Entrega Análisis y Regresión Lineal - 2025

Integrantes:

- Defelipe, Bianca Eugenia - Legajo: 21341/7.
- Gonzalez, Joaquín Manuel - Legajo: 21247/1.
- Loyola, Yanella Nicole - Legajo: 20912/8.

Dataset Elegido

El Dataset *Obesity Levels Based On Eating Habits and Physical Activites* fue elegido por contener los suficientes atributos **números continuos** para el adecuado **análisis estadístico y matemático** que es requerido para satisfacer los objetivos de la **entrega asignada**.

Link: https://www.kaggle.com/datasets/fatemehmehrparvar/obesity-levels

Sobre los atributos del Dataset

El dataset presenta una combinación de atributos donde se presentan **variables continuas** que permiten realizar un análisis de los factores asociados al nivel de obesidad. Entre esas variables destacan la **edad, altura, peso, cantidad de agua consumida y frecuencia de actividad física** de cada persona, las cuales son adecuadas para la aplicación de métodos de regresión lineal.

Motivación Personal

La **motivación principal** que nos hizo elegir este dataset como grupo radica en que la obesidad es actualmente uno de los principales problemas de salud pública a nivel mundial, con consecuencias tanto físicas como psicológicas.

Regresión Lineal Simple

Definir la variable respuesta y las variables predictoras, justificando el motivo de la elección de estas

Variable Respuesta

- *Weight (Peso):* Variable continua que refleja directamente el estado corporal del individuo y constituye uno de los principales indicadores utilizados para determinar niveles de sobrepeso u obesidad.

Variables Predictoras

- *Height (Altura):* Variable continua que tiene una relación fisiológica con el peso ya que el tamaño corporal condiciona la masa total del individuo. En términos estadísticos, se espera una relación lineal positiva: a mayor altura, mayor peso promedio.
- *Age (Edad):* Variable continua que refleja como el metabolismo y la composición corporal varían con la edad. En general, el peso tiende a aumentar con los años debido a cambios hormonales y reducción de la actividad física, lo que la convierte en una variable predictora de interés.
- *CH2O (Consumo de Agua Diaria):* Variable continua que refleja hábitos saludables y de hidratación. Una ingesta adecuada de agua se asocia a una mejor regulación metabólica y puede influir indirectamente en el control del peso corporal.
- *FAF (Frecuencia de Actividad Física):* Variable continua que representa el nivel de actividad física del individuo. La práctica regular de ejercicio contribuye a un mayor consumo calórico y a la regulación del metabolismo lo que suele asociarse con un mejor control del peso corporal.
- *TUE (Tiempo de Uso de Dispositivos Tecnológicos):* Variable entera que actúa como un indicador de sedentarismo. El uso prolongado de dispositivos tecnológicos se vincula con menor gasto energético y mayor probabilidad de aumento de peso.
- *NCP (Número de Comidas Diarias):* Variable continua que representa la cantidad de ingestas completas que la persona realiza en un día. Mantener una frecuencia adecuada de comidas contribuye a una mejor distribución calórica y puede influir en el mantenimiento del peso corporal.

Realizar un análisis de regresión lineal simple entre la variable respuesta y cada variable predictora, para completar el siguiente cuadro:

Justificación de por qué usar los valores máximos de cada Variable Predictora para el valor de x estrella:

- Tomar el **valor máximo** por cada variable nos permite evaluar el comportamiento del modelo en los valores extremos donde la **incertidumbre** es mayor. Además, en el contexto del análisis sobre Obesidad, creemos que interesa predecir para **individuos con características extremas** (ej: máxima altura, máxima edad). Mantenemos este criterio (valor máximo) para todas las variables ya que nos facilita la **comparación entre los intervalos de predicción** de diferentes predictores.

```
from scipy.stats import t
from sympy import symbols, sqrt
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
import numpy as np

# Cargamos el Dataset
df = pd.read_csv("./Dataset/ObesityDataSet_raw_and_data_sinthetic.csv",
delimiter=",")
```

```
# Asignamos variables
y = df["Weight"] # Variable Respuesta
var_predictoras = df[["Height","Age", "CH2O", "FAF", "TUE", "NCP"]] #
Variables Predictoras
```

```
# Variables Globales
y_promedio = y.mean() # y barra
n = len(y) # tamaño de la muestra
Syy = ((y - y_promedio) ** 2).sum() # cálculo de Syy
alpha = 0.05
t_punto_critico = t.ppf((1 - alpha)/2, n - 2)

def calcular_regresion_lineal_simple_e_intervalos(X, var_x, x_estrella):

    # Cálculo de Estadísticos de Regresión
    x_promedio = X.mean() # x barra
    Sxx = ((X - x_promedio) ** 2).sum() # cálculo de Sxx
    Sxy = ((X - x_promedio) * (y - y_promedio)).sum() # cálculo de Sxy
    beta_uno_somb = Sxy / Sxx # cálculo de beta uno sombrero
    beta_cero_somb = y_promedio - beta_uno_somb * x_promedio # cálculo de
beta cero sombrero
    SSr = Syy - beta_uno_somb * Sxy # cálculo de SSr
    varianza_est = SSr / (n - 2) # cálculo de la Varianza Estimada

    # Definición de Recta de Regresión Estimada
    recta_regresion_est = beta_uno_somb.round(4) * var_x +
beta_cero_somb.round(4)

    # Grado de Ajuste
    R2 = 1 - (SSr / Syy) # cálculo de R2
    r = sqrt(R2) # cálculo del Coeficiente de Correlación Lienal

    # Intervalos de Confianza

    # Intervalo de Confianza para B0
    aux = t_punto_critico * sqrt(varianza_est * (1 / n + (x_promedio ** 2
/ Sxx)))
    IC_beta0_inf = beta_cero_somb - aux
    IC_beta0_sup = beta_cero_somb + aux

    # Intervalo de Confianza para B1
    aux_2 = t_punto_critico * sqrt(varianza_est / Sxx)
    IC_beta1_inf = beta_uno_somb - aux_2
    IC_beta1_sup = beta_uno_somb + aux_2

    # Intervalo de Confianza para la Respuesta Media
    aux_3 = t_punto_critico * sqrt(varianza_est * (1 / n + ((x_estrella -
x_promedio) ** 2 / Sxx)))
```

```
    ICM_y_inf = beta_cero_somb + beta_uno_somb * x_estrella - aux_3
    ICM_y_sup = beta_cero_somb + beta_uno_somb * x_estrella + aux_3

    # Intervalo de Predicción
    y_estrella = recta_regresion_est.evalf(subs = {var_x: x_estrella})
    aux_4 = t_punto_critico * sqrt(varianza_est * (1 + 1 / n +
((x_estrella - x_promedio) ** 2) / Sxx))
    IP_y_inf = y_estrella - aux_4
    IP_y_sup = y_estrella + aux_4

    print(f"Recta de Regresión Estimada: {recta_regresion_est}")
    print(f"varianza_est: {varianza_est:.4f}")
    print(f"R2: {R2:.4f}")
    print(f"Coeficiente de Correlación Lineal: {r:.4f}")
    print(f"Intervalo de Confianza para B1: ({IC_beta1_inf:.4f},
{IC_beta1_sup:.4f})")
    print(f"Intervalo de Confianza para B0: ({IC_beta0_inf:.4f},
{IC_beta0_sup:.4f})")
    print(f"Intervalo de Confianza para la Respuesta Media, con x =
{x_estrella}: ({ICM_y_inf:.4f}, {ICM_y_sup:.4f})")
    print(f"Intervalo de Predicción, con x = {x_estrella}:
({IP_y_inf:.4f}, {IP_y_sup:.4f})")
```

```
# Diccionario para mapear nombres de variables a símbolos de SymPy
simbolos_var = {
    'Height': symbols('x_1', real=True),
    'Age': symbols('x_2', real=True),
    'CH2O': symbols('x_3', real=True),
    'FAF': symbols('x_4', real=True),
    'TUE': symbols('x_5', real=True),
    'NCP': symbols('x_6', real=True)
}

# Iterar sobre cada variable predictora
for i, columna in enumerate(var_predictoras.columns, 1):
    print(f"{'-'*80}")
    print(f"ANÁLISIS {i}/6 - VARIABLE PREDICTORA: {columna}")

    X_actual = var_predictoras[columna] # variable X actual
    simbolo_actual = simbolos_var[columna] # símbolo correspondiente para
esta variable
    x_estrella_actual = X_actual.max()  # Usando el valor máximo de cada
variable

    # Llamar a la función con los parámetros específicos de esta variable
    calcular_regresion_lineal_simple_e_intervalos(
        X=X_actual,
        var_x=simbolo_actual,
```

```
        x_estrella=x_estrella_actual
    )
```

```
--------------------------------------------------------------------------------
ANÁLISIS 1/6 - VARIABLE PREDICTORA: Height
Recta de Regresión Estimada: 130.0048*x_1 - 134.6402
varianza_est: 539.0942
R2: 0.2145
Coeficiente de Correlación Lineal: 0.4631
Intervalo de Confianza para B1: (130.3446, 129.6651)
Intervalo de Confianza para B0: (-134.0612, -135.2192)
Intervalo de Confianza para la Respuesta Media, con x = 1.98: (122.8691,
122.6696)
Intervalo de Predicción, con x = 1.98: (124.2288, 121.3098)
--------------------------------------------------------------------------------
ANÁLISIS 2/6 - VARIABLE PREDICTORA: Age
Recta de Regresión Estimada: 0.836*x_2 + 66.2605
varianza_est: 658.1433
R2: 0.0410
Coeficiente de Correlación Lineal: 0.2026
Intervalo de Confianza para B1: (0.8415, 0.8305)
Intervalo de Confianza para B0: (66.3992, 66.1218)
Intervalo de Confianza para la Respuesta Media, con x = 61.0: (117.4625,
117.0516)
Intervalo de Predicción, con x = 61.0: (118.8785, 115.6345)
--------------------------------------------------------------------------------
ANÁLISIS 3/6 - VARIABLE PREDICTORA: CH2O
Recta de Regresión Estimada: 8.5705*x_3 + 69.3764
varianza_est: 658.6924
R2: 0.0402
Coeficiente de Correlación Lineal: 0.2006
Intervalo de Confianza para B1: (8.6276, 8.5133)
Intervalo de Confianza para B0: (69.4965, 69.2564)
Intervalo de Confianza para la Respuesta Media, con x = 3.0: (95.1545,
95.0212)
Intervalo de Predicción, con x = 3.0: (96.6988, 93.4770)
--------------------------------------------------------------------------------
ANÁLISIS 4/6 - VARIABLE PREDICTORA: FAF
Recta de Regresión Estimada: 88.1862 - 1.5838*x_4
varianza_est: 684.4870
R2: 0.0026
Coeficiente de Correlación Lineal: 0.0514
Intervalo de Confianza para B1: (-1.5418, -1.6258)
Intervalo de Confianza para B0: (88.2416, 88.1307)
Intervalo de Confianza para la Respuesta Media, con x = 3.0: (83.5256,
83.3439)
Intervalo de Predicción, con x = 3.0: (85.0781, 81.7915)
--------------------------------------------------------------------------------
ANÁLISIS 5/6 - VARIABLE PREDICTORA: TUE
```

```
Recta de Regresión Estimada: 88.611 - 3.078*x_5
varianza_est: 682.7882
R2: 0.0051
Coeficiente de Correlación Lineal: 0.0716
Intervalo de Confianza para B1: (-3.0194, -3.1366)
Intervalo de Confianza para B0: (88.6635, 88.5585)
Intervalo de Confianza para la Respuesta Media, con x = 2.0: (82.5413,
82.3686)
Intervalo de Predicción, con x = 2.0: (84.0960, 80.8140)
```

```
ANÁLISIS 6/6 - VARIABLE PREDICTORA: NCP
Recta de Regresión Estimada: 3.6177*x_6 + 76.8702
varianza_est: 678.3762
R2: 0.0115
Coeficiente de Correlación Lineal: 0.1075
Intervalo de Confianza para B1: (3.6634, 3.5720)
Intervalo de Confianza para B0: (76.9980, 76.7424)
Intervalo de Confianza para la Respuesta Media, con x = 4.0: (91.4109,
91.2713)
Intervalo de Predicción, con x = 4.0: (92.9759, 89.7061)
```

Tabla Completa

Y	$\hat{y} = \hat{\beta}_1 x + \hat{\beta}_0$	σ^2	R^2	r	IC(β_1)	IC(β_0)	ICM(Y)	IP(Y)
x_1 (Height)	$130.0048x_1 - 134.6402$	539.0942	0.2145	0.4631	(130.3446, 129.6651)	(-134.0612, -135.2192)	(122.8691, 122.6696)	(124.2288, 121.3098)
x_2 (Age)	$0.836x_2 + 66.2605$	658.1433	0.0410	0.2026	(0.8415, 0.8305)	(66.3992, 66.1218)	(117.4625, 117.0516)	(118.8785, 115.6345)
x_3 (CH2O)	$8.5705x_3 + 69.3764$	658.6924	0.0402	0.2006	(8.6276, 8.5133)	(69.4965, 69.2564)	(95.1545, 95.0212)	(96.6988, 93.4770)
x_4 (FAF)	$88.1862 - 1.5838x_4$	684.4870	0.0026	0.0514	(-1.5418, -1.6258)	(88.2416, 88.1307)	(83.5256, 83.3439)	(85.0781, 81.7915)
x_5 (TUE)	$88.611 - 3.078x_5$	682.7882	0.0051	0.0716	(-3.0194, -3.1366)	(88.6635, 88.5585)	(82.5413, 82.3686)	(84.0960, 80.8140)
x_6 (NCP)	$3.6177x_6 + 76.8702$	678.3762	0.0115	0.1075	(3.6634, 3.5720)	(76.9980, 76.7424)	(91.4109, 91.2713)	(92.9759, 89.7061)

Seleccionar la variable predictora que mejor responde a la variable respuesta y comentar los resultados obtenidos en el cuadro sobre la misma.

La variable predictora que mejor responde a la variable respuesta es aquella con mayor Coeficiente de Correlación Lineal. Si analizamos los valores obtenidos en la tabla podemos ver que la mejor variable predictora es Height con un Coeficiente de Correlación Lineal de 0.4631. Esto nos muestra una relación física coherente con el peso del individuo. Si analizamos más información sobre la variable Height podemos ver que:

- Posee una *relación positiva fuerte*, esto se ve en el valor de β_1 que es de 130, por cada metro de altura, el peso aumenta $\text{aprox. } 130\ kg$.

- Presenta el intervalo de confianza para β_1 *más estrecho* entre todas las variables predictoras.
- Presenta la mejor *Bondad de Ajuste* entre todas las *variables predictoras*.

Regresión Lineal Múltiple

Estimar la ecuación de regresión usando el método de descenso del gradiente.

Notas

- Decidimos normalizar las variables predictoras mediante la media y desviación estándar ya que de esta forma, el método de Descenso de Gradiente puede converger más rápido y de una forma estable. Si las variables presentaran escalas con rangos de valores muy diferentes entre ellas, esto podría hacer que la optimización sea inestable.
- Añadimos una columna de 1s para estimar el intercepto β_0 dentro de la notación matricial $X\beta$
- La fórmula utilizada para calcular el gradiente de la función de costo MSE respecto a cada β es $\nabla J(\beta) = \frac{1}{m} \cdot X^T(X\beta - y)$
- Los coeficientes resultantes se interpretan en unidades de desviación estándar. Para volver a coeficientes en la escala original, hay que transformar de vuelta (o estimar los coeficientes sobre X original).

```
# Normalizamos las variables predictoras
predictoras_normalizadas = (var_predictoras - var_predictoras.mean()) /
var_predictoras.std()

# Agregamos columna de 1s para el Intercepto
X_b = np.c_[np.ones((predictoras_normalizadas.shape[0], 1)),
predictoras_normalizadas]

# Inicialización de Parámetros
m, n = X_b.shape
beta_grad = np.zeros(n) # inicialización de los coeficientes en 0
alpha = 0.01 # tasa de aprendizaje
epochs = 3000 # cantidad de iteraciones
min_err = 1e-6 # error mínimo para cóndición de corte

# Descenso de Gradiente
for epoch in range(epochs):
    y_pred = X_b.dot(beta_grad) # calculamos la predicción actual
    error = y_pred - y # calculamos el vector de errores
    grad = (1/m) * X_b.T.dot(error) # calculamos el gradiente de la
función de costo MSE
    beta_grad -= alpha * grad # actualizamos parámetros hacia la dirección
del mínimo
```

```
    if np.linalg.norm(grad, ord=1) < min_err: # chequeamos condición de
corte por error
        print(f"Convergió en la época {epoch}")
        break

print(f"Coeficientes estimados: {beta_grad}")
recta_prediccion = beta_grad[0].round(4) + beta_grad[1].round(4) *
simbolos_var["Height"] + beta_grad[2].round(4) * simbolos_var["Age"] +
beta_grad[3].round(4) * simbolos_var["CH2O"] + beta_grad[4].round(4) *
simbolos_var["FAF"] + beta_grad[5].round(4) * simbolos_var["TUE"] +
beta_grad[6].round(4) * simbolos_var["NCP"]
print(f"Recta de predicción ŷ = {recta_prediccion}")
y_predichas_grad = X_b.dot(beta_grad)
```

```
Convergió en la época 2589
Coeficientes estimados: [86.58605813 12.94621698  4.82634693  3.54434853
-5.05071038 -0.87279396
  0.3556532 ]
Recta de predicción ŷ = 12.9462*x_1 + 4.8263*x_2 + 3.5443*x_3 - 5.0507*x_4
- 0.8728*x_5 + 0.3557*x_6 + 86.5861
```

Estimar la ecuación de regresión usando el método de mínimos cuadrados. Comentar los resultados obtenidos por ambos métodos.

Notas

- Calculamos la solución teórica utilizando el método de la Pseudoinversa $\hat{\beta} = (X^T X)^{-1} X^T$

```
# Agregamos columna de 1s para el Intercepto
Z_b = np.c_[np.ones(predictoras_normalizadas.shape[0]),
predictoras_normalizadas]

# Mínimos cuadrados
beta_cuad = np.linalg.inv(Z_b.T.dot(Z_b)).dot(Z_b.T).dot(y) # Calculamos
la solución teórica por Pseudoinversa
print(f"Coeficientes estimados: {beta_cuad}")
recta_prediccion = beta_cuad[0].round(4) + beta_cuad[1].round(4) *
simbolos_var["Height"] + beta_cuad[2].round(4) * simbolos_var["Age"] +
beta_cuad[3].round(4) * simbolos_var["CH2O"] + beta_cuad[4].round(4) *
simbolos_var["FAF"] + beta_cuad[5].round(4) * simbolos_var["TUE"] +
beta_cuad[6].round(4) * simbolos_var["NCP"]
print(f"Recta de predicción ŷ = {recta_prediccion}")
y_predichas_cuad = X_b.dot(beta_cuad)
```

```
Coeficientes estimados: [86.58605813 12.94621744  4.82634664  3.54434841
-5.0507107  -0.87279417
```

```
  0.35565302]
Recta de predicción ŷ = 12.9462*x_1 + 4.8263*x_2 + 3.5443*x_3 - 5.0507*x_4
- 0.8728*x_5 + 0.3557*x_6 + 86.5861
```

Comparación entre los resultados obtenidos en los 2 métodos

Método de Descenso de Gradiente

- Este método convergió en la iteración número 2589 y dió estos Coeficientes estimados: [86.58605813 12.94621698 4.82634693 3.54434853 -5.05071038 -0.87279396 0.3556532]. Este método es iterativo y, por lo tanto, aproximado, ya que por cada iteración busca acercarse lo más posible al mínimo absoluto de la función teniendo en cuenta el criterio de corte por error que en nuestro caso fue establecido en $1e-6$.

Método de Mínimos Cuadrados

- Se utilizó la solución teórica mediante el método de la Pseudoinversa ($\hat{\beta} = (X^T X)^{-1} X^T$) que nos proporciona la solución exacta para el dataset seleccionado. Los Coeficientes estimados fueron: [86.58605813 12.94621744 4.82634664 3.54434841 -5.0507107 -0.87279417 0.35565302].

Cabe destacar que ambos métodos nos dan dos conjuntos de Coeficientes estimados muy parecidos, esto nos deja ver que la implementación fue realizada correctamente.

¿La adición de más variables predictoras mejoró la estimación en comparación con la obtenida en el inciso c)? Explique.

Si comparamos los resultados obtenidos entre el análisis de una única varaible en la regresión lineal simple, y el análisis de múltiples variables en la regresión lineal múltiple podemos ver que el Coeficiente de Correlación obtenido en la regresión lineal múltiple supera al obtenido por la variable predictora que mejor se comportaba ("Height") en la regresión lineal simple, ya que $|r_a| > |r| \land |r_a| \text{ se acerca más a } 1 \text{ que } |r|$, por lo tanto, podemos concluir que la adición de más variables predictoras mejoró la estimación en comparación con la obtenida en el inciso c).

```
SSr = ((y - y_predichas_cuad)**2).sum()
SSt = ((y - y.mean())**2).sum()
R2 = 1 - SSr/SSt
print(f"R²: {R2.round(4)}")
k = 6 # número de variables predictoras
R2_adj = 1 - (1 - R2)*(len(y) - 1) / (len(y) - k - 1)
print(f"R² ajustado: {R2_adj.round(4)}")
ra = sqrt(R2_adj)
print(f"ra: {ra.round(4)}")
```

```
R²: 0.3072
R² ajustado: 0.3052
```

```
ra: 0.5524
```

Link al repositorio con el código fuente:
https://github.com/JoaquinManuelGonzalez/Trabajo-Practico-Matematica-4