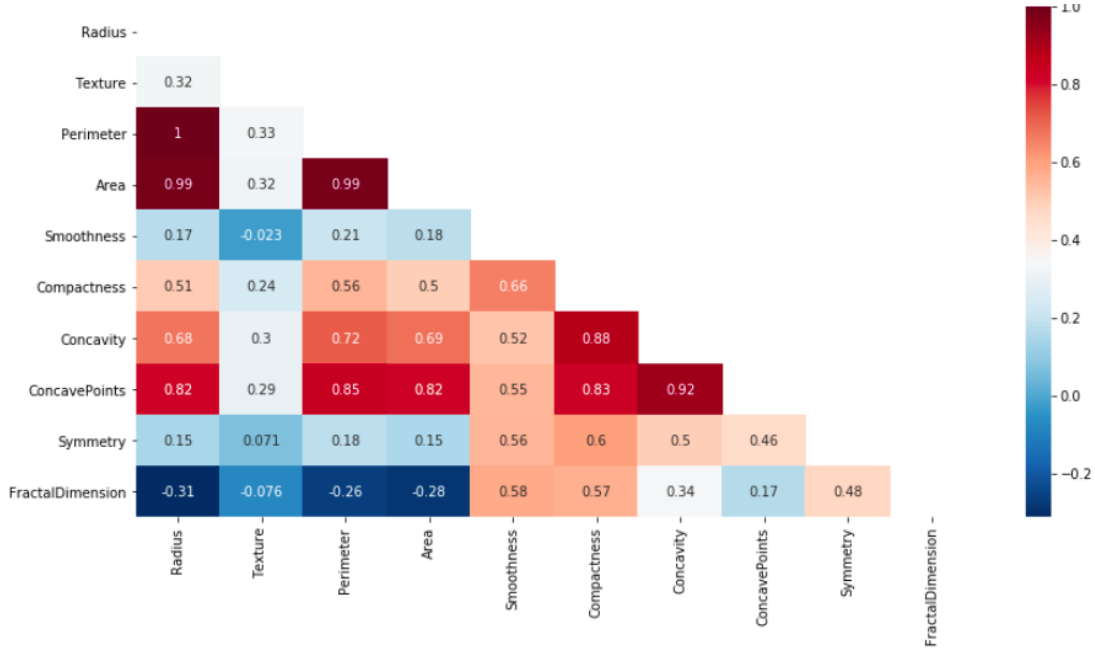
	IDNumber	Diagnosis	Radius	Texture	Perimeter	Area	Smoothness	Compactness	Concavity	ConcavePoints	Symmetry	FractalDimension
0	P-842302	M	17.99	10.38	122.80	1001.0	0.11840	0.27760	0.30010	0.14710	0.2419	0.07871
1	P-842517	M	20.57	17.77	132.90	1326.0	0.08474	0.07864	0.08690	0.07017	0.1812	0.05667
2	P-84300903	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 agruparon los datos para saber cuántos tumores eran malignos y benignos.

```
Diagnosis
B    357
M    212
dtype: int64
```

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 reemplaza el valor de maligno por M y de benigno por B para los datos que se utilizarán.

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

Para esta práctica se generará una predicción de clase utilizando las variables predictoras que se seleccionaron y la variable a predecir será el diagnosis.

```
X = np.array(BCancer[['Texture',
                      'Area',
                      'Smoothness',
                      'Compactness',
                      'Symmetry',
                      'FractalDimension']])
pd.DataFrame(X)

#X = np.array(BCancer[['Radius', 'Texture', 'Perimeter',
#pd.DataFrame(X)
```

	0	1	2	3	4	5
0	10.38	1001.0	0.11840	0.27760	0.2419	0.07871
1	17.77	1326.0	0.08474	0.07864	0.1812	0.05667
2	21.25	1203.0	0.10960	0.15990	0.2069	0.05999
3	20.38	386.1	0.14250	0.28390	0.2597	0.09744
4	14.34	1297.0	0.10030	0.13280	0.1809	0.05883
...	...	...	...	...	...	...
564	22.39	1479.0	0.11100	0.11590	0.1726	0.05623
565	28.25	1261.0	0.09780	0.10340	0.1752	0.05533

```
#Variable clase
Y = np.array(BCancer[['Diagnosis']])
pd.DataFrame(Y)
```

	0
0	Malignant
1	Malignant
2	Malignant
3	Malignant
4	Malignant
...	...
564	Malignant
565	Malignant
566	Malignant
567	Malignant

Se importan nuevas bibliotecas necesarias para las actividades a realizar en esta práctica.

```
from sklearn.tree import DecisionTreeClassifier
from sklearn.metrics import classification_report
from sklearn.metrics import confusion_matrix
from sklearn.metrics import accuracy_score
from sklearn import model_selection
```

Se entrena el modelo a partir de los datos de entrada utilizando la biblioteca de sklearn para generar árboles de decisión clasificadores. Se genera una semilla para los datos aleatorios y el mínimo de los samples de la hoja son 2.

```
#Se entrena el modelo a partir de los datos de entrada
ClasificacionAD = DecisionTreeClassifier(random_state=0, min_samples_leaf=2)
ClasificacionAD.fit(X_train, Y_train)

#ClasificacionAD = DecisionTreeClassifier(max_depth=8, min_samples_split=4, min_samples_leaf=2)
#ClasificacionAD.fit(X_train, Y_train)
```

```
DecisionTreeClassifier(min_samples_leaf=2, random_state=0)
```

A partir del modelo se generó la predicción a partir de los datos de prueba. Además, se calculó el score entre los datos verdaderos y los datos pronosticados.

```
Valores = pd.DataFrame(Y_validation, Y_Clasificacion)
Valores
```

	0
Malignant	Malignant
Benign	Benign
Benign	Benign
Benign	Benign
Benign	Benign
...	...
Benign	Malignant
Benign	Benign
Malignant	Malignant
Malignant	Malignant
Malignant	Benign

114 rows × 1 columns

```
#Se calcula la exactitud promedio de la validación
ClasificacionAD.score(X_validation, Y_validation)
```

```
0.9122807017543859
```

Se generó una tabla de los falsos positivos y negativos para observar cómo funciona el modelo.

```
#Matriz de clasificación
Y_Clasificacion = ClasificacionAD.predict(X_validation)
Matriz_Clasificacion = pd.crosstab(Y_validation.ravel(),
                                   Y_Clasificacion,
                                   rownames=['Real'],
                                   colnames=['Clasificación'])
Matriz_Clasificacion
```

Clasificación	Benign	Malignant
Real		
Benign	62	5
Malignant	5	42

También se genera una tabla para visualizar las variables más significativas para el modelo.

	Variable	Importancia
1	Area	0.701193
3	Compactness	0.170925
0	Texture	0.076840
2	Smoothness	0.041983
4	Symmetry	0.009059
5	FractalDimension	0.000000

Al igual que en la práctica pasada se busca evitar el sobreajuste del modelo ya que existen hojas que tienen mucha especificidad. Se busca tener un modelo más general. Se generó un modelo con máximo de profundidad de 6 y el *score* mejoró a .921.

```
ClasificacionAD = DecisionTreeClassifier(max_depth=6, min_samples_split=4, min_samples_leaf=2)
ClasificacionAD.fit(X_train, Y_train)
```

```
DecisionTreeClassifier(ccp_alpha=0.0, class_weight=None, criterion='gini',
                       max_depth=6, max_features=None, max_leaf_nodes=None,
                       min_impurity_decrease=0.0, min_impurity_split=None,
                       min_samples_leaf=2, min_samples_split=4,
                       min_weight_fraction_leaf=0.0, presort='deprecated',
                       random_state=None, splitter='best')
```

```
#Se calcula la exactitud promedio de la validación
ClasificacionAD.score(X_validation, Y_validation)
```

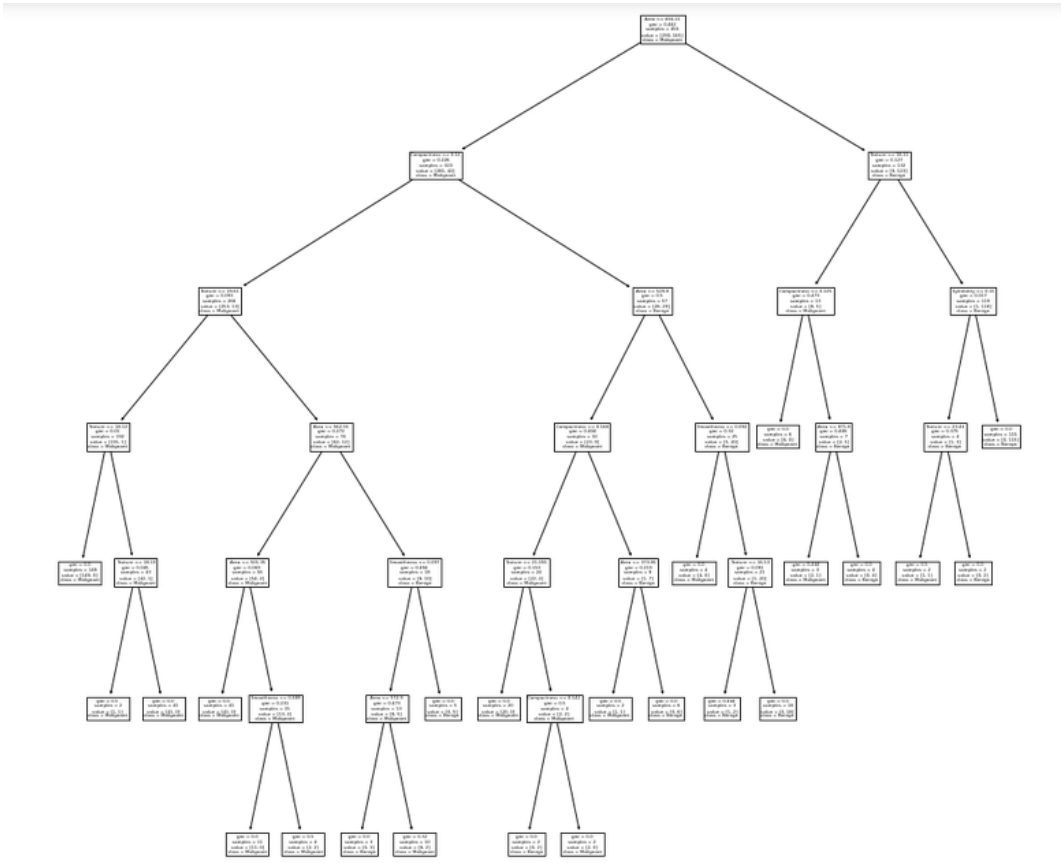
```
0.9210526315789473
```

Al igual que en el modelo pasado se obtuvo una tabla de falsos positivos y falsos negativos. Además se generó una tabla donde se muestra la importancia de cada una de las variables que utiliza el árbol.

Clasificación Real	Benign	Malignant
Benign	62	5
Malignant	4	43

	Variable	Importancia
1	Area	0.706966
3	Compactness	0.163812
0	Texture	0.075177
2	Smoothness	0.051623
4	Symmetry	0.002422
5	FractalDimension	0.000000

A partir de la tabla se tomó la decisión de quitar la variable de fractalDimension, pero el *score* empeoró por lo que se mantuvo el modelo que se generó con profundidad máxima de seis y todos las variables. Gráficamente el modelo que fue generado se puede ver cómo:



Se pueden generar múltiples configuraciones seleccionando diferentes variables y múltiples niveles máximos de profundidad. Como en la práctica pasada se generaron reglas para entender cómo funciona el modelo.

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

```
|--- Area <= 694.15
|   |--- Compactness <= 0.12
|   |   |--- Texture <= 19.61
|   |   |   |--- Texture <= 18.12
|   |   |   |   |--- class: Benign
|   |   |   |--- Texture >  18.12
|   |   |   |   |--- Texture <= 18.19
|   |   |   |   |   |--- class: Benign
|   |   |   |   |--- Texture >  18.19
|   |   |   |   |   |--- class: Benign
|   |   |--- Texture >  19.61
|   |   |   |--- Area <= 562.55
|   |   |   |   |--- Area <= 501.35
|   |   |   |   |   |--- class: Benign
|   |   |   |   |--- Area >  501.35
|   |   |   |   |   |--- Smoothness <= 0.09
|   |   |   |   |   |   |--- class: Benign
|   |   |   |   |   |--- Smoothness >  0.09
|   |   |   |   |   |   |--- class: Benign
|   |   |   |--- Area >  562.55
|   |   |   |   |--- Smoothness <= 0.10
|   |   |   |   |   |--- Area <= 572.90
```

Además, se realizaron predicciones para dos pacientes y se clasificaron para saber si el modelo predice que es benigno o maligno. Para cada uno de los pacientes se tuvieron las siguientes salidas.

```
#Paciente P-842302 (1) -Tumor Maligno-
PacienteID1 = pd.DataFrame({'Texture': [10.38],
                            'Area': [1001.0],
                            'Smoothness': [0.11840],
                            'Compactness': [0.27760],
                            'Symmetry': [0.2419],
                            'FractalDimension': [0.07871]})
ClasificacionAD.predict(PacienteID1)
```

```
array(['Malignant'], dtype=object)
```

```
#Paciente P-92751 (569) -Tumor Benigno-
PacienteID2 = pd.DataFrame({'Texture': [24.54],
                            'Area': [181.0],
                            'Smoothness': [0.05263],
                            'Compactness': [0.04362],
                            'Symmetry': [0.1587],
                            'FractalDimension': [0.05884]})
ClasificacionAD.predict(PacienteID2)
```

```
array(['Benign'], dtype=object)
```

Conclusiones

Se utilizaron árboles de decisión para generar modelos que clasifican un tumor en maligno o benigno utilizando las variables predictoras. Se generaron diferentes configuraciones que permitieron hacer modelos generalizados. Al igual que en la práctica pasada, se generó un modelo utilizando la biblioteca del árbol de decisión y se tuvo un sobreajuste en este caso. No se tuvo un score de .91 lo cual no es tan bueno. Generando un modelo con profundidad máxima de 8 niveles se pudo obtener un árbol con mejor score (.92) y que evita el sobreajuste. Este modelo no obtuvo tan buenos scores por lo que se puede concluir que el modelo anterior es mejor en este ámbito. Sin embargo, no realizan lo mismo, por lo que si se quiere una clasificación convendría utilizar alguno de los modelos antes vistos de clasificación o los bosques aleatorios de clasificación. Se logró analizar los resultados que produce este tipo de modelos y concluir cuál es el mejor y en qué momentos se debe utilizar.