Minería de Datos
Grupo: 1
Dr. Guillermo Gilberto Molero Castillo

Ibañez Guzman Osvaldo
Semestre 2023-1
Tarea #3

Objetivo

Hacer una fusión de datos a través de un algoritmo de aprendizaje automático.

Contexto

Datos sobre retinopatía diabética, tomados con sensores diferentes

Variables

- Patient: Nombre del paciente
- Patient.number: Número de identificación del paciente
- Helth.status: Estado de salud
- Group: Grupo al que pertenece
- f_000001 - f_000070: Valores leídos por los sensores

Desarrollo

Lo primero a realizar es la importación de las bibliotecas que nos ayudarán a la manipulación, creación de matrices, gráficas y formas de visualizar los datos que usaremos, en este caso serán los datos sobre pacientes que sufren de retinopatía diabética, los datos muestran un total de 70 muestras por cada paciente, por lo que buscaremos hacer la fusión en bloques de 10 datos.

Los datos se encuentran en el archivo “DiabeticRetinopathy.csv” por lo que procederemos a leer dichos datos y mostrarlos en pantalla.

```
DatosDiabetes = pd.read_csv("DiabeticRetinopathy.csv")
DatosDiabetes
```

	patient	patient.number	obs	health.status	group	f_000001	f_000002	f_000003	f_000004	f_000005	...	f_000061	
0	paciente151_1	151	1	health	none	2150.745881	1812.296506	1797.425232	2000.402651	2096.775161	...	3.332748	
1	paciente411hpf0.3hzmedian_6	411	6	health	none	201.462060	227.196312	239.527682	242.712595	233.877109	...	15.576554	1
2	paciente237hpf0.3hzmedian_4	237	4	disorder	none	57.266926	42.895909	37.259729	34.940005	31.036661	...	34.657351	3
3	paciente116hpf0.3hzmedian_7	116	7	health	none	229.556540	218.015928	229.962008	214.427812	169.474568	...	18.014346	2
4	paciente131_2	131	2	health	none	94.251665	96.188890	101.315349	102.756134	99.327028	...	7.042471	
...	...	...	...	...	...	...	...	...	...	...	...	...	
1107	paciente396hpf0.3hzmedian_5	396	5	health	none	26.918454	30.309754	32.370351	33.266800	34.510788	...	3.341653	
1108	paciente412hpf0.3hzmedian_3	412	3	health	none	444.511027	430.493455	419.294383	427.814583	464.004222	...	14.687543	1
1109	paciente398hpf0.3hzmedian_7	398	7	health	none	253.650671	337.009270	426.335102	487.426947	525.123416	...	2.039017	
1110	paciente370hpf0.3hzmedian_6	370	6	health	none	1322.642976	1282.723175	1302.099324	1320.569524	1337.036977	...	23.274529	2
1111	paciente406hpf0.3hzmedian_5	406	5	health	none	216.230758	232.530608	224.562959	191.119087	148.402577	...	1.830778	

1112 rows × 75 columns

Posteriormente obtendremos la descripción de los datos, para saber si contamos con valores nulos y en su caso saber cómo tratarlo o eliminarlos de ser necesario, donde se observa que no contamos con valores nulos. Al utilizar la función *describe()* nos mostrar la cantidad de valores por

columna, así como el promedio y su desviación estándar, además los valores mínimos y máximos y sus cuartos percentiles.

```
DatosDiabetes.describe()
```

	patient.number	obs	f_000001	f_000002	f_000003	f_000004	f_000005	f_000006	f_000007	f_000008	...
count	1112.000000	1112.000000	1112.000000	1112.000000	1112.000000	1112.000000	1112.000000	1112.000000	1112.000000	1112.000000	...
mean	270.146583	3.856115	2016.799777	1903.097887	1802.860053	1720.894230	1663.336937	1627.622311	1597.457359	1561.603520	...
std	149.109867	2.121210	5623.463873	5942.420044	6392.486618	6290.537508	5645.872733	5062.374533	4858.983308	4805.405693	...
min	12.000000	1.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	0.000000	...
25%	143.000000	2.000000	94.595032	95.339032	93.371658	88.078723	83.469282	79.522746	75.171477	71.717748	...
50%	261.500000	4.000000	328.782783	310.667368	292.058260	275.527480	271.010004	267.470694	260.898003	255.150837	...
75%	390.250000	6.000000	1332.124207	1250.079057	1168.506726	1113.418396	1045.604225	1007.341541	985.209478	945.582122	...
max	709.000000	16.000000	83032.903920	115217.305100	143934.423200	146001.297100	121126.243000	87568.449470	64605.600060	65526.717950	...

8 rows × 72 columns

Una vez con la descripción de los datos, debemos realizar la fusión de datos, el primer paso es la importación de bibliotecas *linear_model*, *mean_squared_error*, *max_error*, *r2_error*. Estas funciones nos ayudaran a obtener los datos que buscamos para saber que tan confiable fue nuestra fusión.

```
from sklearn import linear_model
from sklearn.metrics import mean_squared_error, max_error, r2_score
```

Procederemos a obtener las variables de control, llamadas “Y_train#”, así como las variables de entrenamiento “X_train#”, para cada uno de los siete bloques, la primer columna será la variable de control, y las restantes nueve serán las de entrenamiento, quedando de la siguiente forma.

```
Y_train1=np.array(Fusion[['f_000001']])
Y_train2=np.array(Fusion[['f_000011']])
Y_train3=np.array(Fusion[['f_000021']])
Y_train4=np.array(Fusion[['f_000031']])
Y_train5=np.array(Fusion[['f_000041']])
Y_train6=np.array(Fusion[['f_000051']])
Y_train7=np.array(Fusion[['f_000061']])
```

```
X_train1=np.array(Fusion[['f_000002','f_000003','f_000004','f_000005','f_000006','f_000007','f_000008','f_000009','f_000010']])
X_train2=np.array(Fusion[['f_000012','f_000013','f_000014','f_000015','f_000016','f_000017','f_000018','f_000019','f_000020']])
X_train3=np.array(Fusion[['f_000022','f_000023','f_000024','f_000025','f_000026','f_000027','f_000028','f_000029','f_000030']])
X_train4=np.array(Fusion[['f_000032','f_000033','f_000034','f_000035','f_000036','f_000037','f_000038','f_000039','f_000040']])
X_train5=np.array(Fusion[['f_000042','f_000043','f_000044','f_000045','f_000046','f_000047','f_000048','f_000049','f_000050']])
X_train6=np.array(Fusion[['f_000052','f_000053','f_000054','f_000055','f_000056','f_000057','f_000058','f_000059','f_000060']])
X_train7=np.array(Fusion[['f_000062','f_000063','f_000064','f_000065','f_000066','f_000067','f_000068','f_000069','f_000070']])
```

Ahora debemos realizar la fusión de los datos, donde usaremos la función *linear_mode.LinearRegression()*, lo que nos permitirá crear la fusión lineal, entre las variables de entrenamiento y las de control, esto lo haremos para poder obtener una nueva columna con los datos estimados, y para poder ver la comparación, los mostraremos en una tabla, comparando las variables de entrenamiento y los datos obtenidos en su fusión lineal.

```
FusionLineal = linear_model.LinearRegression()
FusionLineal.fit(X_train1, Y_train1)
Y_estimacion1 = FusionLineal.predict(X_train1)
pd.DataFrame(Y_estimacion1)
Y_res1 = pd.DataFrame(Y_train1).rename(columns={0:'f_000001'})
Y_res1['FusionLineal'] = Y_estimacion1
Y_res1
```

Finalmente obtendremos algunos datos de dicha fusión, esto para poder saber que tan confiable es dicha aproximación, por lo que obtendremos unas serie de datos como los son los Coeficientes, el Intercepto, residuo, MSE y RMSE y Finalmente el Score, este ultimo es la referencia mas clara de que tan correcta fue nuestra estimación, dicho valor va entre valores de 0 a 1 y entre mas cercano este de 1 serán aproximaciones más confiables.

```
print('Coeficientes: \n', FusionLineal.coef_)
print('Intercepto: \n', FusionLineal.intercept_)
print("Residuo: %.4f" % max_error(Y_train1, Y_estimacion1))
print("MSE: %.4f" % mean_squared_error(Y_train1, Y_estimacion1))
print("RMSE: %.4f" % mean_squared_error(Y_train1, Y_estimacion1, squared=False))
print('Score (Bondad de ajuste): %.4f' % r2_score(Y_train1, Y_estimacion1))
```

Este proceso se realizará con cada uno de los bloques, obteniendo los siguientes datos.

- Primer Bloque

	f_000001	FusionLineal
0	2150.745881	2143.555719
1	201.462060	207.552712
2	57.266926	64.203471
3	229.556540	222.493148
4	94.251665	102.222738
...	...	...
1107	26.918454	35.436919
1108	444.511027	460.918639
1109	253.650671	262.839777
1110	1322.642976	1366.771132
1111	216.230758	224.095823

```
Coeficientes:
 [[  4.80053269 -11.57139106  18.55449899 -21.96426471  20.13030147
  -14.41376425   7.92198903  -3.12289774   0.66129766]]
Intercepto:
 [8.71819797]
Residuo: 941.0289
MSE: 6538.8115
RMSE: 80.8629
Score (Bondad de ajuste): 0.9998
```

- Segundo Bloque

	f_000002	FusionLineal
0	844.858268	855.239840
1	296.698397	301.619377
2	11.138035	11.832930
3	79.388782	79.974980
4	66.385228	66.959887
...	...	...
1107	57.549711	59.445989
1108	354.442472	350.649290
1109	595.371016	588.739834
1110	936.954768	918.113405
1111	127.542226	126.715501

```
Coeficientes:
 [[  4.79275648 -11.11596539  16.89583208 -18.72293033  15.42362268
   -9.14711521   3.49309089  -0.52428316  -0.09269177]]
Intercepto:
 [0.8407671]
Residuo: 1270.5264
MSE: 4491.3888
RMSE: 67.0178
Score (Bondad de ajuste): 0.9998
```

- Tercer Bloque

	f_000003	FusionLineal
0	133.676555	132.047779
1	146.568303	146.947656
2	5.057278	7.962395
3	29.220098	32.478996
4	74.089783	77.152015
...	...	...
1107	28.053109	30.097125
1108	350.849196	348.505223
1109	254.576442	253.447084
1110	585.882022	574.649472
1111	52.514265	54.911389

```
Coeficientes:
 [[  4.40091604 -10.33564733  17.34028365 -22.90621138  24.63812757
  -21.06927013  13.5382593   -6.06804116   1.4666983 ]]
Intercepto:
 [3.0104504]
Residuo: 774.7812
MSE: 2732.2738
RMSE: 52.2712
Score (Bondad de ajuste): 1.0000
```

- Cuarto Bloque

	f_000004	FusionLineal
0	14.852367	13.604921
1	59.985605	57.401275
2	2.613416	1.845556
3	9.752962	8.836480
4	49.584232	48.261049
...	...	...
1107	7.082412	6.176952
1108	133.830701	131.814265
1109	49.315054	47.678920
1110	167.683163	167.829162
1111	19.077191	18.038816

```
Coeficientes:
 [[  5.4646621  -13.951182    22.56598364 -26.52126458  24.26983
  -17.98200565  11.14124061  -5.64728287   1.68903185]]
Intercepto:
 [-0.86787093]
Residuo: 852.1279
MSE: 3261.5737
RMSE: 57.1102
Score (Bondad de ajuste): 1.0000
```

- Quinto Bloque

	f_000005	FusionLineal
0	6.923552	7.137405
1	11.909529	12.297005
2	1.470060	0.878202
3	5.960877	5.841531
4	26.900017	27.593725
...	...	...
1107	1.582758	1.546105
1108	9.708407	9.781487
1109	11.051755	11.361666
1110	38.731026	38.292359
1111	4.963137	5.098913

```
Coeficientes:
 [[  5.54455988 -14.27169415  25.14439238 -34.3951246   37.28287263
  -31.29292898  18.56014153  -6.2489735    0.64064318]]
Intercepto:
 [0.04851217]
Residuo: 454.1070
MSE: 952.8708
RMSE: 30.8686
Score (Bondad de ajuste): 1.0000
```

- Sexto Bloque

	f_000006	FusionLineal
0	1.204269	0.374534
1	3.582889	3.037625
2	8.379165	8.506977
3	7.939512	7.455433
4	6.171104	5.459249
...	...	...
1107	2.022155	1.223596
1108	4.369374	3.755182
1109	2.066788	1.214367
1110	12.595333	12.180500
1111	1.201846	0.323783

```
Coeficientes:
 [[  5.86112328 -15.7978905   26.87635334 -33.12216776  31.35440186
  -22.8166342   11.93130277  -3.89168502   0.61981746]]
Intercepto:
 [-0.91251747]
Residuo: 122.1228
MSE: 81.0466
RMSE: 9.0026
Score (Bondad de ajuste): 1.0000
```

- Séptimo Bloque

	f_000007	FusionLineal
0	3.332748	3.176296
1	15.576554	15.866286
2	34.657351	34.821010
3	18.014346	17.778700
4	7.042471	6.933479
...	...	...
1107	3.341653	3.216594
1108	14.687543	14.705759
1109	2.039017	1.874222
1110	23.274529	23.355598
1111	1.830778	1.709971

```
Coeficientes:
 [[  5.67157617 -14.58501977  22.87103045 -25.47551675  21.82257052
  -14.17047832   6.36339438  -1.66517458   0.17236442]]
Intercepto:
 [-0.18813974]
Residuo: 134.1997
MSE: 125.2841
RMSE: 11.1930
Score (Bondad de ajuste): 1.0000
```

Podemos apreciar en cada uno de los bloques, que el Score o la Bondad de ajuste es 1 o muy cercano a este valor, lo cual nos indica que los valores son similares y la fusión de datos resulto bastante acertada.

Enlace GitHub Tarea 3

Cuaderno Jupyter

https://github.com/OsvaldoIG/MineriaDatos/tree/main/T3