Fecha de entrega 5 Junio 2025	Tarea 2 Introducción al Diseño de Experimentos Dra. Norma Icoquih Zapata Peñasco Dra. Gabriela Berenice Díaz Cortés	29 de Mayo 2025

Ejercicio: Construcción manual de una red neuronal de clasificación
Vamos a construir una red neuronal de clasificación con función de activación sigmoide. La arquitectura de la red consta de:

- **Entradas:** 2 características por muestra.
- **Neurona única:** con un vector de pesos y un sesgo.
- **Función de activación:** Sigmoide.
- **Función de pérdida:** Error cuadrático medio (MSE).

Las entradas están dadas por la matriz:

$$X = \begin{bmatrix} 1 & 2 \\ 0 & 1 \\ 2 & 0 \\ 1 & 1 \\ 0 & 0 \\ 1 & 0 \\ 2 & 2 \\ 3 & 1 \\ 0 & 2 \\ 1 & 1 \end{bmatrix}$$

.
Los pesos y sesgo iniciales para la neurona son:

$$\mathbf{w} = [0{,}1, 0{,}2], \quad b = 0{,}1.$$

Las etiquetas reales (targets) son:

$$Y = \begin{bmatrix} 0, 1, 1, 1, 1, 1, 0, 0, 0, 1 \end{bmatrix}.$$

La tasa de aprendizaje es $\eta = 0{,}01$.
Parte 1: Cálculo manual para una muestra
Para una sola muestra de entrada $\mathbf{x} = [x_1, x_2]$ y etiqueta real y, con pesos $\mathbf{w} = [w_1, w_2]$ y sesgo b, se realiza lo siguiente:
Paso 1: Calcula la salida lineal de la neurona

$$z = w_1 x_1 + w_2 x_2 + b.$$

Paso 2: Calcula la salida después de la función de activación sigmoide

$$\hat{y} = \sigma(z) = \frac{1}{1 + e^{-z}}.$$

Paso 3: Calcula la función de pérdida (MSE) para la salida obtenida

$$J = \frac{1}{2}(\hat{y} - y)^2 = \frac{1}{2}\left(\frac{1}{1 + e^{-z}} - y\right)^2.$$

Para simplificar, puedes trabajar con el error de cada neurona:

$$e^{(k)} = \hat{y}^{(k)} - y^{(k)}.$$

Paso 4: Calcula el gradiente de la pérdida respecto a los pesos y el sesgo aplicando la regla de la cadena
En este paso, debes calcular cómo cambia la función de pérdida J con respecto a los parámetros del modelo: los pesos w_1, w_2 y el sesgo b. Para ello, **aplica explícitamente la regla de la cadena paso a paso**.

Parte A: Derivada de la función sigmoide

Primero, demuestra que la derivada de la función sigmoide $\sigma(z)$ está dada por:

$$\sigma(z) = \frac{1}{1+e^{-z}} \quad \Rightarrow \quad \sigma'(z) = \sigma(z)(1-\sigma(z)).$$

Parte B: Aplicación de la regla de la cadena

Sabemos que:

- La función de activación es $\hat{y} = \sigma(z)$, donde $z = w_1 x_1 + w_2 x_2 + b$.
- La función de pérdida es $J = \frac{1}{2}(\hat{y} - y)^2$.

Usando la **regla de la cadena**, escribe paso a paso las derivadas necesarias para obtener:

Gradiente respecto al peso w_1:

$$\frac{\partial J}{\partial w_1} = \frac{\partial J}{\partial \hat{y}} \cdot \frac{\partial \hat{y}}{\partial z} \cdot \frac{\partial z}{\partial w_1}.$$

Gradiente respecto al peso w_2:

$$\frac{\partial J}{\partial w_2} = \frac{\partial J}{\partial \hat{y}} \cdot \frac{\partial \hat{y}}{\partial z} \cdot \frac{\partial z}{\partial w_2}.$$

Gradiente respecto al sesgo b:

$$\frac{\partial J}{\partial b} = \frac{\partial J}{\partial \hat{y}} \cdot \frac{\partial \hat{y}}{\partial z} \cdot \frac{\partial z}{\partial b}.$$

Paso 5: Actualiza los parámetros con gradiente descendente
Con tasa de aprendizaje η:

$$w_1 \leftarrow w_1 - \eta \frac{\partial J}{\partial w_1},$$

$$w_2 \leftarrow w_2 - \eta \frac{\partial J}{\partial w_2},$$

$$b \leftarrow b - \eta \frac{\partial J}{\partial b}.$$

Parte 2: Implementación en Python para 10 muestras

En esta sección, implementarás un programa en Python que automatice el proceso de aprendizaje de una neurona con función de activación sigmoide, utilizando **gradiente descendente** para ajustar los pesos y el sesgo a partir de un conjunto de 10 muestras. Ejecuta el entrenamiento durante 100 epocas para cada una de

las siguientes tasas:

$$\eta \in \{0{,}001, 0{,}01, 0{,}05, 0{,}1, 0{,}5\}.$$

Entrena el modelo usando la función de pérdida (MSE) Grafica la convergencia para cada época, el pseudocodigo es el siguente:

> Para cada tasa de aprendizaje η.
>
> Inicializa los pesos w_1, w_2 y el sesgo b.
>
> Para cada época:
>
> - Para cada muestra, calcula la salida usando la función sigmoide.
> - Calcula el error y los gradientes.
> - Actualiza los pesos y el sesgo con la tasa η.

Parte 3: Cálculo de métricas de clasificación

Luego de obtener las salidas activadas $\hat{y}^{(k)}$ para las 10 muestras, analiza el desempeño del modelo para la **mejor** y la **peor** tasa de aprendizaje (η) ejecutando lo siguiente:

1. **Conversión de salidas a etiquetas binarias:**

   Convierte las salidas continuas de la red en etiquetas binarias utilizando un umbral de $0{,}5$:

   $$\hat{y}_{\text{pred}}^{(k)} = \begin{cases} 1 & \text{si } \hat{y}^{(k)} \geq 0{,}5, \\ 0 & \text{si } \hat{y}^{(k)} < 0{,}5. \end{cases}$$

2. **Cálculo de métricas de desempeño:**

   Evalúa el desempeño del modelo en las 10 muestras usando las siguientes métricas:

   - **Exactitud (Accuracy):**

     $$\text{Accuracy} = \frac{\text{Número de predicciones correctas}}{\text{Número total de predicciones}}.$$

   - **Precisión (Precision):**
     Es la proporción de verdaderos positivos respecto a todos los positivos predichos:

     $$\text{Precision} = \frac{TP}{TP + FP}.$$

   - **Recall (Sensibilidad):**
     Es la proporción de verdaderos positivos respecto a todos los positivos reales:

     $$\text{Recall} = \frac{TP}{TP + FN}.$$

   - **F1-Score:**
     La media armónica entre precisión y sensibilidad:

     $$F1 = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}.$$

   - **Matriz de confusión:**
     La matriz de confusión permite visualizar el desempeño de un clasificador binario. Su estructura es la siguiente:

	Predicho 0	**Predicho 1**
Real 0	TN	FP
Real 1	FN	TP

     Donde:

     - **TP** (True Positives): casos positivos correctamente clasificados.
     - **TN** (True Negatives): casos negativos correctamente clasificados.
     - **FP** (False Positives): casos negativos clasificados incorrectamente como positivos.
     - **FN** (False Negatives): casos positivos clasificados incorrectamente como negativos.

Adicionalmente calcula la frontera de , la cual está dada por la ecuación:

$$w_1 x_1 + w_2 x_2 + b = 0.$$

Despejando x_2 en función de x_1:

$$x_2 = -\frac{w_1}{w_2} x_1 - \frac{b}{w_2}.$$

Grafica los puntos y la frontera de decisión, para los puntos con clase 0 usa un símbolo o color diferente que para puntos con clase 1.