Basile Álvarez Andrés José

No. Cuenta: 316617187

Email: andresbasile123@gmail.com

Fecha: 2/12/21

Inteligencia Artificial

Grupo III

Semestre 2022-1

Reporte Práctica 14: Clasificación (Bosques Aleatorios)

Objetivo: Clasificar registros clínicos de tumores malignos y benignos de cáncer de mama a partir de imágenes digitalizadas

Fuente de datos: Estudios clínicos a partir de imágenes digitalizadas de pacientes con cáncer de mama de Wisconsin (WDBC, Wisconsin Diagnostic Breast Cancer), donde:

- ID number: Identifica al paciente (valor discreto).
- Diagnosis: Diagnóstico (M=maligno, B=benigno).
- Radius: Media de las distancias del centro y puntos del perímetro.
- Texture: Desvaición estándar de la escala de grises.
- Perimeter: Valor del perímetro del cáncer de mama.
- Area: Valor del área del cáncer de mama.
- Smoothness: Variación de la longitud del radio.
- Compactness: Perímetro ^2 / Area -1
- Concavity: Caída o gravedad de las curvas de nivel.
- Concave Points: Número de sectores de contorno cóncavo.
- Symmetry: Simetría de la imagen
- Fractal dimensión: Aproximación de frontera – 1.

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

Características Generales:

En esta práctica, utilizaremos la clasificación por árboles de decisión, utilizando un bosque como una forma de generalización (utilizando varios árboles para mejorar el pronóstico). A esto se le conoce como Bosque Aleatorio, el cual es un algoritmo de aprendizaje automático de gran uso actualmente. Se supone que, al combinar los resultados, los errores se compensan con otros y se tiene una clasificación que generaliza mejor al problema.

Desarrollo

Primeramente, tenemos que definir aquellas bibliotecas de Python que nos serán útiles para importar, limpiar y analizar los datos contenidos en el archivo separado por comas *WDBCOriginal.csv*. Estas serán: *pandas* (manipulación y análisis de datos), *matplotlib* (para la

creación de gráficas y visualización de los datos), *numpy* para utilizar vectores y matrices de *n* dimensiones.

Más tarde, importamos el archivo *WDBCOriginal.csv* y lo primero que hacemos es guardarlo en un DataFrame. La importación del archivo se realizó a partir del explorador de archivos que abrimos utilizando el comando *files.upload()*. Una vez importados los datos, mostramos el DataFrame que los contiene:

```
BCancer = pd.read_csv('WDBCOriginal(1).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

Figura 1: Data Frame con algunos de los datos de tumores de mama.

Una vez hecho lo anterior, se hace una agrupación de los datos:

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

Figura 2: Agrupación de los datos según el diagnóstico.

Una vez hecho lo anterior, utilizamos una matriz de correlaciones con el propósito de seleccionar variables significativas en el conjunto de datos y reducir la dimensionalidad de este. En la figura 3 se muestra la matriz de correlación de Pearson, utilizando colores para distinguir las variables que tienen una mayor dependencia y así poder realizar la selección de variables.

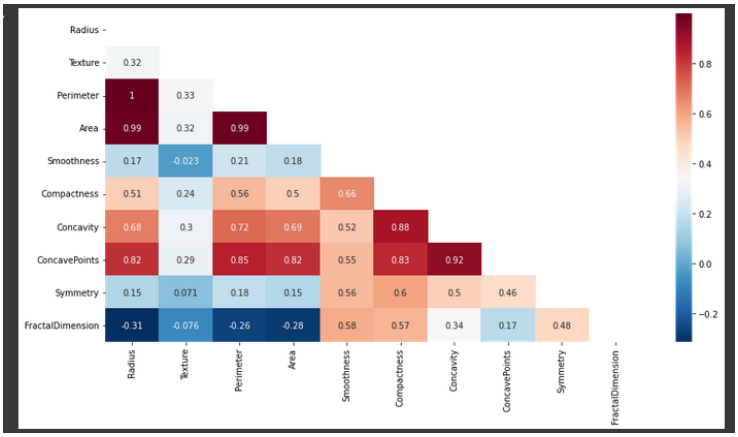


Figura 3: Matriz con las correlaciones entre variables.

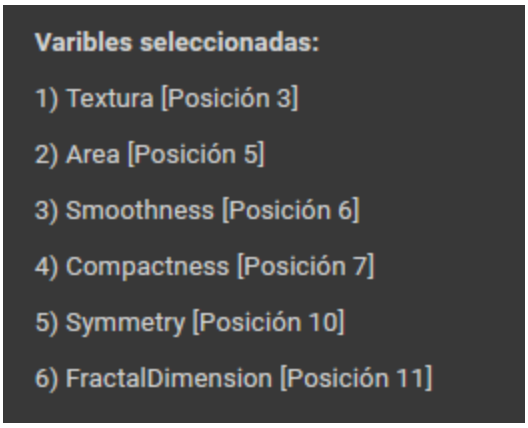
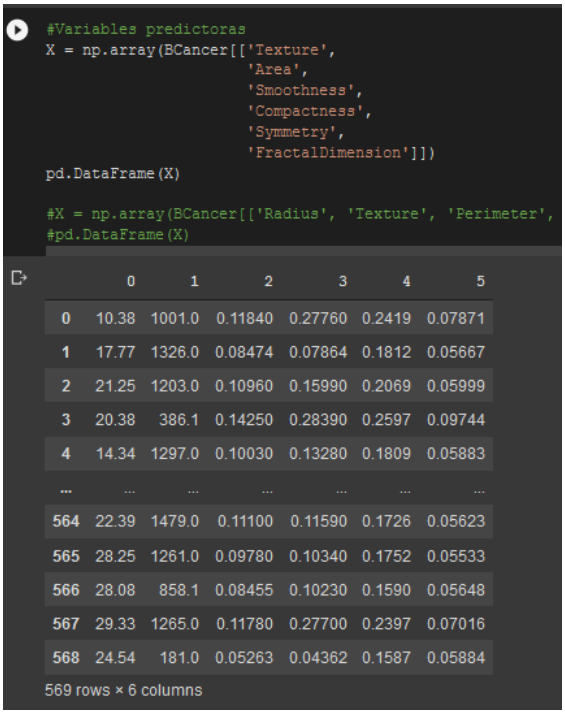


Figura 4: Variables seleccionadas

Definimos entonces un *array* con las variables predictoras que nos permitirán pronosticar el valor del diagnóstico en la clasificación, como se muestra en la figura 5.



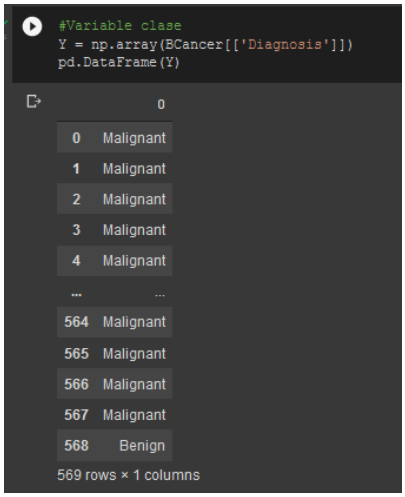
```
#Variables predictoras
X = np.array(BCancer[['Texture',
                      'Area',
                      'Smoothness',
                      'Compactness',
                      'Symmetry',
                      'FractalDimension']])
pd.DataFrame(X)

#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
566	28.08	858.1	0.08455	0.10230	0.1590	0.05648
567	29.33	1265.0	0.11780	0.27700	0.2397	0.07016
568	24.54	181.0	0.05263	0.04362	0.1587	0.05884

569 rows × 6 columns

Figura 5: Parte de la matriz con las variables predictoras.

Además, definimos un vector con la variable clase o variable a clasificar *Diagnosis*, como se muestra en la figura 6.



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

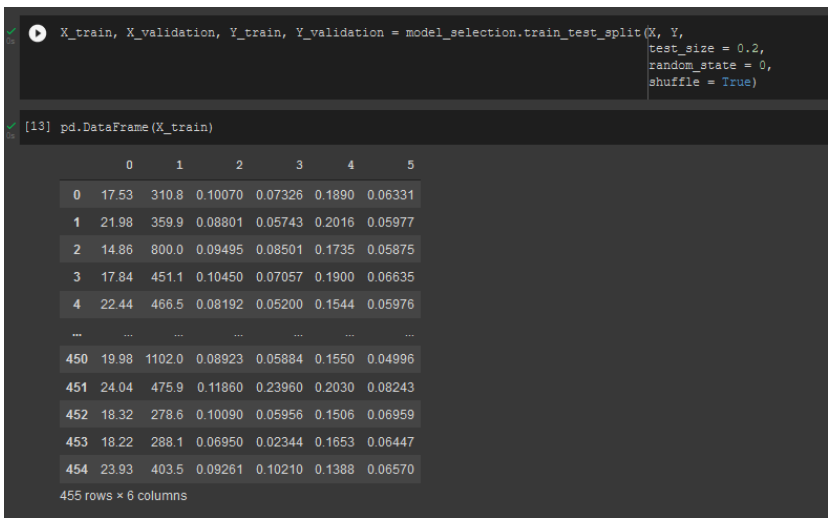
	0
0	Malignant
1	Malignant
2	Malignant
3	Malignant
4	Malignant
...	...
564	Malignant
565	Malignant
566	Malignant
567	Malignant
568	Benign

569 rows × 1 columns

Figura 6: Parte de la matriz con la variable clase.

Para poder aplicar el algoritmo, importamos el *RandomForestClassifier*, las funciones *classification_report*, *confusion_matrix*, *accuracy_score* y *model_selection*.

Realizamos la división de los datos para determinar el tamaño del conjunto de pruebas y del conjunto de entrenamiento, de manera similar a lo que se realizó en prácticas pasadas:

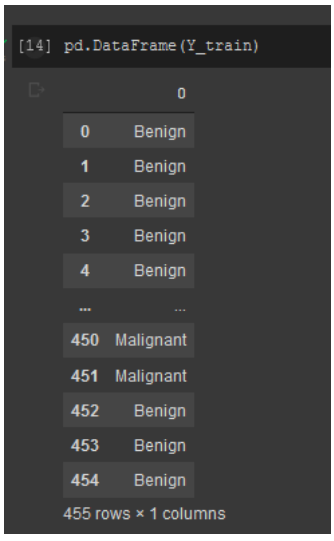


```
X_train, X_validation, Y_train, Y_validation = model_selection.train_test_split(X, Y,
                                                                                test_size = 0.2,
                                                                                random_state = 0,
                                                                                shuffle = True)
```

```
[13] pd.DataFrame(X_train)
```

	0	1	2	3	4	5
0	17.53	310.8	0.10070	0.07326	0.1890	0.06331
1	21.98	359.9	0.08801	0.05743	0.2016	0.05977
2	14.86	800.0	0.09495	0.08501	0.1735	0.05875
3	17.84	451.1	0.10450	0.07057	0.1900	0.06635
4	22.44	466.5	0.08192	0.05200	0.1544	0.05976
...	...	...	...	...	...	...
450	19.98	1102.0	0.08923	0.05884	0.1550	0.04996
451	24.04	475.9	0.11860	0.23960	0.2030	0.08243
452	18.32	278.6	0.10090	0.05956	0.1506	0.06959
453	18.22	288.1	0.06950	0.02344	0.1653	0.06447
454	23.93	403.5	0.09261	0.10210	0.1388	0.06570

455 rows × 6 columns



```
[14] pd.DataFrame(Y_train)
```

	0
0	Benign
1	Benign
2	Benign
3	Benign
4	Benign
...	...
450	Malignant
451	Malignant
452	Benign
453	Benign
454	Benign

455 rows × 1 columns

Figura 7: División de los datos.

Una vez hecho lo anterior, se generan las clasificaciones de bosques aleatorios y comparamos con los valores reales:

```
#Se etiquetan las clasificaciones
Y_Clasificacion = ClasificacionBA.predict(X_validation)
pd.DataFrame(Y_Clasificacion)
```

	0
0	Malignant
1	Benign
2	Benign
3	Benign
4	Benign
...	...
109	Malignant
110	Benign
111	Malignant
112	Malignant
113	Benign

114 rows × 1 columns

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

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

114 rows × 1 columns

Figura 8: Clasificación con árboles aleatorios.

Obtenemos los parámetros del modelo, viendo que tenemos una exactitud de 0.9385. Vemos otros parámetros:

```
Criterio:
 gini
Importancia variables:
 [0.12681465 0.45987493 0.07943492 0.21421778 0.05473754 0.06492018]
Exactitud 0.9385964912280702
              precision    recall  f1-score   support

      Benign       0.94      0.96      0.95        67
   Malignant       0.93      0.91      0.92        47

    accuracy                           0.94       114
   macro avg       0.94      0.94      0.94       114
weighted avg       0.94      0.94      0.94       114
```

Figura 10: Obtención de los parámetros del modelo

Imprimimos, además, la matriz de clasificación, donde vemos que hubo 3 tumores benignos clasificados erróneamente como malignos y 4 tumores malignos clasificados como benignos.

Figura 11: Matriz de clasificación

Clasificación	Benign	Malignant
Real		
Benign	64	3
Malignant	4	43

En la matriz de confusión se utilizó 114 instancias de prueba, clasificándose de manera errónea 7 casos. Esto hace que el modelo tenga un 93.85% de exactitud y un 94% de precisión para los casos Benignos y 93% para los casos malignos. Por otro lado, el error promedio es de 6.15%.

Hecho lo anterior, importamos *graphviz*, un visualizador de árboles. Con esta biblioteca, mostramos el árbol de manera gráfica (uno de los árboles dentro del bosque aleatorio, en este caso el árbol con el índice 10):

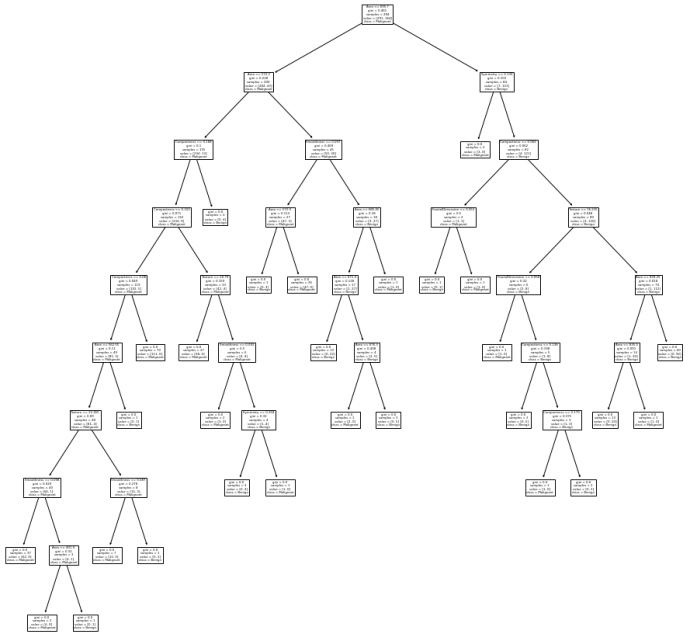


Figura 12: Árbol generado a partir de nuestro modelo de clasificación para bosques aleatorios, en este caso el árbol con índice 10 dentro del bosque.

Más tarde, importamos *export_text* para visualizar el árbol recién creado en forma de texto, observando cada uno de los elementos que lo conforman. Además, como ya contamos con el modelo hecho, podemos ahora hacer nuevas clasificaciones únicamente ingresando los valores de las variables que seleccionamos. De esta forma, por ejemplo, para valores de 'Texture': [10.38], ‘Area’: [1001], 'Smoothness': [0.11840], 'Compactness': [0.27760], 'Symmetry': [0.2419], 'FractalDimension': [0.07871], obtenemos que el modelo clasifica al tumor como maligno.

Como otro ejemplo, para valores de 'Texture': [24.54], ‘Area’: [181], 'Smoothness': [0.526], 'Compactness': [0.043], 'Symmetry': [0.1587], 'FractalDimension': [0.058], obtenemos que el modelo clasifica al tumor como benigno.

Y justamente eso es lo más importante de realizar este tipo de modelos: que su clasificación pueda ser utilizada por un usuario (médico) para ingresar valores de los tumores de los pacientes y obtener una posible clasificación del tumor.

Conclusiones

A lo largo de esta práctica, tuvimos como principal objetivo el obtener clasificaciones del diagnóstico de tumores utilizando bosques aleatorios. Al obtener los parámetros del modelo creado con bosques aleatorios, notamos que hay un mejor desempeño en la clasificación en comparación con la clasificación que puede realizar un árbol de decisión (el cual tuvo efectividad del 91.2%, comparado con el 93.85% obtenido por el bosque aleatorio de esta práctica). Por lo tanto, podríamos decir que el bosque aleatorio fue una mejoría sobre la clasificación que podíamos hacer con un único árbol de decisión.

En general, creo que se cumplieron los objetivos de la práctica y que ésta fue útil para comprender de mejor manera un ejemplo de aprendizaje supervisado sencillo y las posibles aplicaciones de este tipo de algoritmos.