Universidad Nacional
Autónoma de México

Inteligencia Artificial

Clasificación (Aplicado en Salud)

Grupo 3

Nombre:
Barreiro Valdez Alejandro

Práctica 10

Profesor: Dr. Guillermo Gilberto Molero Castillo

14 de abril de 2022

Introducción

Se generará un modelo de regresión logística o clasificación para datos de registros clínicos de tumores malignos y benignos de cáncer de mama. Con este modelo se buscará predecir si un tumor será maligno o benigno. También se buscará analizar el modelo de regresión para conocer su exactitud, precisión y cuántos valores de falsos positivos y falsos negativos tiene a partir de los valores reales.

Objetivos

Clasificar registros clínicos de tumores malignos y benignos de cancer de mama a partir de imágenes digitalizadas a partir de estudios clínicos a partir de imágenes digitalizadas de pacientes con cáncer de mama de Wisconsin (WDBC, Wisconsin Diagnostic Breast Cancer).

Desarrollo

Se importan todas las bibliotecas útiles para generar la regresión logística. Se cargan los datos de sobre el cáncer de mama.

```
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
```

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

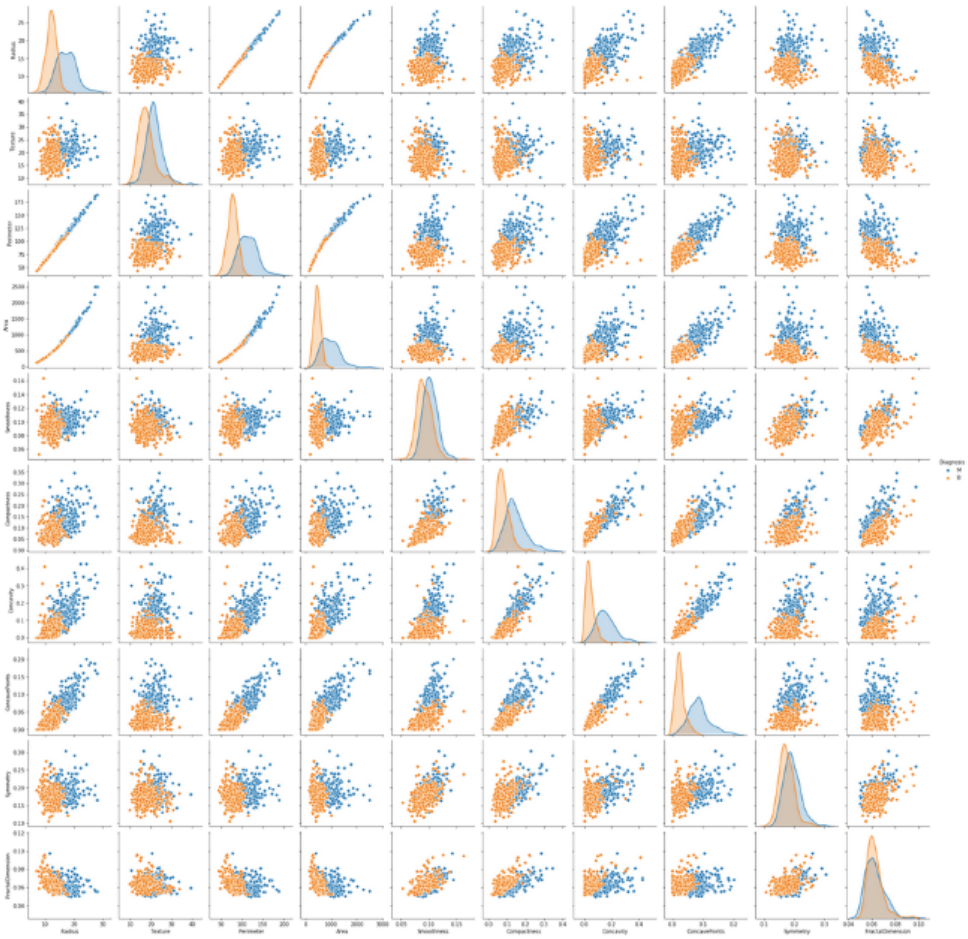
	IDNumber	Diagnosis	Radius	Texture	Perimeter	Area	Smoothness	Compactness	Concavity	ConcavePoints	Symr
0	P-842302	M	17.99	10.38	122.80	1001.0	0.11840	0.27760	0.30010	0.14710	0.
1	P-842517	M	20.57	17.77	132.90	1326.0	0.08474	0.07864	0.08690	0.07017	0.
2	P-84300903	M	19.69	21.25	130.00	1203.0	0.10960	0.15990	0.19740	0.12790	0.
3	P-84348301	M	11.42	20.38	77.58	386.1	0.14250	0.28390	0.24140	0.10520	0.
4	P-84358402	M	20.29	14.34	135.10	1297.0	0.10030	0.13280	0.19800	0.10430	0.
...	...	...	...	...	...	...	...	...	...	...	
564	P-926424	M	21.56	22.39	142.00	1479.0	0.11100	0.11590	0.24390	0.13890	0.
565	P-926682	M	20.13	28.25	131.20	1261.0	0.09780	0.10340	0.14400	0.09791	0.

Se agrupan los datos por diagnosis para saber si el tumor es maligno o benigno y saber cuántos datos de cada uno existen.

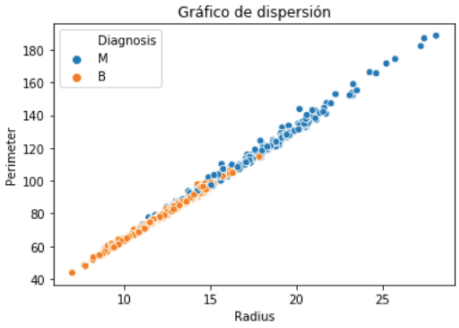
```
print(BCancer.groupby('Diagnosis').size())
```

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

Para la selección de variables se genera un diagrama de las matrices de correlaciones para realizar una evaluación visual.

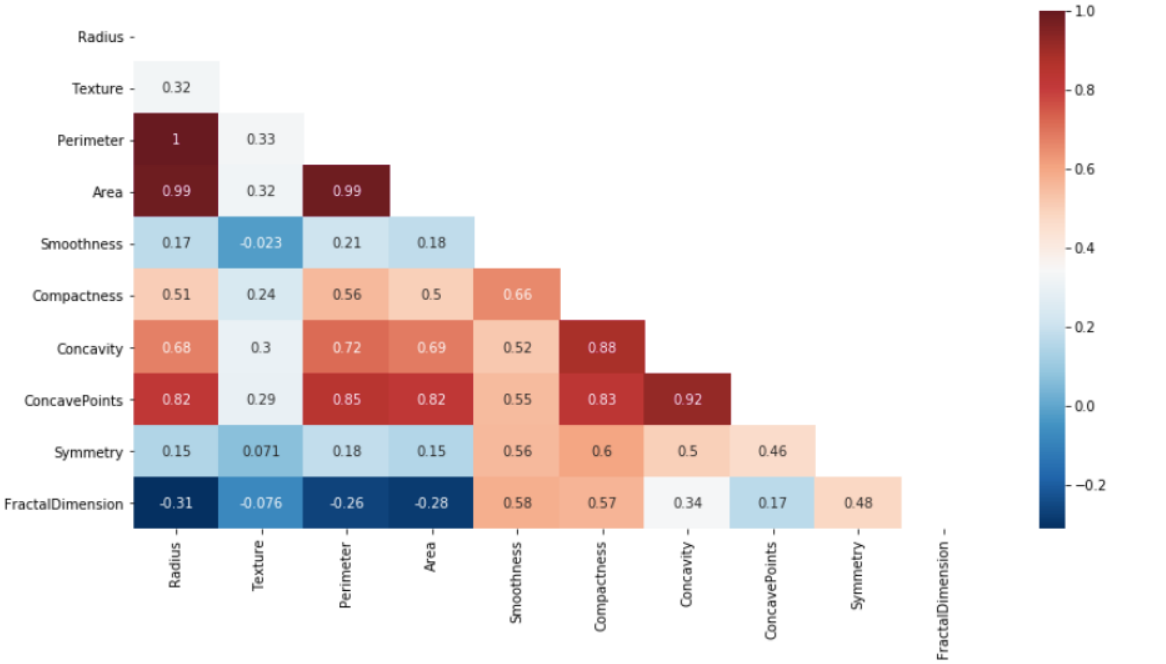


Se muestra de manera individual la correlación que existe entre perímetro y radio. Se observa cómo existe una alta correlación ya que estos valores siguen la misma tendencia.



Se obtiene la correlación de variables utilizando la matriz de correlación por el método de Pearson. La matriz con un mapa de calor se ve de la siguiente manera. A partir de esta imagen se realiza la selección de variables a utilizar para la clasificación.

```
plt.figure(figsize=(14,7))
MatrizInf = np.triu(BCancer.corr())
sns.heatmap(BCancer.corr(), cmap='RdBu_r', annot=True, mask=MatrizInf)
plt.show()
```



Las variables seleccionadas son las siguientes: Textura, Area, Smoothness, Compactness, Symmetry y FractalDimension. Además en los datos originales se reemplaza el valor de maligno por un cero y el valor benigno por un uno. El valor de diagnosis ahora se encuentra con ceros y unos. Estos es útil para el modelo que se creará en los siguientes pasos.

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

	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
...	...	...	...	...	...	...	...	...	...	...	...	...
564	P-926424	0	21.56	22.39	142.00	1479.0	0.11100	0.11590	0.24390	0.13890	0.1726	0.05623
565	P-926682	0	20.13	28.25	131.20	1261.0	0.09780	0.10340	0.14400	0.09791	0.1752	0.05533
566	P-926954	0	16.60	28.08	108.30	858.1	0.08455	0.10230	0.09251	0.05302	0.1590	0.05648
567	P-927241	0	20.60	29.33	140.10	1265.0	0.11780	0.27700	0.35140	0.15200	0.2397	0.07016
568	P-92751	1	7.76	24.54	47.92	181.0	0.05263	0.04362	0.00000	0.00000	0.1587	0.05884

569 rows × 12 columns

A partir de las variables seleccionadas se genera una matriz con las variables predictoras y con el diagnosis se genera la variable de clase.

```
#Variables predictoras
X = np.array(BCancer[['Texture', 'Area', 'Smoothness', 'Compactness', 'Symmetry', 'FractalDimension']])
#X = BCancer.iloc[:, [3, 5, 6, 7, 10, 11]].values  #iloc para seleccionar filas y columnas según su posición
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

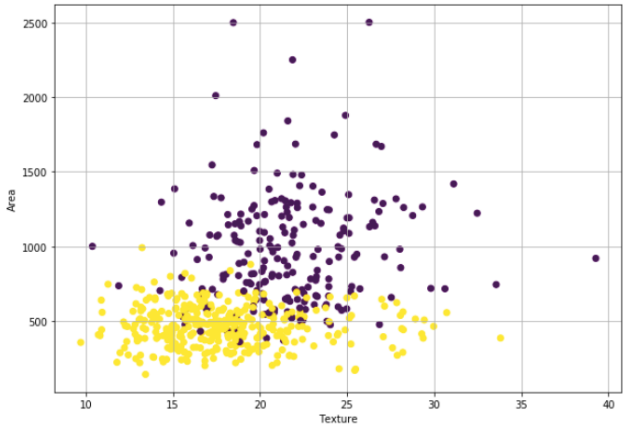
```
#Variable clase
Y = np.array(BCancer[['Diagnosis']])
pd.DataFrame(Y)
```

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

569 rows × 1 columns

Se grafican los datos con su diagnosis utilizando el área y la textura para visualizar la organización de los datos.

```
plt.figure(figsize=(10, 7))
plt.scatter(X[:,0], X[:,1], c = BCancer.Diagnosis)
plt.grid()
plt.xlabel('Texture')
plt.ylabel('Area')
plt.show()
```



Se realiza la aplicación del algoritmo para clasificar. Para esto se importan las bibliotecas necesarias para generar el modelo de regresión logística.

```
#Se importan las bibliotecas necesarias para generar el modelo de regresión logística
from sklearn import linear_model
from sklearn import model_selection
from sklearn.metrics import classification_report
from sklearn.metrics import confusion_matrix
from sklearn.metrics import accuracy_score
```

Al igual que en la práctica pasada se utilizarán datos de prueba y de entrenamiento por lo que se separan los datos en estos grupos para su posterior uso.

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

Con los datos de entrenamiento y el modelo de sklearn se genera una regresión logística.

```
#Se entrena el modelo a partir de los datos de entrada
Clasificacion = linear_model.LogisticRegression()
Clasificacion.fit(X_train, Y_train)
```

Para cada uno de los datos de la agrupación de prueba se genera la probabilidad de que pertenezcan a un grupo u otro.

```
#Predicciones probabilísticas de los datos de prueba
Probabilidad = Clasificacion.predict_proba(X_validation)
pd.DataFrame(Probabilidad)
```

	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
...	...	...
109	0.057255	9.427452e-01
110	0.990748	9.252494e-03
111	0.066344	9.336558e-01
112	0.186568	8.134320e-01
113	1.000000	3.283193e-10

114 rows × 2 columns

Después, se genera una predicción utilizando los datos de las probabilidades y se almacenan dichos datos en un Data Frame. Estos datos servirán para compararlos con los valores reales de prueba.

```
#Predicciones con clasificación final
Predicciones = Clasificacion.predict(X_validation)
pd.DataFrame(Predicciones)
```

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

114 rows × 1 columns

Con estos datos y los de pruebas de la variable a predecir se genera la exactitud promedio de la validicación. Este valor es bastante alto y corresponde al 93% de exactitud para estos valores de validación.

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

```
0.9385964912280702
```

Posteriormente se genera una tabla de los falsos positivos y falsos negativos así como los valores que se encuentran bien clasificados. La tabla generada es la siguiente.

Clasificación	0	1
Real		
0	39	6
1	1	68

Además, se generó un pequeño reporte con los datos que tiene el modelo generado. Se imprime la exactitud del modelo, precisión, recall y soporte de las dos etiquetas con las que se cuentan.

```
#Reporte de la clasificación
print("Exactitud", Clasificacion.score(X_validation, Y_validation))
print(classification_report(Y_validation, Y_Clasificacion))
```

```
Exactitud 0.9385964912280702
              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
```

Por último se generó la ecuación del modelo de clasificación y se predijo si el tumor es benigno o maligno para los datos de un paciente. Para los datos de la ecuación se consultaron los atributos del modelo y a partir de ellos se puede escribir la ecuación.

```
#Ecuación del modelo
print("Intercept:", Clasificacion.intercept_)
print('Coeficientes: \n', Clasificacion.coef_)
```

```
Intercept: [12.0257237]
Coeficientes:
 [[-0.19554751 -0.01115866 -0.70751733 -2.59203115 -1.02579301 -0.25791963]]
```

La ecuación quedaría como:

$$Prob = \frac{1}{1 + e^{-(a+bX)}}$$

$$a + bX =$$

$$12.025 - 0.195Texture - 0.011Area - 0.707Smoothness - 2.592Compactness - 1.025Symmetry - 0.257FractalDimension$$

Se generó una predicción a partir del modelo con los siguientes datos de la variable predictora: Textura: 12.38, Area: 1500.0, Smoothness: 0.11840, Compactness: 0.27760, Symmetry: 0.2419 y Fractal Dimension: 0.07871. La predicción para los valores de dicho paciente fue de un tumor benigno.

```
#Paciente P-842302 (1) -Tumor Maligno-
PacienteID3 = pd.DataFrame({'Texture': [12.38],
                            'Area': [1500.0],
                            'Smoothness': [0.11840],
                            'Compactness': [0.27760],
                            'Symmetry': [0.2419],
                            'FractalDimension': [0.07871]})
Clasificacion.predict(PacienteID3)
```

```
array([0])
```

Conclusiones

Con esta práctica se logró generar un modelo de regresión logística para la clasificación de los registros de tumores malignos y benignos de pacientes con cáncer de mama. Con este modelo y su ecuación se logró predecir la probabilidad para cada uno de los escenarios y se clasificaron los datos utilizando los valores de las variables predictores. Con este modelo también se puede predecir si un tumor es maligno o benigno con los datos de las variables predictores como se realizó en la última parte de la práctica. Además, se analizó la exactitud, precisión y la tabla de falsos positivos y negativos que generó el modelo a partir de sus datos de prueba. Como en la práctica pasada, se utilizaron grupos de datos de entrenamiento y de prueba y se vio la utilidad de ellos. Se pudo visualizar de buena manera el funcionamiento y utilidad de una regresión logística y cómo clasificar a un grupo de datos con este modelo.