Resolución de Ejercicios 8.1 - 8.7:
ALUMNO: HENRY CCOARITE DUEÑAS

Análisis Numérico en R

1. Ejercicio 8.1: Análisis de Crecimiento de Usuarios

Explicación

Este ejercicio analiza el crecimiento de usuarios de una startup. Usamos la **primera derivada (tasa de crecimiento)** para ver qué tan rápido crecen mes a mes y la **segunda derivada (aceleración)** para determinar si ese crecimiento se está acelerando o frenando.

Resolución Matemática

Datos: $f(t) = [10, 15, 23, 34, 48, 65, 85]$, $h = 1$.

- **Tasa mes 4 (centrada):**

$$f'(4) \approx \frac{f(5) - f(3)}{2h} = \frac{48 - 23}{2(1)} = 12{,}5$$

- **Tasa mes 1 (adelante):**

$$f'(1) \approx \frac{f(2) - f(1)}{h} = \frac{15 - 10}{1} = 5{,}0$$

- **Tasa mes 7 (atrás):**

$$f'(7) \approx \frac{f(7) - f(6)}{h} = \frac{85 - 65}{1} = 20{,}0$$

- **Aceleración (centrada), ej. mes 2:**

$$f''(2) \approx \frac{f(3) - 2f(2) + f(1)}{h^2} = \frac{23 - 2(15) + 10}{1} = 3{,}0$$

Código R

```r
# --- DATOS INICIALES ---
meses <- 1:7
usuarios <- c(10, 15, 23, 34, 48, 65, 85) # en miles
h <- 1
datos_usuarios <- data.frame(Mes = meses, Usuarios = usuarios)
print("--- DATOS INICIALES ---")
print(datos_usuarios)

# --- TAREA 1: Tasa mes 4 (centrada) ---
i <- 4
tasa_mes4 <- (usuarios[i+1] - usuarios[i-1]) / (2*h)
print("--- TAREA 1: Tasa mes 4 ---")
print(sprintf("f'(4) = %.1f", tasa_mes4))
```

```r

# --- TAREA 2: Tasa mes 1 (adelante) ---
i <- 1
tasa_mes1 <- (usuarios[i+1] - usuarios[i]) / h
print("--- TAREA 2: Tasa mes 1 ---")
print(sprintf("f'(1) = %.1f", tasa_mes1))

# --- TAREA 3: Tasa mes 7 (atras) ---
i <- 7
tasa_mes7 <- (usuarios[i] - usuarios[i-1]) / h
print("--- TAREA 3: Tasa mes 7 ---")
print(sprintf("f'(7) = %.1f", tasa_mes7))

# --- TAREA 4: Aceleracion (centrada) ---
meses_aceleracion <- 2:6
aceleracion <- numeric(length(meses_aceleracion))
for(j in 1:length(meses_aceleracion)) {
  i <- meses_aceleracion[j]
  aceleracion[j] <- (usuarios[i+1] - 2*usuarios[i] + usuarios[i-1]) / h^2
}
resultados_aceleracion <- data.frame(Mes = meses_aceleracion,
                                     Segunda_Derivada = aceleracion)
print("--- TAREA 4: Aceleracion ---")
print(resultados_aceleracion)

# --- TAREA 5: Interpretacion ---
print("--- TAREA 5: Interpretacion ---")
print("Crecimiento acelerado con aceleracion constante (f'' = 3)")
```

Salida (Output)

```
--- DATOS INICIALES ---
  Mes Usuarios
1   1       10
2   2       15
3   3       23
4   4       34
5   5       48
6   6       65
7   7       85
[1] "--- TAREA 1: Tasa mes 4 ---"
[1] "f'(4) = 12.5"
[1] "--- TAREA 2: Tasa mes 1 ---"
[1] "f'(1) = 5.0"
[1] "--- TAREA 3: Tasa mes 7 ---"
[1] "f'(7) = 20.0"
[1] "--- TAREA 4: Aceleracion ---"
  Mes Segunda_Derivada
1   2                3
2   3                3
3   4                3
4   5                3
5   6                3
[1] "--- TAREA 5: Interpretacion ---"
[1] "Crecimiento acelerado con aceleracion constante (f'' = 3)"
```

2. Ejercicio 8.2: Optimización de Función de Pérdida

Explicación

Aquí analizamos la función de **pérdida (Loss)** de un modelo de Machine Learning. La **primera derivada (L')** nos dice qué tan rápido está aprendiendo (convergiendo). La **segunda derivada (L'')** nos dice si la convergencia se está desacelerando (lo cual es bueno, $L'' > 0$) y usamos la derivada para estimar un valor futuro.

Resolución Matemática

Datos: $L(t) = [2{,}45, 1{,}82, 1{,}35, 1{,}08, 0{,}95, 0{,}89]$, $h = 10$.

- **Tasa época 20 (centrada):** $(i = 3)$

$$L'(20) \approx \frac{L(30) - L(10)}{2h} = \frac{1{,}08 - 1{,}82}{2(10)} = -0{,}037$$

- **Segunda derivada época 30 (centrada):** $(i = 4)$

$$L''(30) \approx \frac{L(40) - 2L(30) + L(20)}{h^2} = \frac{0{,}95 - 2(1{,}08) + 1{,}35}{100} = 0{,}0014$$

- **Criterio de parada $|L'| < 0{,}01$:** $L'(40\text{-}50) = (0{,}89 - 0{,}95)/10 = -0{,}006$. Magnitud $0{,}006 < 0{,}01$. Se cumple en **época 50**.

- **Estimación $L(25)$:**

$$L(25) \approx L(20) + L'(20)(25 - 20) = 1{,}35 + (-0{,}037)(5) = 1{,}165$$

Código R

```r
# --- DATOS INICIALES ---
epocas <- c(0, 10, 20, 30, 40, 50)
loss <- c(2.45, 1.82, 1.35, 1.08, 0.95, 0.89)
h <- 10
datos_loss <- data.frame(Epoca = epocas, Loss = loss)
print("--- DATOS INICIALES ---")
print(datos_loss)

# --- TAREA 1: Tasa de cambio en epoca 20 (centrada) ---
i <- 3 # Indice de epoca 20
tasa_loss_20 <- (loss[i+1] - loss[i-1]) / (2*h)
print("--- TAREA 1: Tasa epoca 20 ---")
print(sprintf("L'(20) = %.3f", tasa_loss_20))

# --- TAREA 2: Segunda derivada en epoca 30 (centrada) ---
i <- 4 # Indice de epoca 30
segunda_deriv_30 <- (loss[i+1] - 2*loss[i] + loss[i-1]) / h^2
print("--- TAREA 2: Segunda derivada epoca 30 ---")
print(sprintf("L''(30) = %.4f", segunda_deriv_30))

# --- TAREA 3: Epoca donde |DeltaL/h| < 0.01 ---
n <- length(epocas)
tasa_intervalo <- (loss[2:n] - loss[1:(n-1)]) / h
intervalos_df <- data.frame(
  Intervalo = sprintf("%d-%d", epocas[1:(n-1)], epocas[2:n]),
  Tasa_por_epoca = tasa_intervalo,
  Magnitud = abs(tasa_intervalo)
```

```r
)
print("--- TAREA 3: Criterio de parada ---")
print(intervalos_df, digits=4)
epoca_parada <- epocas[which(intervalos_df$Magnitud < 0.01) + 1]
print(sprintf("Criterio < 0.01 se cumple en epoca: %d", epoca_parada))

# --- TAREA 4: Estimacion loss en epoca 25 ---
epoca_target <- 25
i_base <- 3 # Indice de epoca 20
loss_25 <- loss[i_base] + tasa_loss_20 * (epoca_target - epocas[i_base])
print("--- TAREA 4: Estimacion Loss en 25 ---")
print(sprintf("L(25) aprox L(20) + L'(20)*5 = %.3f", loss_25))
```

Salida (Output)

```
--- DATOS INICIALES ---
  Epoca Loss
1     0 2.45
2    10 1.82
3    20 1.35
4    30 1.08
5    40 0.95
6    50 0.89
[1] "--- TAREA 1: Tasa epoca 20 ---"
[1] "L'(20) = -0.037"
[1] "--- TAREA 2: Segunda derivada epoca 30 ---"
[1] "L''(30) = 0.0014"
[1] "--- TAREA 3: Criterio de parada ---"
  Intervalo Tasa_por_epoca Magnitud
1      0-10        -0.0630   0.0630
2     10-20        -0.0470   0.0470
3     20-30        -0.0270   0.0270
4     30-40        -0.0130   0.0130
5     40-50        -0.0060   0.0060
[1] "Criterio < 0.01 se cumple en epoca: 50"
[1] "--- TAREA 4: Estimacion Loss en 25 ---"
[1] "L(25) aprox L(20) + L'(20)*5 = 1.165"
```

3. Ejercicio 8.3: Análisis de Series Temporales de Ventas

Explicación

Este ejercicio mide la **velocidad (f')** y **aceleración (f'')** de las ventas diarias. Esto nos permite identificar los días de mayor crecimiento (Vie), mayor desaceleración (Mié) y mayor aceleración (Jue), además de usar la última velocidad conocida para predecir las ventas del día siguiente.

Resolución Matemática

Datos: $f(t) = [45, 52, 61, 58, 73, 89, 95]$, $h = 1$.

- **Velocidad (f'):** $f'(1) \approx (52 - 45)/1 = 7{,}0$ (Adelante). $f'(2) \approx (61 - 45)/2 = 8{,}0$ (Centrada). $f'(3) \approx (58 - 52)/2 = 3{,}0$ (Centrada). ... $f'(7) \approx (95 - 89)/1 = 6{,}0$ (Atrás).

- **Aceleración (f”):** $f''(2) \approx (61-2*52+45)/1^2 = 2{,}0$. $f''(3) \approx (58-2*61+52)/1^2 = -12{,}0$. $f''(4) \approx (73 - 2 * 58 + 61)/1^2 = 18{,}0$.
- **Proyección (Día 8):** $f(8) \approx f(7) + f'(7) \times h = 95 + 6{,}0 \times 1 = 101{,}0$.

Código R

```r
# --- DATOS INICIALES ---
ventas <- c(45, 52, 61, 58, 73, 89, 95)
dias <- c("Lun", "Mar", "Mie", "Jue", "Vie", "Sab", "Dom")
h <- 1
n <- length(ventas)
datos_ventas <- data.frame(Dia = dias, Ventas = ventas)
print("--- DATOS INICIALES ---")
print(datos_ventas)

# --- TAREA 1: Velocidad (f') ---
f1_adelante <- (ventas[2] - ventas[1]) / h
f1_central <- (ventas[3:n] - ventas[1:(n-2)]) / (2*h)
f1_atras <- (ventas[n] - ventas[n-1]) / h
velocidad <- c(f1_adelante, f1_central, f1_atras)
df_velocidad <- data.frame(Dia = dias, Velocidad_Ventas = velocidad)
print("--- TAREA 1: Velocidad (f') ---")
print(df_velocidad, digits=3)

# --- TAREA 2: Aceleracion (f'') ---
aceleracion <- rep(NA, n)
for (i in 2:(n-1)) {
    aceleracion[i] <- (ventas[i+1] - 2*ventas[i] + ventas[i-1]) / h^2
}
df_aceleracion <- data.frame(Dia = dias, Aceleracion = aceleracion)
print("--- TAREA 2: Aceleracion (f'') ---")
print(df_aceleracion, digits=3)
max_acel_dia <- dias[which.max(aceleracion)]
print(sprintf("Dia con mayor aceleracion: %s (%.1f)", max_acel_dia, max(aceleracion
    , na.rm=T)))

# --- TAREA 3: Desaceleracion (Mie) ---
f_pp_3 <- aceleracion[3] # f''(Mie)
print("--- TAREA 3: Desaceleracion (Mie) ---")
print(sprintf("La magnitud de la desaceleracion (Mie) fue: %.1f", f_pp_3))

# --- TAREA 4: Extrapolacion Dia 8 ---
f_p_7 <- velocidad[n] # f'(Dom)
f_8 <- ventas[n] + f_p_7 * h
print("--- TAREA 4: Extrapolacion Dia 8 ---")
print(sprintf("f(8) aprox f(7) + f'(7)*h = %.0f + %.1f * %d = %.1f",
            ventas[n], f_p_7, h, f_8))
```

Salida (Output)

```
--- DATOS INICIALES ---
  Dia Ventas
1 Lun     45
2 Mar     52
3 Mie     61
4 Jue     58
5 Vie     73
6 Sab     89
```

```
7 Dom     95
[1] "--- TAREA 1: Velocidad (f') ---"
  Dia Velocidad_Ventas
1 Lun              7.0
2 Mar              8.0
3 Mie              3.0
4 Jue              6.0
5 Vie             15.5
6 Sab             11.0
7 Dom              6.0
[1] "--- TAREA 2: Aceleracion (f'') ---"
  Dia Aceleracion
1 Lun          NA
2 Mar         2.0
3 Mie       -12.0
4 Jue        18.0
5 Vie         1.0
6 Sab       -10.0
7 Dom          NA
[1] "Dia con mayor aceleracion: Jue (18.0)"
[1] "--- TAREA 3: Desaceleracion (Mie) ---"
[1] "La magnitud de la desaceleracion (Mie) fue: -12.0"
[1] "--- TAREA 4: Extrapolacion Dia 8 ---"
[1] "f(8) aprox f(7) + f'(7)*h = 95 + 6.0 * 1 = 101.0"
```

4. Ejercicio 8.4: Gradiente de la Función Sigmoide

Explicación

Calculamos el **gradiente (derivada)** de la función de activación sigmoide, fundamental en *backpropagation*. Comparamos la aproximación numérica (con un paso grande $h = 1$) con la derivada analítica real ($\sigma'(x) = \sigma(x)(1 - \sigma(x))$) para evaluar el error y observar la simetría de la derivada.

Resolución Matemática

Datos: $\sigma(x) = [..., 0{,}2689, 0{,}5000, 0{,}7311, ...]$, $h = 1$.

- **Gradiente en x=0 (centrada):**

$$\sigma'(0) \approx \frac{\sigma(1) - \sigma(-1)}{2h} = \frac{0{,}7311 - 0{,}2689}{2(1)} = 0{,}2311$$

- **Gradiente en x=2 (centrada):**

$$\sigma'(2) \approx \frac{\sigma(3) - \sigma(1)}{2h} = \frac{0{,}9526 - 0{,}7311}{2(1)} = 0{,}11075$$

- **Comparación Analítica (x=0):**

$$\sigma'(0) = \sigma(0)(1 - \sigma(0)) = 0{,}5(1 - 0{,}5) = 0{,}25$$

El error numérico es $0{,}2311 - 0{,}25 = -0{,}0189$.

Código R

```r
# --- DATOS INICIALES ---
x <- c(-3.0, -2.0, -1.0, 0.0, 1.0, 2.0, 3.0)
sigma_x <- c(0.0474, 0.1192, 0.2689, 0.5000, 0.7311, 0.8808, 0.9526)
h <- 1
datos_sigma <- data.frame(x = x, sigma_x = sigma_x)
print("--- DATOS INICIALES ---")
print(datos_sigma)

# --- TAREA 1: sigma'(0) (centrada) ---
i_cero <- which(x == 0)
grad_0_num <- (sigma_x[i_cero + 1] - sigma_x[i_cero - 1]) / (2*h)
print("--- TAREA 1: Gradiente en x=0 ---")
print(sprintf("sigma'(0) num = %.5f", grad_0_num))

# --- TAREA 2: sigma'(-2) y sigma'(2) (centrada) ---
i_neg2 <- which(x == -2)
grad_neg2_num <- (sigma_x[i_neg2 + 1] - sigma_x[i_neg2 - 1]) / (2*h)
i_pos2 <- which(x == 2)
grad_pos2_num <- (sigma_x[i_pos2 + 1] - sigma_x[i_pos2 - 1]) / (2*h)
print("--- TAREA 2: Gradientes en x=-2 y x=2 ---")
print(sprintf("sigma'(-2) num = %.5f", grad_neg2_num))
print(sprintf("sigma'(2) num = %.5f", grad_pos2_num))

# --- TAREA 3: Comparacion con analitica ---
grad_analitico <- sigma_x * (1 - sigma_x)
f1n <- rep(NA, length(x))
f1n[i_cero] <- grad_0_num
f1n[i_neg2] <- grad_neg2_num
f1n[i_pos2] <- grad_pos2_num

df_comp <- data.frame(x = x,
                      Numerico = f1n,
                      Analitico = grad_analitico)
print("--- TAREA 3: Comparacion Numerica vs Analitica ---")
print(df_comp, digits=5)

# --- TAREA 4 y 5: H y Simetria ---
print("--- TAREA 4 y 5: Comentarios ---")
print("h=1 es muy grande, idealmente h seria < 0.01.")
print("La derivada es simetrica (f'(-2) = f'(2)).")
```

Salida (Output)

```
--- DATOS INICIALES ---
    x sigma_x
1 -3.0  0.0474
2 -2.0  0.1192
3 -1.0  0.2689
4  0.0  0.5000
5  1.0  0.7311
6  2.0  0.8808
7  3.0  0.9526
[1] "--- TAREA 1: Gradiente en x=0 ---"
[1] "sigma'(0) num = 0.23110"
[1] "--- TAREA 2: Gradientes en x=-2 y x=2 ---"
[1] "sigma'(-2) num = 0.11075"
[1] "sigma'(2) num = 0.11075"
```

```
[1] "--- TAREA 3: Comparacion Numerica vs Analitica ---"
    x Numerico Analitico
1 -3.0       NA   0.04519
2 -2.0  0.11075   0.10500
3 -1.0       NA   0.19661
4  0.0  0.23110   0.25000
5  1.0       NA   0.19661
6  2.0  0.11075   0.10500
7  3.0       NA   0.04519
[1] "--- TAREA 4 y 5: Comentarios ---"
[1] "h=1 es muy grande, idealmente h seria < 0.01."
[1] "La derivada es simetrica (f'(-2) = f'(2))."
```

5. Ejercicio 8.5: Detección de Anomalías en Latencia

Explicación

Usamos las derivadas para **detectar anomalías** en la latencia de un sistema. La **primera derivada (f')** es muy sensible a cambios bruscos y nos alerta sobre saltos o caídas repentinas. La **segunda derivada (f'')** nos ayuda a localizar el "pico.exacto de la anomalía (el punto de inflexión donde la aceleración cambia de signo).

Resolución Matemática

Datos: $f(t) = [120, 125, 128, 135, 280, 290, 275, 155]$, $h = 1$.

- **Tasa de cambio (f'):** $f'(3) \approx (280 - 128)/(2 * 1) = 76{,}0$ (Anomalía) $f'(6) \approx (155 - 290)/(2 * 1) = -67{,}5$ (Anomalía)
- **Pico de anomalía (f''):** $f''(3) \approx (280 - 2 * 135 + 128)/1^2 = 138$ (Inicio) $f''(4) \approx (290 - 2 * 280 + 135)/1^2 = -135$ (Pico) El cambio de signo de +138 a -135 ocurre en $t = 4$.
- **Salto (3-4), (Adelante):** $f'_{\text{adelante}}(3) = (280 - 135)/1 = 145{,}0$ ms/hora.
- **Momentos** $|f'| > 50$**:** Horas: 3 (76.0), 4 (77.5), 6 (-67.5), 7 (-120.0).

Código R

```r
# --- DATOS INICIALES ---
horas <- 0:7
latencia <- c(120, 125, 128, 135, 280, 290, 275, 155)
h <- 1
n <- length(latencia)
datos_latencia <- data.frame(Hora = horas, Latencia_ms = latencia)
print("--- DATOS INICIALES ---")
print(datos_latencia)

# --- TAREA 1: Tasa de cambio (f') ---
f1_adelante <- (latencia[2] - latencia[1]) / h
f1_central <- (latencia[3:n] - latencia[1:(n-2)]) / (2*h)
f1_atras <- (latencia[n] - latencia[n-1]) / h
velocidad <- c(f1_adelante, f1_central, f1_atras)
df_velocidad <- data.frame(Hora = horas, Tasa_ms_hora = velocidad)
print("--- TAREA 1: Tasa de cambio (f') ---")
print(df_velocidad, digits=3)

# --- TAREA 2: Pico de anomalia (f'') ---
```

```r
aceleracion <- rep(NA, n)
for (i in 2:(n-1)) {
    aceleracion[i] <- (latencia[i+1] - 2*latencia[i] + latencia[i-1]) / h^2
}
df_aceleracion <- data.frame(Hora = horas, Aceleracion = aceleracion)
print("--- TAREA 2: Aceleracion (f'') ---")
print(df_aceleracion, digits=3)
pico_hora <- horas[which(diff(sign(aceleracion)) == -2) + 1]
print(sprintf("El pico (cambio de f'' de + a -) ocurre en la HORA %d", pico_hora))

# --- TAREA 3: Magnitud del salto (Hora 3-4) ---
salto_3_4 <- (latencia[5] - latencia[4]) / h
print("--- TAREA 3: Magnitud del salto (f' adelante en 3) ---")
print(sprintf("f'(3) [adelante] = %.0f ms/hora", salto_3_4))

# --- TAREA 4: Tasa de recuperacion ---
print("--- TAREA 4: Tasa de recuperacion (Hora 6 y 7) ---")
print(sprintf("f'(6) [centrada] = %.1f ms/hora", velocidad[7]))
print(sprintf("f'(7) [atras] = %.1f ms/hora", velocidad[8]))

# --- TAREA 5: Momentos de anomalia (|f'| > 50) ---
umbral_anomalia <- 50
horas_anomalia <- horas[abs(velocidad) > umbral_anomalia]
print("--- TAREA 5: Momentos de anomalia (|f'| > 50) ---")
print(horas_anomalia)
```

Salida (Output)

```
--- DATOS INICIALES ---
  Hora Latencia_ms
1    0         120
2    1         125
3    2         128
4    3         135
5    4         280
6    5         290
7    6         275
8    7         155
[1] "--- TAREA 1: Tasa de cambio (f') ---"
  Hora Tasa_ms_hora
1    0          5.0
2    1          4.0
3    2          5.0
4    3         76.0
5    4         77.5
6    5         -2.5
7    6        -67.5
8    7       -120.0
[1] "--- TAREA 2: Aceleracion (f'') ---"
  Hora Aceleracion
1    0          NA
2    1          -2
3    2           4
4    3         138
5    4        -135
6    5         -25
```

```
7    6        -105
8    7          NA
[1] "El pico (cambio de f'' de + a -) ocurre en la HORA 4"
[1] "--- TAREA 3: Magnitud del salto (f' adelante en 3) ---"
[1] "f'(3) [adelante] = 145 ms/hora"
[1] "--- TAREA 4: Tasa de recuperacion (Hora 6 y 7) ---"
[1] "f'(6) [centrada] = -67.5 ms/hora"
[1] "f'(7) [atras] = -120.0 ms/hora"
[1] "--- TAREA 5: Momentos de anomalia (|f'| > 50) ---"
[1] 3 4 6 7
```

6. Ejercicio 8.6: Tasa de Conversión y ROI Marginal

Explicación

Se analiza el **Retorno de Inversión (ROI) marginal** de una campaña de marketing. La **primera derivada (f')** representa este ROI marginal (cuánto % extra de conversión se obtiene por cada $k adicional). La **segunda derivada (f'')** nos indica si hay **rendimientos decrecientes** (si $f'' < 0$, cada $k adicional rinde menos que el anterior).

Resolución Matemática

Datos: $f(t) = [2{,}1, 3{,}8, 5{,}2, 6{,}1, 6{,}7, 7{,}0]$, $h = 5$.

- **ROI marginal (f'):** $f'(0) \approx (3{,}8 - 2{,}1)/5 = 0{,}34$. $f'(10) \approx (6{,}1 - 3{,}8)/(2 * 5) = 0{,}23$. $f'(25) \approx (7{,}0 - 6{,}7)/5 = 0{,}06$.
- **Rango $f' > 0{,}2$:** Viendo los cálculos, $f'(0) = 0{,}34$, $f'(5) = 0{,}31$, $f'(10) = 0{,}23$. Niveles de gasto: **0, 5, 10 ($k)**.
- **Segunda derivada en $15k (f"):**

$$f''(15) \approx \frac{f(20) - 2f(15) + f(10)}{h^2} = \frac{6{,}7 - 2(6{,}1) + 5{,}2}{5^2} = -0{,}012$$

(Confirma rendimientos decrecientes).

Código R

```r
# --- DATOS INICIALES ---
gasto <- c(0, 5, 10, 15, 20, 25)
conversion <- c(2.1, 3.8, 5.2, 6.1, 6.7, 7.0)
h <- 5
n <- length(gasto)
datos_conversion <- data.frame(Gasto_k = gasto, Conversion_pct = conversion)
print("--- DATOS INICIALES ---")
print(datos_conversion)

# --- TAREA 1: ROI marginal (f') ---
f1_adelante <- (conversion[2] - conversion[1]) / h
f1_central <- (conversion[3:n] - conversion[1:(n-2)]) / (2*h)
f1_atras <- (conversion[n] - conversion[n-1]) / h
roi_marginal <- c(f1_adelante, f1_central, f1_atras)
df_roi <- data.frame(Gasto_k = gasto, ROI_Marginal = roi_marginal)
print("--- TAREA 1: ROI marginal (f') ---")
print(df_roi, digits=3)

```

```r
# --- TAREA 2: Rango de gasto con ROI marginal > 0.2 ---
umbral_roi <- 0.2
gasto_ideal <- gasto[roi_marginal > umbral_roi]
print("--- TAREA 2: Rango de gasto con ROI > 0.2 ---")
print(sprintf("Gasto con ROI > %.1f: %s ($k)", umbral_roi, paste(gasto_ideal,
    collapse=", ")))

# --- TAREA 3: Segunda derivada en $15k ---
i_15k <- which(gasto == 15)
f_pp_15k <- (conversion[i_15k+1] - 2*conversion[i_15k] + conversion[i_15k-1]) / h^2
print("--- TAREA 3: Segunda derivada (f'') en $15k ---")
print(sprintf("f''(15) = %.3f (Rendimientos decrecientes)", f_pp_15k))

# --- TAREA 4: Recomendacion gasto > $25k ---
print("--- TAREA 4: Recomendacion gasto > $25k ---")
print(sprintf("No. El ROI marginal en $25k es %.2f (casi nulo).", roi_marginal[n]))
```

Salida (Output)

```
--- DATOS INICIALES ---
  Gasto_k Conversion_pct
1       0            2.1
2       5            3.8
3      10            5.2
4      15            6.1
5      20            6.7
6      25            7.0
[1] "--- TAREA 1: ROI marginal (f') ---"
  Gasto_k ROI_Marginal
1       0         0.34
2       5         0.31
3      10         0.23
4      15         0.15
5      20         0.09
6      25         0.06
[1] "--- TAREA 2: Rango de gasto con ROI > 0.2 ---"
[1] "Gasto con ROI > 0.2: 0, 5, 10 ($k)"
[1] "--- TAREA 3: Segunda derivada (f'') en $15k ---"
[1] "f''(15) = -0.012 (Rendimientos decrecientes)"
[1] "--- TAREA 4: Recomendacion gasto > $25k ---"
[1] "No. El ROI marginal en $25k es 0.06 (casi nulo)."
```

7. Ejercicio 8.7: Feature Engineering con Derivadas

Explicación

Este ejercicio es una aplicación de **Feature Engineering**. A partir de una señal de temperatura base, creamos nuevas "features"(columnas) que describen su comportamiento: la **Velocidad (f')** y la **Aceleración (f'')**. Estas nuevas features son muy útiles para detectar anomalías (alertas) y para alimentar modelos de Machine Learning, por lo que también las normalizamos.

Resolución Matemática

Datos: $f(t) = [20{,}1, 20{,}3, 20{,}8, 21{,}5, 22{,}6, 24{,}2, 26{,}1, 28{,}5]$, $h = 1$.

- **Tarea 1: Velocidad (f'):** $f'(0) \approx (20{,}3 - 20{,}1)/1 = 0{,}20$ (Adelante). $f'(1) \approx (20{,}8 - 20{,}1)/2 = 0{,}35$ (Centrada). ... $f'(7) \approx (28{,}5 - 26{,}1)/1 = 2{,}40$ (Atrás).
- **Tarea 2: Aceleración (f"):** $f''(1) \approx (20{,}8 - 2 * 20{,}3 + 20{,}1)/1^2 = 0{,}3$. $f''(4) \approx (24{,}2 - 2 * 22{,}6 + 21{,}5)/1^2 = 0{,}5$.
- **Tarea 3: Alerta ($f' > 0{,}8$):** Se activa en $t = 3, 4, 5, 6, 7$.
- **Tarea 4: Normalización f' (ej. $t = 7$):** $\text{mín}(f') = 0{,}20, \text{máx}(f') = 2{,}40$.

$$f'_{\text{norm}}(7) = \frac{2{,}40 - 0{,}20}{2{,}40 - 0{,}20} = 1{,}0$$

Código R

```r
# Cargar los datos iniciales
tiempo <- 0:7
temp <- c(20.1, 20.3, 20.8, 21.5, 22.6, 24.2, 26.1, 28.5)
df <- data.frame(Tiempo = tiempo, Temp = temp)

# h = 1 segundo (seg n el enunciado)
h <- 1
n <- nrow(df)

print("--- DATOS INICIALES ---")
print(df)

# Tarea 1: Calcular la Velocidad (1ra Derivada)
vel_t0 <- (df$Temp[2] - df$Temp[1]) / h
vel_central <- (df$Temp[3:n] - df$Temp[1:(n - 2)]) / (2 * h)
vel_t7 <- (df$Temp[n] - df$Temp[n - 1]) / h
df$Velocidad <- c(vel_t0, vel_central, vel_t7)

print("--- TAREA 1: DATAFRAME CON VELOCIDAD ---")
print(df, digits=3)

# Tarea 2: Calcular la Aceleraci n (2da Derivada)
accel_bordes <- NA
f_x_plus_h <- df$Temp[3:n]
f_x <- df$Temp[2:(n - 1)]
f_x_minus_h <- df$Temp[1:(n - 2)]
accel_central <- (f_x_plus_h - 2 * f_x + f_x_minus_h) / (h^2)
df$Aceleracion <- c(accel_bordes, accel_central, accel_bordes)

print("--- TAREA 2: DATAFRAME CON ACELERACI N ---")
print(df, digits=3)

# Tarea 3: Detecci n de Alerta (> 0.8 C /s)
umbral_alerta <- 0.8
df$Alerta <- ifelse(is.na(df$Velocidad), FALSE, df$Velocidad > umbral_alerta)
momentos_alerta <- df$Tiempo[df$Alerta]

print("--- TAREA 3: DETECCI N DE ALERTA ---")
print(paste("Momentos (segundos) con Velocidad >", umbral_alerta, ":"))
print(momentos_alerta)
print("DataFrame con la columna 'Alerta':")
print(df, digits=3)

# Tarea 4: Normalizaci n Min-Max
min_max_scaler <- function(x) {
  (x - min(x, na.rm = TRUE)) / (max(x, na.rm = TRUE) - min(x, na.rm = TRUE))
}
```

```r
df$Velocidad_Norm <- min_max_scaler(df$Velocidad)
df$Aceleracion_Norm <- min_max_scaler(df$Aceleracion)

print("--- TAREA 4: DATAFRAME FINAL NORMALIZADO ---")
print(df, digits=4)
```

Salida (Output)

```
--- DATOS INICIALES ---
  Tiempo Temp
1      0 20.1
2      1 20.3
3      2 20.8
4      3 21.5
5      4 22.6
6      5 24.2
7      6 26.1
8      7 28.5
--- TAREA 1: DATAFRAME CON VELOCIDAD ---
  Tiempo Temp Velocidad
1      0 20.1      0.20
2      1 20.3      0.35
3      2 20.8      0.60
4      3 21.5      0.90
5      4 22.6      1.35
6      5 24.2      1.75
7      6 26.1      2.15
8      7 28.5      2.40
--- TAREA 2: DATAFRAME CON ACELERACIÓN ---
  Tiempo Temp Velocidad Aceleracion
1      0 20.1      0.20          NA
2      1 20.3      0.35         0.3
3      2 20.8      0.60         0.2
4      3 21.5      0.90         0.4
5      4 22.6      1.35         0.5
6      5 24.2      1.75         0.3
7      6 26.1      2.15         0.5
8      7 28.5      2.40          NA
--- TAREA 3: DETECCIÓN DE ALERTA ---
[1] "Momentos (segundos) con Velocidad > 0.8 :"
[1] 3 4 5 6 7
[1] "DataFrame con la columna 'Alerta':"
  Tiempo Temp Velocidad Aceleracion Alerta
1      0 20.1      0.20          NA FALSE
2      1 20.3      0.35         0.3 FALSE
3      2 20.8      0.60         0.2 FALSE
4      3 21.5      0.90         0.4  TRUE
5      4 22.6      1.35         0.5  TRUE
6      5 24.2      1.75         0.3  TRUE
7      6 26.1      2.15         0.5  TRUE
8      7 28.5      2.40          NA  TRUE
--- TAREA 4: DATAFRAME FINAL NORMALIZADO ---
```

	Tiempo	Temp	Velocidad	Aceleracion	Alerta	Velocidad_Norm	Aceleracion_Norm
1	0	20.1	0.20	NA	FALSE	0.00000000	NA
2	1	20.3	0.35	0.3	FALSE	0.06818182	0.3333333
3	2	20.8	0.60	0.2	FALSE	0.18181818	0.0000000
4	3	21.5	0.90	0.4	TRUE	0.31818182	0.666667
5	4	22.6	1.35	0.5	TRUE	0.52272727	1.0000000
6	5	24.2	1.75	0.3	TRUE	0.70454545	0.3333333
7	6	26.1	2.15	0.5	TRUE	0.88636364	1.0000000
8	7	28.5	2.40	NA	TRUE	1.00000000	NA