Cervantes Rubí Brandon

Inteligencia Artificial 2022-2, G.3, 16/04/2022
Practica No.10 Clasificación Regresión Logística.

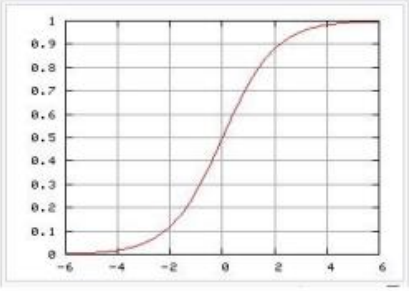
Objetivo.

Clasificar registros clínicos de tumores malignos y benignos de cancer de mama a partir de imágenes digitalizadas.

Características.

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

La regresión logística busca predecir valores binarios los cuales corresponden a la etiqueta de los registros (0/1, verdadero/falso, etc). Lo hace aplicando una transformación a la regresión lineal ya que por sí sola una regresión lineal no nos serviría para predecir esta variable binaria.



Para hacer la transformación se usa la función sigmoide la cual es la siguiente.

$$\hat{Y} = \frac{1}{1 + e^{-(a+b_i x_i)}}$$

Dicha función asigna una probabilidad la cual puede ir de 0 a 1, donde si es mayor a 0.5 asigna el valor de 1 y si es menor o igual a 0.5 entonces asigna el valor de 0.

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.

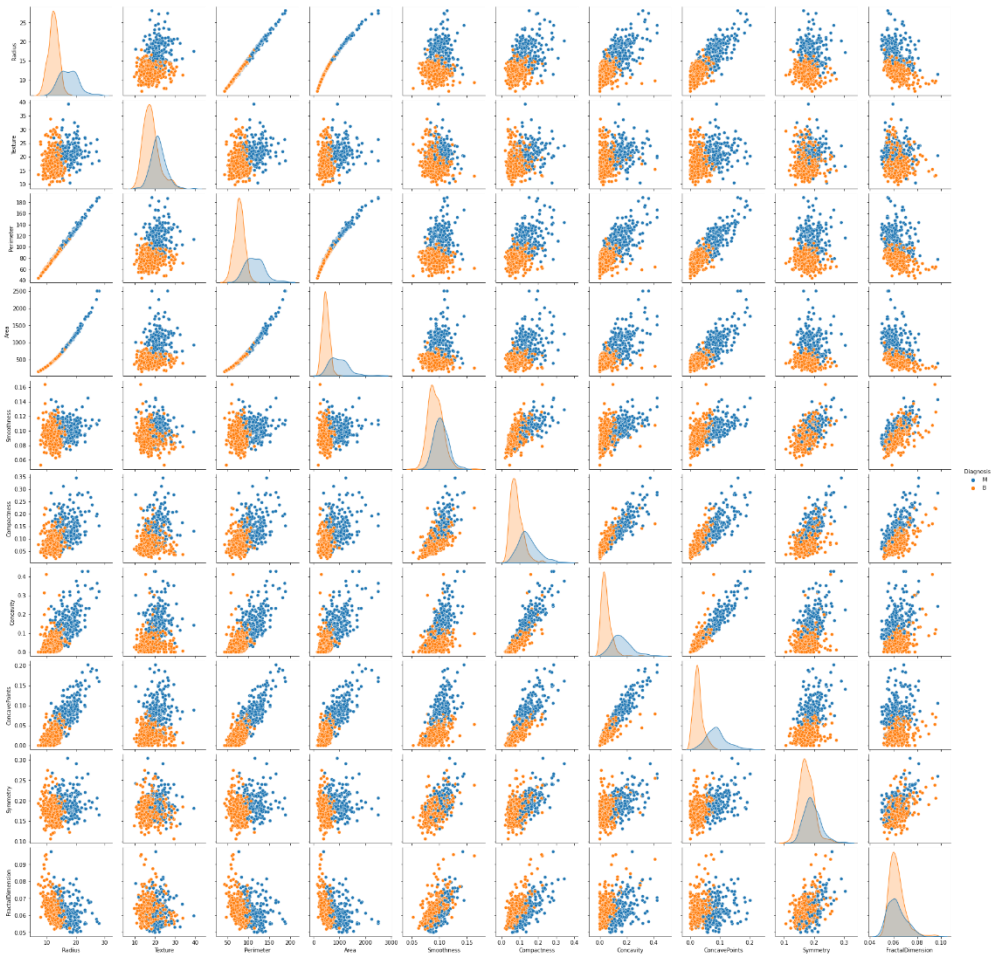
Después tenemos que hacer la lectura de nuestros datos los cuales están relacionados a las características de los tumores presentados en cada uno de los pacientes relacionadas como su área, textura y la etiqueta la cual nos indica si se trata de un tumor maligno o benigno que nos ayudara más adelante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

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.

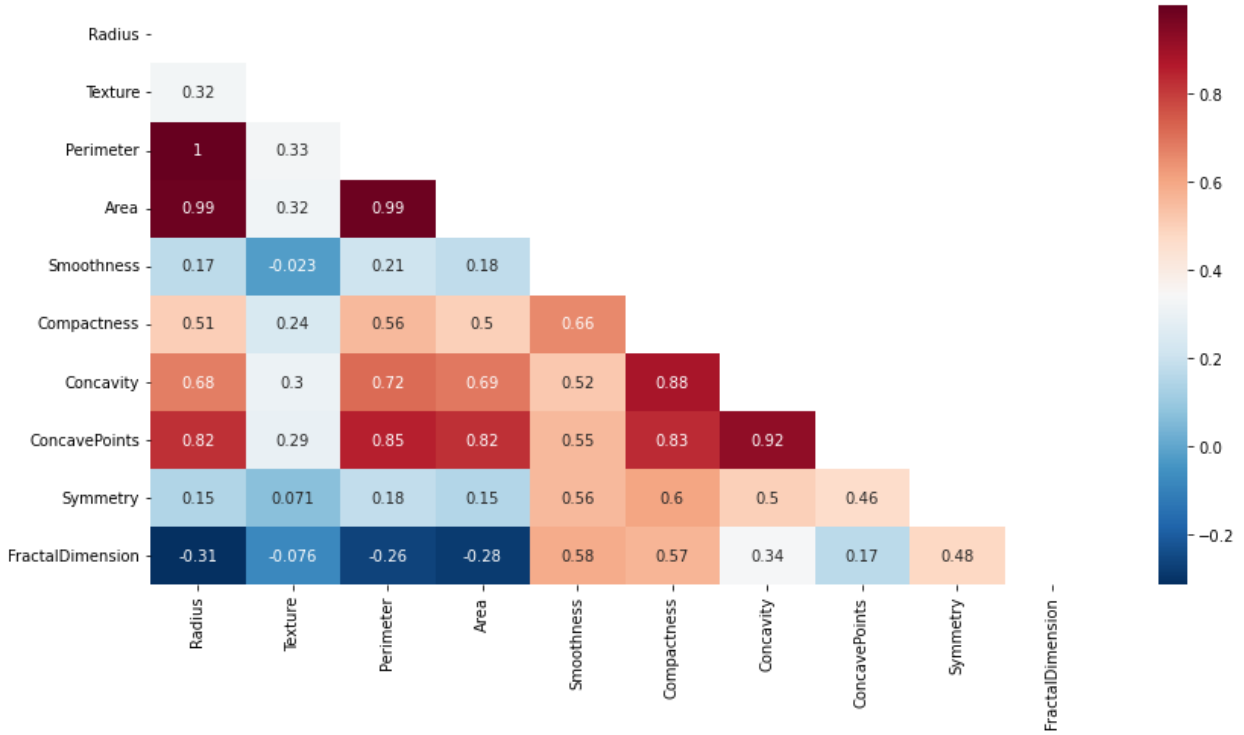
A continuación, se muestra una parte del conjunto de datos leído

	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

Para las 12 variables en el modelo se decidió hacer un análisis para la selección de características, como primer paso se grafico la matriz de dispersión para las variables con el objetivo de visualizar si se tienen variables con algún tipo de relación fuerte para que puedan ser consideradas desechables.



Podemos observar que pudiera existir una relación fuerte entre radio/perímetro, área/radio, y concatyty/concave points por la forma en la que se presentan los puntos. Sin embargo para asegurarnos se realizó un análisis de correlación generando la matriz de correlaciones y 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.

En el caso de la variable dependiente tenemos la clasificación del tumos, maligno o benigno, sin embargo para trabajar con estas etiquetas es necesario que primero se transformen de valores nominales (M: Maligno y B: Benigno) por valores numéricos (0: Maligno y 1: Benigno), esta elección se hace indistintamente de la siguiente manera.

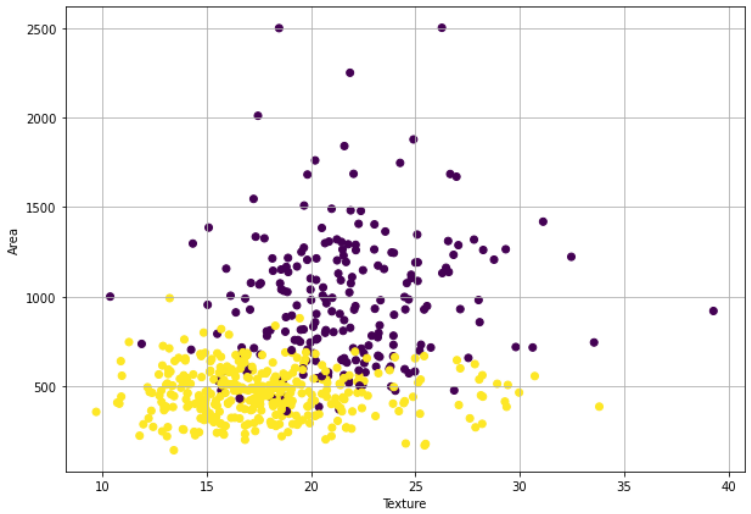
```
BCancer = BCancer.replace({'M': 0, 'B': 1})
BCancer
```

De esta manera, ahora el `dataframe` ahora tiene la siguiente forma:

	IDNumber	Diagnosis	Radius	Texture	Perimeter	Area	Smoothness	Compactness	Concavity	ConcavePoints	Symmetry	FractalDimension
0	P-842302	0	17.99	10.38	122.80	1001.0	0.11840	0.27760	0.30010	0.14710	0.2419	0.07871
1	P-842517	0	20.57	17.77	132.90	1326.0	0.08474	0.07864	0.08690	0.07017	0.1812	0.05667
2	P-84300903	0	19.69	21.25	130.00	1203.0	0.10960	0.15990	0.19740	0.12790	0.2069	0.05999
3	P-84348301	0	11.42	20.38	77.58	386.1	0.14250	0.28390	0.24140	0.10520	0.2597	0.09744
4	P-84358402	0	20.29	14.34	135.10	1297.0	0.10030	0.13280	0.19800	0.10430	0.1809	0.05883

En el mismo podemos hacer una separación y un conteo para conocer el numero de casos con un diagnostico de tumor maligno y benigno ya que es importante mantener un equilibrio. De esta manera se obtuvo que se tienen 212 pacientes con un tumor maligno y 357 con un tumor benigno.

Con los datos preparados ahora tenemos que seleccionar la variable dependiente del modelo (X) la cual será si el tumor es benigno o maligno (0/1) y cuáles serán las variables independientes (Y) para predecir esta etiqueta; Texture, Area, Smoothness, Compactness, Symmetry y FractalDimension. Con esto podemos hacer 2 arreglos de `numpy` a los cuales llamaremos `X` y `Y`.



En la grafica de arriba podemos ver una separación de los datos relacionando su área y su textura, en morado podemos ver a los pacientes con un tumor maligno los cuales tienden a tener una mayor área y en amarillo a aquellos pacientes con un tumor benigno los cuales están en su mayoría alrededor de las 500 unidades de área.

Habiendo creado los arreglos para los conjuntos de las variables dependientes e independientes aún faltaría el separar el conjunto de 559 registros en uno de entrenamiento y otro para las pruebas, para ello se usó la librería de `sklearn model_selection` con su método `train_test_split` el cual recibe como datos el arreglo de `numpy` con las variables independientes del modelo (X), el arreglo de `numpy` con las variables dependientes (Y) o etiquetas, cuál será el tamaño en porcentaje del conjunto de pruebas, una semilla y el atributo `shuffle` el cual mezcla los datos de tal forma que no se tomen según la secuencia dada en un inicio. Como resultado ahora se tienen los 4 conjuntos 2 para entrenamiento y 2 para las pruebas con una proporción de 80% (455 registros) para el entrenamiento y 20% (114 registros) para las pruebas.

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

Ya con cada uno de los conjuntos de datos separados para el modelo podemos comenzar a crearlo, para ello nos ayudaremos de las librerías de `linear_model` para generar el modelo, `mean_squared_error`, `max_error` y `r2_score` los cuales nos van a ayudar a generar las métricas para evaluar la efectividad del modelo una vez lo tengamos.

Después de la separación del conjunto de datos se procede con la creación del modelo generado como un objeto de la clase `linear_model.LogisticRegression` (Al cual llamamos Clasificacion) podemos utilizar el método `fit` pasando como datos los parámetros de los arreglos con los conjuntos de entrenamiento. Después de esto se utilizó el modelo de clasificación generado a partir de estos datos con el método `predict` para generar las etiquetas predichas por el modelo y poder compararlas con las etiquetas reales para asignarle sus métricas de desempeño.

```
Clasificacion = linear_model.LogisticRegression()
Clasificacion.fit(X_train, Y_train)
```

```
Probabilidad = Clasificacion.predict_proba(X_validation)
pd.DataFrame(Probabilidad)

Predicciones = Clasificacion.predict(X_validation)
pd.DataFrame(Predicciones)
```

	0	1
0	0.050099	9.499011e-01
1	0.003135	9.968647e-01
2	0.057000	9.430004e-01
3	0.011637	9.883630e-01
4	0.065728	9.342722e-01

La primera predicción (predict_proba) nos entregara un arreglo con las probabilidades para cada registro de que pertenezca a la etiqueta 0 y a la etiqueta 1.

La segunda de ellas nos entrega un arreglo ordenado con las etiquetas predichas para cada uno de los registros en el conjunto de datos de prueba. Esto con el objetivo de asignar las métricas de desempeño, como primera de ellas podemos obtener el score con el método `score` de nuestro clasificador creado el cual tiene un valor de 93.85% que para el contexto medico con el que se esta trabajando es un valor más que aceptable.

Podemos además generar la matriz de clasificación la cual nos muestra en la diagonal principal cual es el numero de aciertos, todo lo que se encuentre fuera de esta diagonal se suma para contabilizar los errores. Podemos ver que el modelo se equivoco 7 veces de 114 posibles, cometiendo un mayor número de errores en las etiquetas que se predicen como benignos cuando en realidad son malignos.

Clasificación	0	1
Real		
0	39	6
1	1	68

Adicional a esto también es posible generar una matriz la cual resuma el comportamiento de como es que se comporta el modelo de clasificación con los datos de prueba en cada una de las etiquetas.

	precision	recall	f1-score	support
0	0.97	0.87	0.92	45
1	0.92	0.99	0.95	69
accuracy			0.94	114
macro avg	0.95	0.93	0.93	114
weighted avg	0.94	0.94	0.94	114

En ella podemos ver que se comporta mejor clasificando aquellos tumores que son malignos, el `recall` que indica la `sensibilidad` para el caso de las etiquetas positivas (1) y la `especificidad` para los casos negativos (0), entre otras métricas.

Con los atributos `coef_` e `intercept_` del modelo (Clasificación) podemos consultar cuales son los coeficientes de nuestro hiperplano el cual se va a ajustar a la mayoría de los registros siguiendo la siguiente ecuación.

$$\hat{Y} = \frac{1}{1 + e^{-(a+b_i x_i)}}$$

Donde $a + b_i x_i$ equivale a la ecuación de la regresión que se muestra a continuación.

$$a + bX = 12.025 - 0.195(Texture) - 0.011(Area) - 0.707S(moothness)$$
$$- 2.592(Compactness) - 1.025(Symmetry) - 0.257(FractalDimension)$$

Por último, es importante tener en cuenta que con este modelo podemos llegar a hacer extrapolaciones ingresando nuevos valores para cada una de las variables independientes que seleccionamos para el modelo en un inicio, los cuales en este caso serían las características con las que presenta el paciente.

Para probar nuestro modelo ingresamos datos para una textura de 12.38, un area de 1500.0, una suavidad de 0.11, una compacidad de 0.27, una simetría de 0.24 y una dimensión fractal de 0.08 las cuales corresponden al paciente con ID 3 obteniendo un diagnóstico de tumor maligno. Por otro lado también se hizo la prueba con el paciente de ID 2 obteniendo un diagnostico de tumor benigno acertando para ambos casos.

Conclusiones.

En esta práctica se logró generar un modelo de clasificación el cual presenta un diagnostico a los pacientes con un tumor prediciendo si este es maligno o benigno dependiendo de las características que presente las cuales son textura, área, suavidad, compactness, simetría y dimensión fractal. Para ello se implemento un modelo el cual hace una transformación lineal a la regresión lineal que se trabajo previamente en practicas anteriores con la función sigmoide para asignar una probabilidad y crear un limite en 0.5 para que todo lo que este por debajo o igual se etiquete como 0 y lo que este por arriba se etiquete como 1 resultando en una clasificacion binaria.

Podemos además medir el desempeño del modelo evaluándolo con los datos de prueba generando así una matriz de clasificacion la cual muestra en la diagonal principal el conteo de aciertos separados por clase y todo lo demás que la compone son los errores que se cometieron obteniendo un score de 93.85% el cual representa con un porcentaje que tan preciso resulto ser el modelo donde para el contexto medico con el que se trabaja es mas que aceptable y esto lo podemos apreciar si extrapolamos con nuevos valores para generar una clasificación.

Fuentes.

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

User guide: contents. (2022). Scikit-Learn. Recuperado 2022, de https://scikit-learn.org/stable/user_guide.html