

11 DE ABRIL DE 2023

# PRÁCTICA 8

Fusión de Datos Sobre Retinopatía Diabética

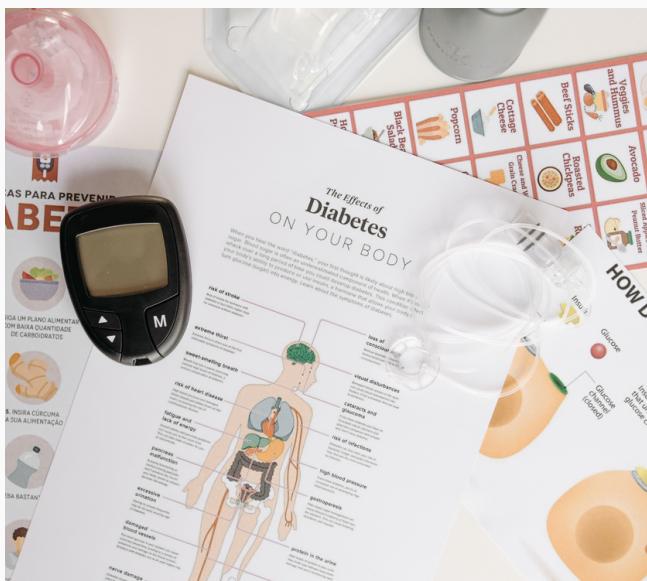


## Objetivo de la práctica

Por Angel Damian Monroy Mendoza

No. de Cuenta: 316040707

Obtener datos fusionados mediante las técnicas tradicionales como el promedio y la mediana, así como mediante el algoritmo de machine learning: Regresión Lineal; lo que puede servir de base para el análisis e identificación de patrones de interés que tengan que ver con el usuario o del entorno, con el objetivo de definir acciones para la inferencia de un determinado contexto.



## Características:

- El conjunto de datos trata sobre la retinopatía diabética, la cual aparece como consecuencia del daño en los vasos sanguíneos de la retina (parte posterior del ojo).
- El azúcar en la sangre, no controlado correctamente, es un factor de riesgo; pues, puede llevar a la adquisición de la enfermedad crónica: Diabetes. Los síntomas más comunes de un descontrol del azúcar en la sangre son:
  - Visión borrosa.
  - Visión oscura.
  - Dificultad para percibir los colores
  - Ceguera.
- Algunos de los datos recabados en este conjunto se refieren al número del paciente, al estado de salud y a 70 diferentes frecuencias que fueron recabadas a partir de estudios de ElectroRetinoGramma (ERG), los cuales arrojan cierto patrón para identificar principios de Retinopatía Diabética.

# Desarrollo

Para el desarrollo de esta práctica dividimos el proceso en seis secciones:

1. Importación de las Bibliotecas.
2. Acceso a los Datos.
3. Análisis Exploratorio de Datos.
4. Fusión de Datos de Modo Tradicional.
5. Fusión de Datos Mediante Algoritmos de ML.
6. Conclusiones.

## 1. Importación de las Bibliotecas.

En este primer apartado se importó la biblioteca de pandas para la manipulación y análisis de los datos, la biblioteca de numpy para crear vectores y matrices de n dimensiones, la biblioteca de matplotlib.pyplot para la generación de gráficas a partir de los datos y la biblioteca de seaborn que, de igual forma, corresponde a la visualización de los datos.

Posteriormente, se instalaron las bibliotecas para el Modelo Lineal, así como lo necesario para saber la eficiencia del modelo y su bondad de ajuste con mean\_squared\_error, mean\_absolute\_error, r2\_score y model\_selection

## 2. Acceso a los Datos.

En este apartado se cargaron los datos correspondientes a la Retinopatía proporcionados por el profesor, los cuales se leyeron como archivo '.csv' con la biblioteca de pandas. Aquí, se eliminaron las columnas que no nos iban a servir para nuestro objetivo de fusionar los datos y se obtuvo el recuento de la variable 'health.status', pues será la variable objetivo cuando se realice el Análisis de Datos y se apliquen algoritmos de Inteligencia Artificial. En este recuento podemos observar que los datos están balanceados, pues tenemos el mismo número de la clase 'Health' y la clase 'Disorder', lo cual nos dará un mejor rendimiento a la hora de aplicar algún algoritmo de clasificación en nuestro conjunto de datos.



health.status	f_000001	f_000002	f_000003
health	2150.745881	1812.296506	1797.425232
health	201.462060	227.196312	239.527682
disorder	57.266926	42.895909	37.259729
health	229.556540	218.015928	229.962008
health	94.251665	96.188890	101.315349
...	...	...	...
1107	26.918454	30.309754	32.370351
1108	444.511027	430.493455	419.294383
1109	253.650671	337.009270	426.335102
1110	1322.642976	1282.723175	1302.099324
1111	216.230758	232.530608	224.562959
1112 rows × 71 columns			
print(Retinopatia.groupby('health.status').size())			
health.status			
disorder	556		
health	556		

# Desarrollo

## 3. Análisis Exploratorio de Datos.

```
Retinopatia.shape  
(1112, 71)  
Retinopatia.isnull().sum()  
  
health.status      0  
f_000001          0  
f_000002          0  
f_000003          0  
f_000004          0
```

### 1) Estructura de los datos.

Para saber la dimensión de nuestro conjunto de datos bastó con utilizar el atributo `shape` de Pandas, pues proporciona la estructura general de los datos obteniendo 1112 filas por 71 columnas. Para conocer los tipos de datos que se tienen, se utilizó el atributo `dtypes`, el cuál nos arrojó que nuestro conjunto de datos tiene una variable categórica de tipo objeto y las demás son numéricas de tipo flotante

### 2) Identificación de Datos faltantes.

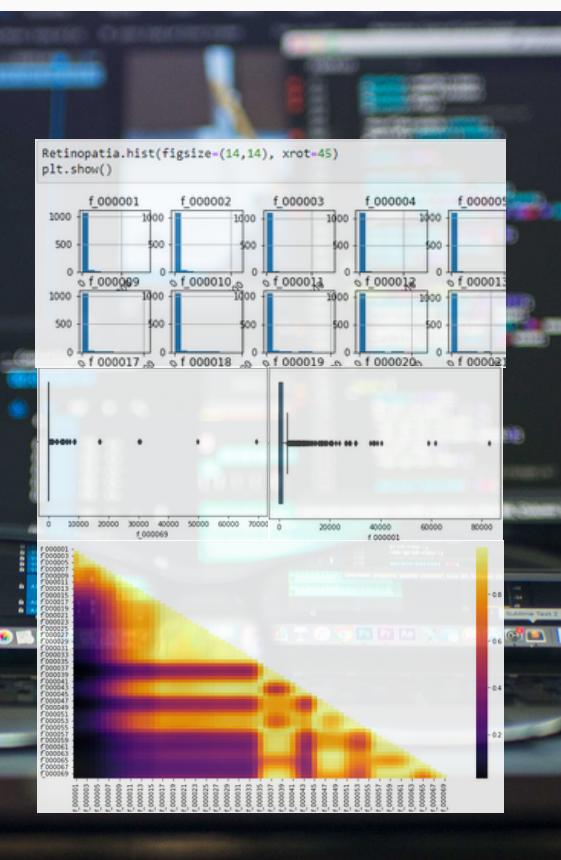
Con el objetivo de identificar si se tenían datos faltantes, se utilizó la función de Pandas `isnull().sum()` que regresa la suma de todos los valores nulos (faltantes) de cada variable. Aquí podemos observar que ninguna de las columnas tuvo suma de valores nulos diferentes de cero. Asimismo, se confirmó este resultado a través de la función `info()` para obtener el tipo de datos y la suma de valores nulos.

### 3) Detección de Valores Atípicos.

Como tenemos varias formas de identificar estos datos, sólo se utilizaron los histogramas para ver la distribución de nuestras variables numéricas y se notó que todas presentaban un sesgo hacia la izquierda, lo cuál indicó que existía la posibilidad de datos atípicos. Para confirmar esto, puesto que son 70 columnas, se realizaron graficas de caja y bigote de 4 columnas tomadas arbitrariamente para observar los datos; aquí se observó que sí habían valores lejanos de la distribución, sin embargo, se decidió no podar estos datos debido a que fueron datos recabados por especialistas.

### 4) Identificación de Relaciones entre Variables.

Para este paso se utilizó la matriz de correlación de Pearson y, dado que son muchas columnas, se realizó un mapa de calor del triángulo inferior sin la etiqueta del valor de su relación para tener una mejor idea de las relaciones que había. Aquí se observó que varias frecuencias tienen una fuerte correlación, sin embargo, no se decidió realizar selección de características, debido a que este mapa de calor serviría como referencia después de la fusión de datos,





D I A B E T E S

# Desarrollo

## 4. Fusión de Datos de Modo Tradicional.

Para este proceso de fusionar los datos de forma tradicional, quitamos la columna que contenía la variable objetivo y nos quedamos con las 70 frecuencias del conjunto de datos. Posteriormente, se creó un DataFrame vacío de nombre Fusion, el cuál fue donde se fueron anexando las columnas obtenidas de las fusiones.

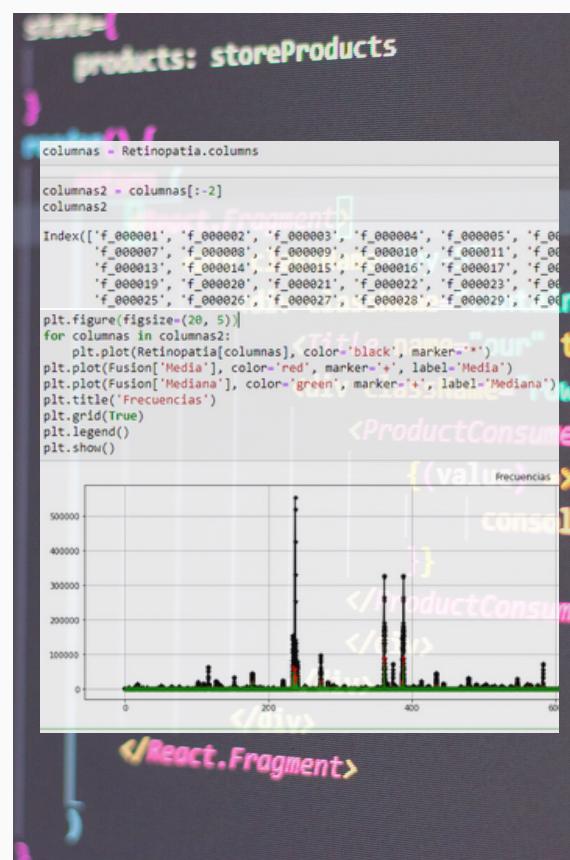
La primer fusión de forma tradicional corresponde a la *media*, que se obtuvo instanciando todos los datos de las 70 frecuencias en un objeto de nombre *PromedioSimple*. Después, a este objeto se le sacó el promedio con la función *.mean(axis='columns')* y se instanció en la primer columna del DataFrame *Fusion*. Asimismo, también se anexó esta columna al conjunto de datos de Retinopatia para obtener una breve comparación entre las 70 frecuencias y la media de todas ellas.

El segundo método tradicional es la *mediana*, que de igual forma se obtuvo instanciando los datos de las 70 frecuencias en un objeto de nombre *Mediana*, en el que se le sacó su mediana con la función *.median(axis='columns')* y se instanció en la segunda columna de *Fusion*. En este paso también se anexó esta columna al conjunto de Retinopatia.

Lo que se buscó después fue graficar el comportamiento de las 70 columnas junto con la *Media* y la *Mediana*, pues se pretende observar si esta forma de fusionar los datos logra obtener una vista global, unificada, coherente y precisa de las frecuencias. Para esto, primero se obtuvieron los nombres de las columnas del conjunto de datos de Retinopatia y se guardó en un objeto de nombre *columnas*. En segundo lugar, como ya habíamos anexado las columnas de *Media* y la *Mediana*, buscamos solo guardar el nombre de las columnas de las frecuencias, por lo que se accedieron a ellas con el objeto *columnas*, desde el principio hasta dos elementos antes del final y se guardó en el objeto *columnas2*. Por último, se realizó una gráfica con un ciclo *for* para graficar cada una de las 70 columnas y después graficar la *Media* y la *Mediana*, y observar el comportamiento.

```
Mediana = Retinopatia.loc[:, 'f_000001':'f_000070']
Fusion['Mediana'] = Mediana.median(axis='columns')
Fusion
```

	Media	Mediana
0	306.643438	14.326068
1	90.611842	32.227710
2	16.251806	7.343551
3	37.748200	18.585033
4	43.123021	42.899036
...	...	...
1107	15.707513	4.688833
1108	166.457003	51.353510





# Desarrollo

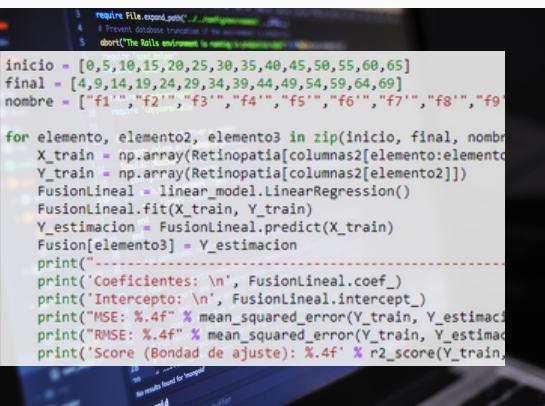
## 5. Fusion de Datos Mediante Algoritmos de ML.

Para la fusión de datos mediante un algoritmo lineal se solicitó que se obtuvieran paquetes de 5 frecuencias en donde 4 de ellas serían las entradas del algoritmo y la 5° sería la estimación, es decir, una frecuencia que estuviera conformada por parte de información de las otras cuatro; a esto se refiere con fusionar los datos mediante un algoritmo de *Machine Learning*, en la que la frecuencia obtenida se le llamó '*frecuencia prima (f#)*'. Como estas frecuencias primas tuvieron que ser de paquetes de 5 frecuencias originales, se obtuvieron un total de 14 frecuencias primas para las que se calcularon sus *coeficientes*, *interceptos*, el *error cuadrático medio*, la *raíz del error cuadrático medio* y su *bondad de ajuste* para cada una de ellas.

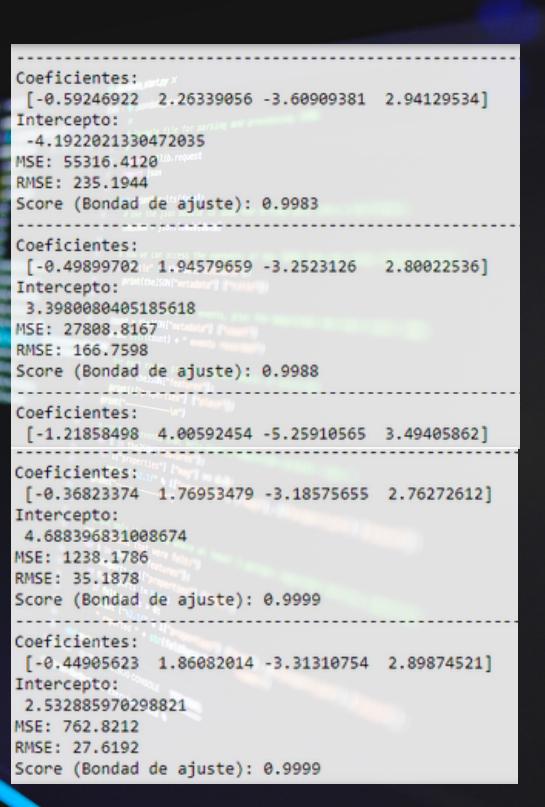
Como las líneas de código que se tenían que ocupar para el algoritmo de *Regresión Lineal* eran repetitivas para los 14 casos, se optó por realizar un bloque de código que pudiera calcular tanto las estimaciones como los parámetros asociados a la *Regresión Lineal*, por esta razón se realizaron tres listas:

- `inicio = [0, 5, 10, 15, ..., 50, 55, 60, 65]`
- `final = [4, 9, 14, 19, ..., 54, 59, 64, 69]`
- `nombre = ["f1", "f2", "f3", ..., "f12", "f13", "f14"]`

Esto fue debido a que, cuando se está aplicando un algoritmo de *Machine Learning*, en casi todos los casos se tienen que tener un conjunto de entrada y salida de prueba, que corresponden a los índices de las columnas que se muestran en las listas *inicio* y *final*. Posteriormente, se instancia el modelo de regresión lineal en un objeto llamado "*FusionLineal*" y este se ajusta con las entradas y salidas de prueba con la función `fit(X_train, Y_train)`. Luego se realiza la estimación (salida), que en nuestro caso son las frecuencias primas, con la ayuda de la función `"predict(X_train)"` y el resultado se guarda en "*Y\_estimacion*" que en la siguiente línea se anexa como columna al DataFrame *Fusion* con el nombre correspondiente en la lista de *nombre*. Por último, se realiza la impresión de los parámetros asociados a la *Regresión Lineal* y todo esto se realiza en un ciclo `for` para ir recorriendo las tres listas ya mencionadas.



```
3 require PTA.expose_path
4 # Prevent database transaction if no connection
5 abort_if_no_connection()
6
7 inicio = [0,5,10,15,20,25,30,35,40,45,50,55,60,65]
8 final = [4,9,14,19,24,29,34,39,44,49,54,59,64,69]
9 nombre = ["f1","f2","f3","f4","f5","f6","f7","f8","f9"]
10
11 for elemento, elemento2, elemento3 in zip(inicio, final, nombre):
12     X_train = np.array(Retinopatia[columnas2[elemento:elemento2]])
13     Y_train = np.array(Retinopatia[columnas2[elemento2:]])
14     FusionLineal = linear_model.LinearRegression()
15     FusionLineal.fit(X_train, Y_train)
16     Y_estimacion = FusionLineal.predict(X_train)
17     Fusion[elemento3] = Y_estimacion
18
19 print("-----")
20 print("Coeficientes: \n", FusionLineal.coef_)
21 print("Intercepto: \n", FusionLineal.intercept_)
22 print("MSE: %.4f" % mean_squared_error(Y_train, Y_estimacion))
23 print("RMSE: %.4f" % mean_squared_error(Y_train, Y_estimacion))
24 print("Score (Bondad de ajuste): %.4f" % r2_score(Y_train,
```

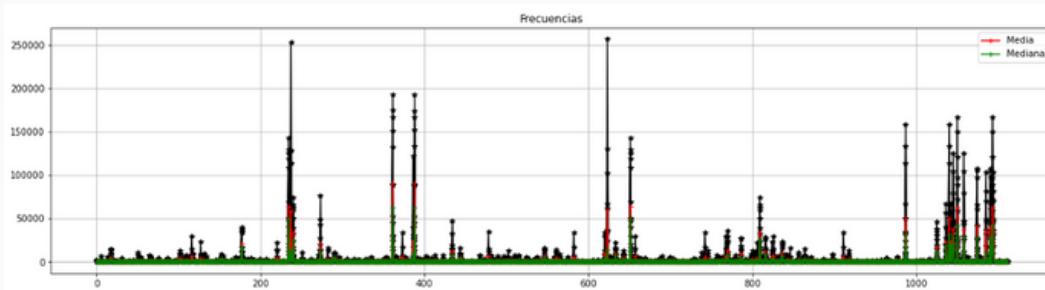


```
Coeficientes:
[-0.59246922  2.26339056 -3.60909381  2.94129534]
Intercepto:
-4.1922021330472035
MSE: 55316.4120
RMSE: 235.1944
Score (Bondad de ajuste): 0.9983
-----
Coeficientes:
[-0.49899702  1.94579659 -3.2523126   2.80022536]
Intercepto:
3.398008405185618
MSE: 27808.8167
RMSE: 166.7598
Score (Bondad de ajuste): 0.9988
-----
Coeficientes:
[-1.21858498  4.00592454 -5.25910565  3.49405862]
-----
Coeficientes:
[-0.36823374  1.76953479 -3.18575655  2.76272612]
Intercepto:
4.688396831008674
MSE: 1238.1786
RMSE: 35.1878
Score (Bondad de ajuste): 0.9999
-----
Coeficientes:
[-0.44905623  1.86082014 -3.31310754  2.89874521]
Intercepto:
2.532885970298821
MSE: 762.8212
RMSE: 27.6192
Score (Bondad de ajuste): 0.9999
```

# Desarrollo

## 5. Fusión de Datos Mediante Algoritmos de ML.

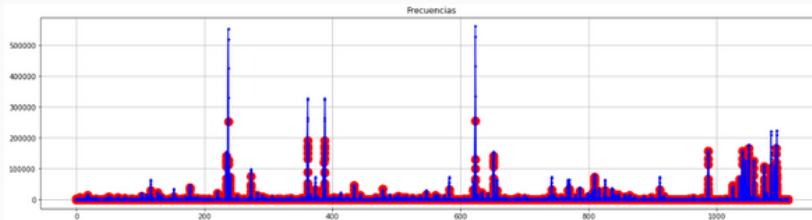
Una vez que se obtuvieron las frecuencias primas y sus respectivos parámetros asociados al algoritmo de Regresión Lineal, se imprimió el DataFrame Fusion para observar los datos fusionados de una forma más condensada y ordenada. Después de esto, se realizó la obtención del nombre de las columnas para que, al igual que con los datos originales, pudieramos graficar la *Media*, la *Mediana* y las frecuencias primas, por lo que, después de guardar el nombre de las columnas de las frecuencias antes mencionadas, se realizó la gráfica mediante un for que recorría los nombres guardados y luego se graficaron la media y la mediana, en donde se obtuvo el siguiente resultado:



Por último, se realizó el mapa de calor de la media, la mediana y las frecuencias primas, con el fin de observar qué tanto cambió con respecto al original y observar la correlación entre pares de variables de los datos fusionados.

## Conclusiones

De esta práctica podemos concluir que la fusión de los datos te ayudan a obtener una vista global, unificada, coherente y precisa del fenómeno que estás estudiando. Esto se pudo confirmar con la siguiente gráfica que muestra cómo las aproximaciones lineales de las frecuencias primas (rojo) describen bien el comportamiento general de los datos (azul), caso contrario a los métodos tradicionales de media y mediana. Esto puede facilitar mejores predicciones y análisis a través del aprendizaje automático, así como una buena base para el análisis e identificación de patrones.



	Media	Mediana	pcr	ft	extens	f2	f3	f4	f5	f6	f7	f8	f9	f10	f11	f12
0	306.643438	14.328068	2220.190601	992.184337												
1	90.611842	32.227710	240.093269	290.028108												
2	16.251806	7.343551	27.264120	16.444638												
3	37.748200	18.585033	153.998872	70.206878												
4	43.123021	42.899036	94.259153	77.438055												
1107	15.787513	4.688833	29.482106	59.596838												
1108	166.457003	51.353510	451.879799	350.371475												
1109	159.912036	24.169915	503.294415	624.601896												
1110	387.832595	67.589846	1300.272459	1031.531271												
1111	43.277631	11.705662	145.674209	114.039823												

1112 rows × 16 columns

