Universidad Nacional Autónoma de México

Facultad de Ingeniería

PRÁCTICA 12

PRONÓSTICO CON REGRESIÓN LINEAL MÚLTIPLE

ENFOQUE DE APRENDIZAJE SUPERVISADO

Minería de Datos

Profesor:

Dr. Molero Castillo Guillermo Gilberto

Grupo 1

Alumna:

Monroy Velázquez Alejandra Sarahí

No. Cuenta: 314000417

OBJETIVO

Obtener un pronóstico del área del tumor de mama a través de un modelo de regresión lineal múltiple.

DESARROLLO

El conjunto de datos con el que se trabajará corresponde a estudios clínicos a partir de imágenes digitalizadas de pacientes con cáncer de mama de Wisconsin (WDBC, Wisconsin Diagnostic Breast Cancer):

Variable	Descripción	Tipo
ID number	Identifica al paciente	Discreto
Diagnosis	Diagnostico (M=maligno, B=benigno)	Booleano
Radius	Media de las distancias del centro y puntos del perímetro	Continuo
Texture	Desviación estándar de la escala de grises	Continuo
Perimeter	Valor del perímetro del cáncer de mama	Continuo
Area	Valor del área del cáncer de mama	Continuo
Smoothness	Variación de la longitud del radio	Continuo
Compactness	Perímetro ^ 2 /Area - 1	Continuo
Concavity	Caída o gravedad de las curvas de nivel	Continuo
Concave points	Número de sectores de contorno cóncavo	Continuo
Symmetry	Simetría de la imagen	Continuo
Fractal dimension	"Aproximación de frontera" - 1	Continuo

Primero comenzamos la importación de bibliotecas correspondientes que nos ayudarán para la realización del código, las cuales son *pandas* para la manipulación y análisis de datos, *numpy* para crear vectores y matrices, *matplotlib* para la generación de gráficas, así como *seaborn* para la visualización de datos. Por último, la biblioteca files para subir el archivo *csv*.

```
[1]  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
```

```
[2]  from google.colab import files
     files.upload()
```

```
Elegir archivos  WDBCOriginal.csv
• WDBCOriginal.csv(application/vnd.ms-excel) - 45977 bytes, last modified: 11/11/2021 - 100% done
Saving WDBCOriginal.csv to WDBCOriginal.csv
{'WDBCOriginal.csv': b'\xef\xbb\xbfIDNumber,Diagnosis,Radius,Texture,Perimeter,Area,Smoothne
```

Una vez importadas, el *dataframe* se lee y se despliega en pantalla:

```
[3] 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

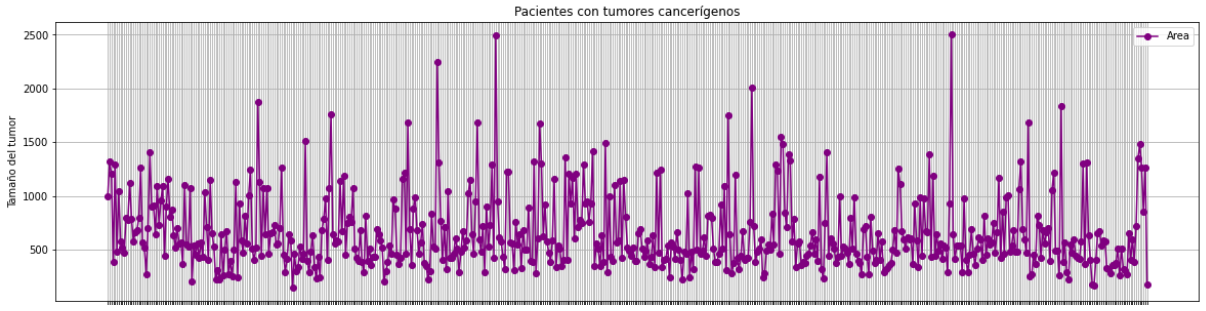
569 rows × 12 columns

Ahora que el *dataframe* está cargado, comenzamos con los pasos vistos en clase.

Evaluación Visual

Graficamos el área, o tamaño del tumor en cada paciente, para visualizar un panorama general de este dato:

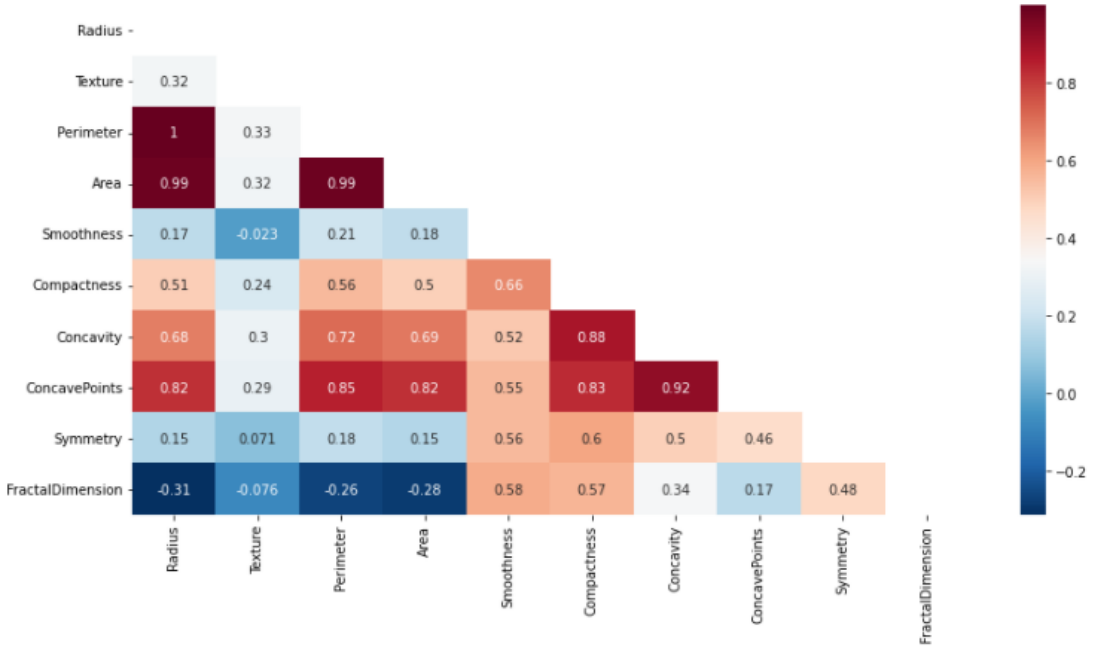
```
[5] plt.figure(figsize=(20, 5))
    plt.plot(BCancer['IDNumber'], BCancer['Area'], color='purple', marker='o', label='Area')
    plt.xlabel('Paciente')
    plt.ylabel('Tamaño del tumor')
    plt.title('Pacientes con tumores cancerígenos')
    plt.grid(True)
    plt.legend()
    plt.show()
```



Selección de Características

Antes de pasar a la aplicación del algoritmo, se realiza una selección de caracteristicas, para descartar aquellas variables poco relevantes, o redundante, en este caso se hara un analisis correlacional de datos. Para ello generamos un mapa de calor mostrando solo la matriz inferior:

```
[6]  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:

1) Textura [Posición 3]

2) Area [Posición 5]

3) Smoothness [Posición 6]

4) Compactness [Posición 7]

5) Symmetry [Posición 10]

6) FractalDimension [Posición 11]

7) Perimeter [Posición 4] - Para calcular el área del tumor

Aplicación del algoritmo

Para comenzar a aplicar el algoritmo necesitamos importar la librería *sklearn*, ya que utilizaremos los módulos *linear_model*, *mean_squared_error*, *max_error*, *r2_score* y *model_selection*.

```
[7]  from sklearn import linear_model
     from sklearn.metrics import mean_squared_error, max_error, r2_score
     from sklearn import model_selection
```

Lo siguiente será seleccionar las variables predictoras (X) y la variable a pronosticar (Y):

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

     #['Radius', 'Texture', 'Perimeter', 'Smoothness', 'Compact
```

	0	1	2	3	4	5
0	10.38	122.80	0.11840	0.27760	0.2419	0.07871
1	17.77	132.90	0.08474	0.07864	0.1812	0.05667
2	21.25	130.00	0.10960	0.15990	0.2069	0.05999
3	20.38	77.58	0.14250	0.28390	0.2597	0.09744
4	14.34	135.10	0.10030	0.13280	0.1809	0.05883
...	...	...	...	...	...	...
564	22.39	142.00	0.11100	0.11590	0.1726	0.05623
565	28.25	131.20	0.09780	0.10340	0.1752	0.05533
566	28.08	108.30	0.08455	0.10230	0.1590	0.05648
567	29.33	140.10	0.11780	0.27700	0.2397	0.07016
568	24.54	47.92	0.05263	0.04362	0.1587	0.05884

569 rows × 6 columns

```
[9]  Y = np.array(BCancer[['Area']])
     pd.DataFrame(Y)
```

	0
0	1001.0
1	1326.0
2	1203.0
3	386.1
4	1297.0
...	...
564	1479.0
565	1261.0
566	858.1
567	1265.0
568	181.0

569 rows × 1 columns

Las variables predictoras son: *Texture*, *Perimeter*, *Smoothness*, *Compactness*, *Symmetry* y *FractalDimension*; mientras que la variable a pronosticar será *Area*.

Se hace la división de los datos, donde se generará el entrenamiento y se despliega tanto la variable *x_train*, como *y_train*:

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

```
[11] pd.DataFrame(X_train)
     #pd.DataFrame(X_test)
```

	0	1	2	3	4	5
0	18.22	84.45	0.12180	0.16610	0.1709	0.07253
1	22.44	71.49	0.09566	0.08194	0.2030	0.06552
2	20.76	82.15	0.09933	0.12090	0.1735	0.07070
3	23.84	82.69	0.11220	0.12620	0.1905	0.06590
4	18.32	66.82	0.08142	0.04462	0.2372	0.05768
...	...	...	...	...	...	...
450	15.18	88.99	0.09516	0.07688	0.2110	0.05853
451	15.10	141.30	0.10010	0.15150	0.1973	0.06183
452	18.60	81.09	0.09965	0.10580	0.1925	0.06373
453	18.70	120.30	0.11480	0.14850	0.2092	0.06310
454	13.78	81.78	0.09667	0.08393	0.1638	0.06100

455 rows × 6 columns

```
[12] pd.DataFrame(Y_train)
     #pd.DataFrame(Y_test)
```

	0
0	493.1
1	378.4
2	480.4
3	499.0
4	340.9
...	...
450	587.4
451	1386.0
452	481.9
453	1033.0
454	492.1

455 rows × 1 columns

Luego, se entrena el modelo a través de Regresión Lineal Múltiple:

```
[13] RLMultiple = linear_model.LinearRegression()
     RLMultiple.fit(X_train, Y_train)

     LinearRegression()
```

Y se genera el pronóstico:

```
[14] #Se genera el pronóstico
     Y_Pronostico = RLMultiple.predict(X_test)
     pd.DataFrame(Y_Pronostico)
```

	0
0	405.607887
1	334.291077
2	505.762398
3	207.726058
4	604.229256
...	...
109	394.439214
110	1107.202694
111	541.131191
112	570.702628
113	2044.635054

114 rows × 1 columns

Obtención de los coeficientes, intercepto, error y score & conformación del modelo de pronóstico

En este paso obtenemos los coeficientes, el intercepto, el error y el score, los cuales nos permitirán realizar el modelo de pronóstico con la siguiente formula:

$$Y = a + b_1X_1 + b_2X_2 \ldots + b_nX_n + u$$

```
[16] print('Coeficientes: \n', RLMultiple.coef_)
     print('Intercepto: \n', RLMultiple.intercept_)
     print("Residuo: %.4f" % max_error(Y_test, Y_Pronostico))
     print("MSE: %.4f" % mean_squared_error(Y_test, Y_Pronostico))
     print("RMSE: %.4f" % mean_squared_error(Y_test, Y_Pronostico, squared=False))
     print('Score (Bondad de ajuste): %.4f' % r2_score(Y_test, Y_Pronostico))
```

```
Coeficientes:
 [[ 6.86261446e-01  1.63885604e+01  2.50787388e+01 -1.40602548e+03
   1.46803422e+02  6.23269303e+03]]
Intercepto:
 [-1140.33616115]
Residuo: 456.3649
MSE: 3083.2634
RMSE: 55.5271
Score (Bondad de ajuste): 0.9769
```

El modelo queda de la siguiente manera:

$$Y = -1140.34 + 0.69(\text{Texture}) + 16.39(\text{Perimeter}) + 25.08(\text{Smoothness}) - 1406.03(\text{Compactness}) + 146.80(\text{Symmetry}) + 6232.69(\text{FractalDimension}) + 456.36$$

- Se tiene un Score de 0.9769, el cual indica que el pronóstico del Area del tumor se logrará con un 97.69% de efectividad.
- Además, los pronósticos del modelo final se alejan en promedio 3083.26 y 55.53 unidades del valor real, esto es, MSE y RMSE, respectivamente

Nuevos pronósticos

Para nuevos pronósticos implementamos el siguiente bloque de código, donde cada variable predictora tiene un valor; esto servirá para predecir el valor del área del tumor:

```
[ ] AreaTumorID600 = pd.DataFrame({'Texture': [18.32],
                                   'Perimeter': [166.82],
                                   'Smoothness': [0.08142],
                                   'Compactness': [0.04462],
                                   'Symmetry': [0.2372],
                                   'FractalDimension': [0.05768]})
    RLMultiple.predict(AreaTumorID600)
```

```
array([[1939.80435773]])
```

CONCLUSIÓN

Este algoritmo que aprendimos a implementar es muy sencillo, pero es muy útil dentro de diferentes áreas, como vimos en este ejemplo, entrenamos un set de datos para predecir el área de un tumor, lo que podría ser decisivo para un tratamiento de cáncer, o para tener una idea si se tratara de un tumor maligno o benigno, o incluso cuan riesgoso es de operar, entre muchas otras opciones. Si esto lo extrapolamos a otra industria u área, podría ayudar en la toma de decisiones de acuerdo con varios factores, lo que queda claro es que es un algoritmo muy potente, y si se combina con otros más resulta ser una buena herramienta para aquellos que consideren implementar sistemas predictivos.