IA - Clase 1B

Aprendizaje de Máquina

(ML – Machine Learning)

Aprendizaje de Máquina
(ML – Machine Learning)

- El objetivo es aprender patrones y relaciones a partir de datos sin tener que programar explícitamente para cada tarea.

- No indicamos paso a paso qué hacer.

- Entrenamos un modelo que ajusta sus parámetros internos para predecir, clasificar o tomar decisiones.

Aprendizaje de Máquina
(ML – Machine Learning)

- El sistema aprende una función o relación a partir de ejemplos para luego generalizar a datos nuevos.

- Un modelo puede entenderse como una función.

Aprendizaje de Máquina
(ML – Machine Learning)

- ¿ Qué es un modelo ?
  - Representación matemática o computacional que captura la relación entre las entradas (caracteŕisticas, variables) y las salidas (predicciones o decisiones) a partir de datos de entrenamiento.

$$\hat{y} = f(\mathbf{x}; \theta)$$

  - x: vector de entradas o variables independientes (ej. temperatura, humedad)
  - θ: parámetros del modelo (valores que se ajustan durante el entrenamiento).
  - f: estructura matemática que transforma las entradas en salidas.
  - Salida: predicción ($\hat{y}$) o decisión que el modelo genera.

Aprendizaje de Máquina (ML – Machine Learning)
Etapas

- Definición del problema
  - ¿Qué queremos predecir, clasificar o detectar?
  - Tipo de aprendizaje (supervisado, no supervisado, por refuerzo).

- Recolección de datos
  - Obtener datos relevantes y suficientes.
  - Fuentes: sensores, bases de datos, APIs, experimentos.

Aprendizaje de Máquina (ML – Machine Learning)
Etapas

- Preprocesamiento de datos
  - Limpieza de datos: manejo de valores faltantes, outliers, escalado.
  - Transformación de variables: normalización, codificación de categorías.

- Selección del modelo
  - Elegir un algoritmo adecuado: regresión, SVM, árboles, redes neuronales, etc.

Aprendizaje de Máquina (ML – Machine Learning)
Etapas

- Entrenamiento del modelo
  - Ajustar los parámetros θ para minimizar el error.
  - División en conjuntos de entrenamiento y validación.

- Evaluación del modelo
  - Medir el rendimiento en datos no vistos (test set).
  - Usar métricas adecuadas (accuracy, F1, MSE, etc.).

Aprendizaje de Máquina (ML – Machine Learning)
Etapas

- Ajuste y optimización
  - Mejorar el modelo ajustando hiperparámetros, cambiando arquitectura o más datos.

- Despliegue y monitoreo
  - Implementar el modelo en producción.
  - Supervisar su rendimiento y reentrenar si es necesario.

Aprendizaje de Máquina (ML – Machine Learning)
Representación Matemática

Representación matemática del proceso. El objetivo es aprender una función f a partir de ejemplos (xi,yi)

$$\hat{y}_i = f(\mathbf{x}_i; \theta)$$

θ es el mejor conjunto de parámetros que vamos a encontrar. Son los parámetros que hacen que el modelo funcione mejor según un criterio.*

$$\theta^* = \arg\min_{\theta} \frac{1}{N} \sum_{i=1}^{N} \ell(y_i, \hat{y}_i)$$

Los parámetros θ son los valores que el algoritmo puede ajustar para aprender (en una regresión lineal, estos son los coeficientes de la recta). Se ajustan minimizando un error global en el conjunto de datos

N es el número total de datos de entrenamiento, la cantidad de ejemplos que tenemos. 1/N es para obtener el error promedio, para que la cantidad total no dependa del nro de ejemplos

$$\ell(y_i, \hat{y}_i)$$

Es la función de pérdida o error para el ejemplo i.Mide qué tan diferente es la predicción del valor real.Puede ser el error cuadrático o la diferencia absoluta.

Aprendizaje de Máquina (ML – Machine Learning)
Representación Matemática

$$\theta^* = \arg\min_{\theta} \frac{1}{N} \sum_{i=1}^{N} \ell(y_i, \hat{y}_i)$$

El operador arg min θ indica que buscamos los parámetros θ que minimizan ese error promedio.

Buscamos encontrar los parámetros θ* que hagan que la función f(x;θ)f prediga lo más cerca posible de los valores reales y, minimizando el error promedio sobre todos los datos.

Aprendizaje de Máquina (ML – Machine Learning)
Función de Pérdida

- La función de pérdida (ℓ) es una fórmula matemática que mide qué tan mal está prediciendo nuestro modelo.
  - Si el valor predicho $\hat{y}_i$ está muy cerca del valor real y, la pérdida es pequeña. Si está muy lejos, la pérdida es grande.
- En muchos modelos clásicos (regresión lineal por ej.) la minimización de la función de pérdida se puede hacer:
  1. Derivando y
  2. buscando el punto donde la derivada es cero (corresponde a un mínimo - o un mínimo local),
  3. y obtengo un mínimo analítico o solución cerrada.

Aprendizaje de Máquina (ML – Machine Learning)
Función de Pérdida

- Ejemplo de Función de pérdida (ℓ) para problemas de regresión, Error Cuadrático Medio (MSE):

$$\mathcal{L}(\theta) = \frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y}_i)^2$$

- N = número de datos
- y_i = valor real
- $\hat{y}_i = f(\mathbf{x}_i;\theta)$ = valor predicho por el modelo
- θ = parámetros del modelo

- El objetivo del entrenamiento es encontrar los parámetros θ que minimicen la función de pérdida.
  - "buscar los valores de θ que hacen que la función de pérdida sea lo más pequeña posible". $\theta^* = \arg\min_{\theta} \mathcal{L}(\theta)$
  - En cálculo, sabemos que para encontrar el mínimo de una función suave podemos derivar la función con respecto a sus parámetros (θ) e igualar la derivada a cero.

Aprendizaje de Máquina (ML – Machine Learning)
Función de Pérdida

- Igualar la derivada a cero.
- Resolver para θ.

$$\frac{\partial \mathcal{L}(\theta)}{\partial \theta} = 0$$

- Si la pérdida tiene solución analítica
  - Podemos derivar y resolver directamente.

- Si la pérdida no tiene solución simple (como en redes neuronales)
  - Usamos métodos iterativos como descenso por gradiente, que ajusta θ en la dirección contraria al gradiente para acercarnos al mínimo.

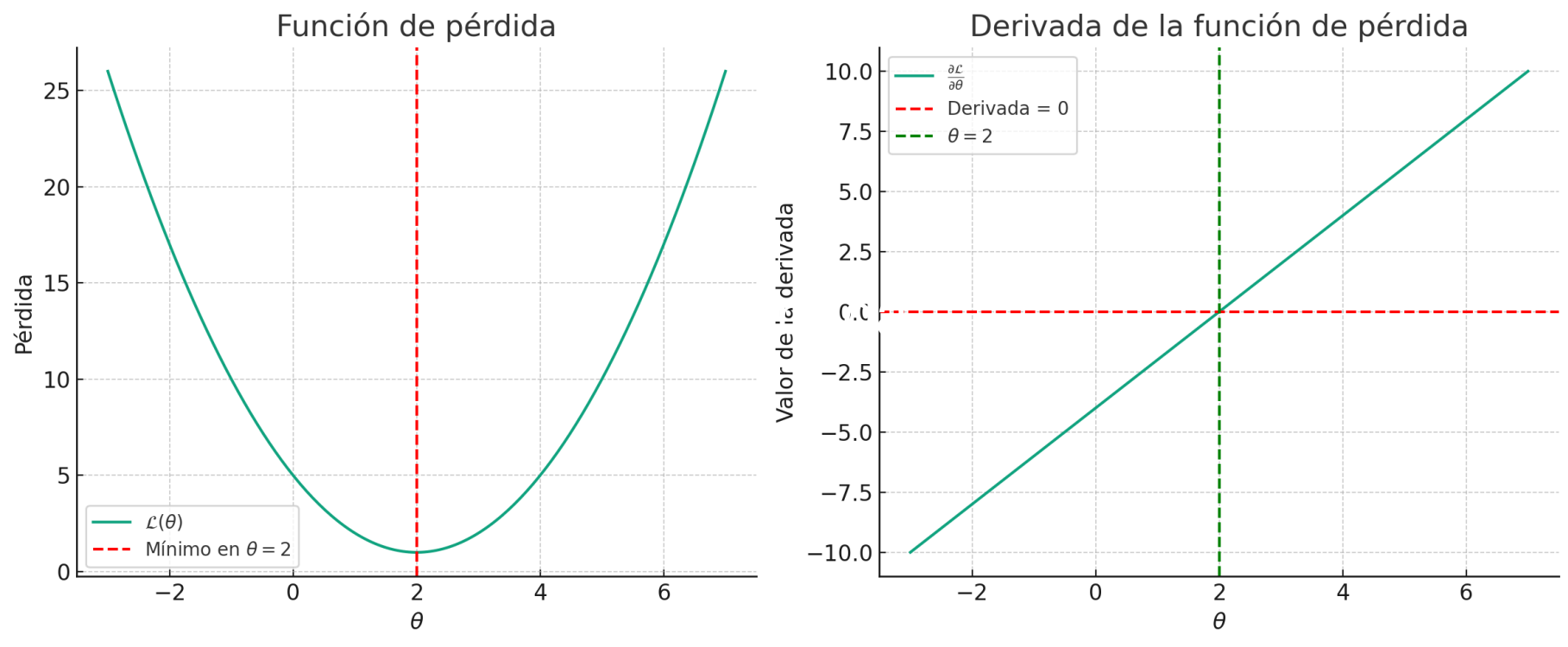
Aprendizaje de Máquina (ML – Machine Learning)
Función de Pérdida

La función de pérdida mide el error.

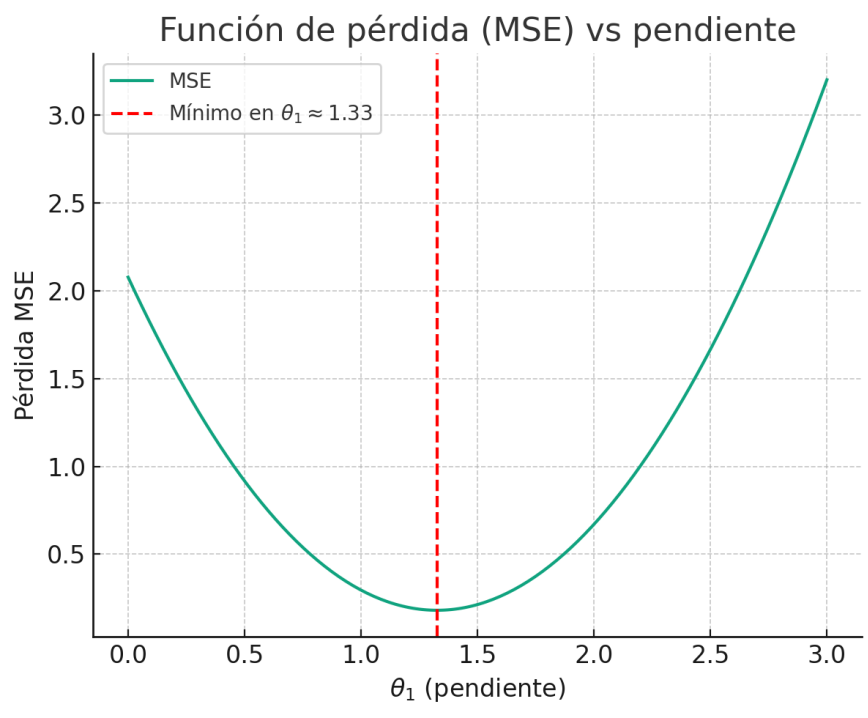
Minimizarla significa encontrar el mejor modelo.

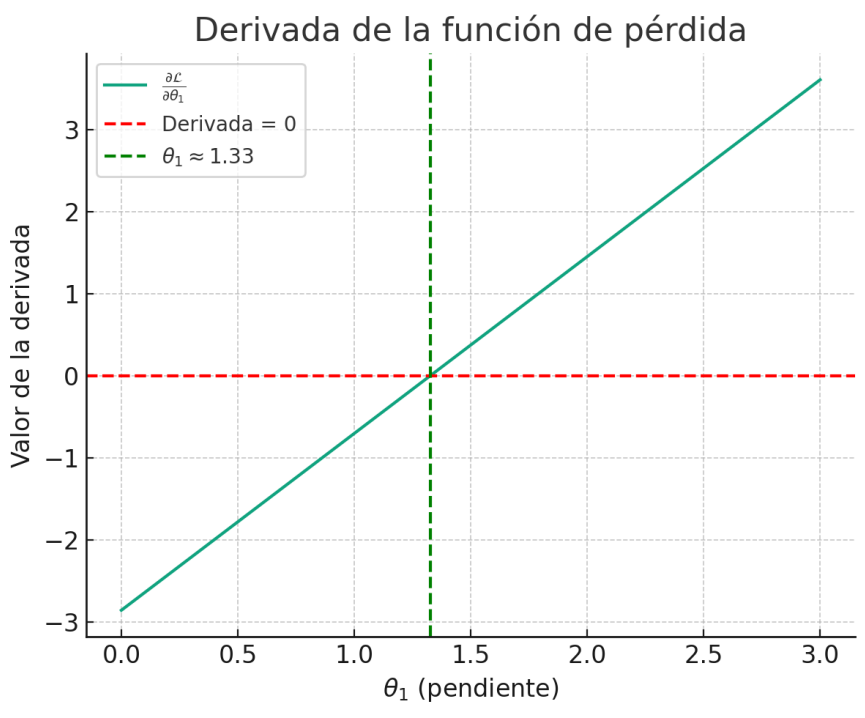
La derivada nos dice en qué dirección mover los parámetros para reducir ese error.

Aprendizaje de Máquina (ML – Machine Learning)
Función de Pérdida

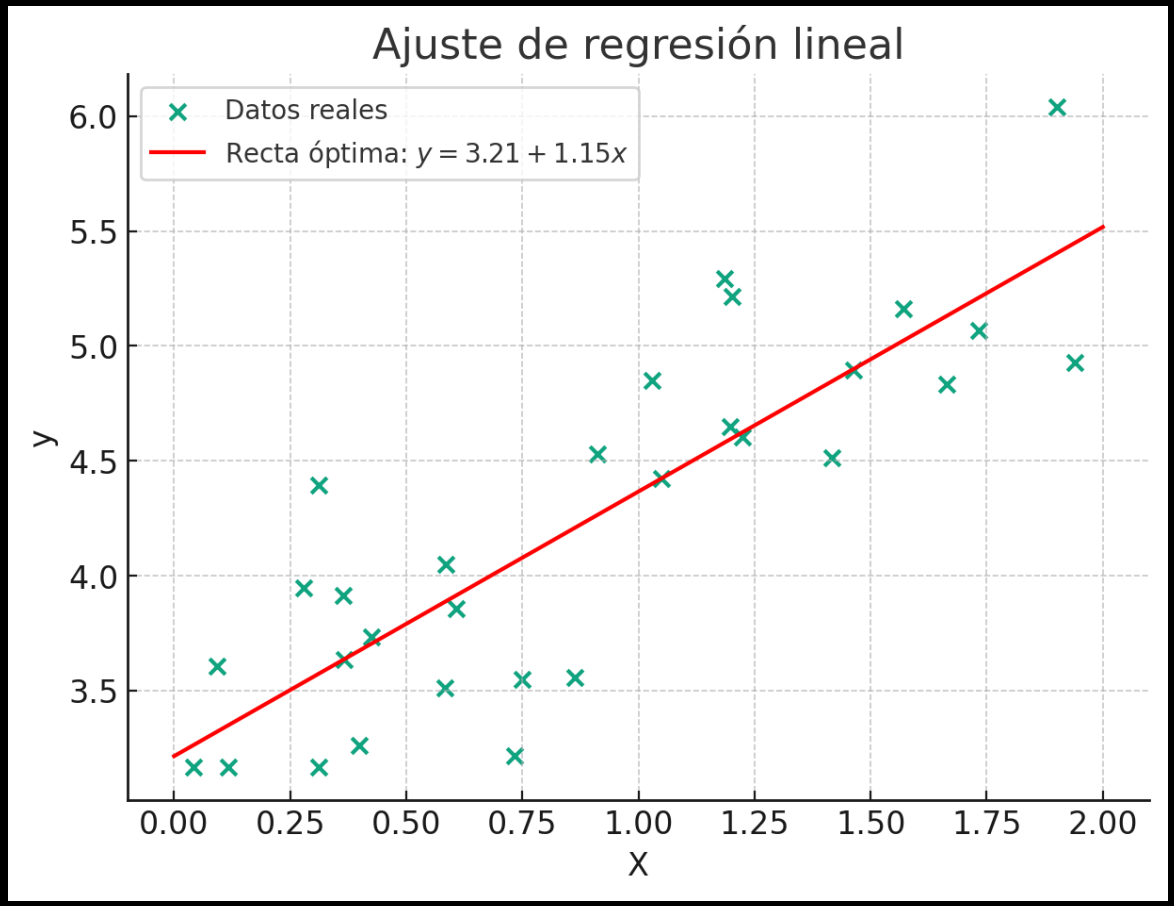


Aprendizaje de Máquina (ML – Machine Learning)
Función de Pérdida





Aprendizaje de Máquina (ML – Machine Learning)
Función de Pérdida



- Los puntos verdes son los datos reales.
- La línea roja es la ecuación que minimiza la función de pérdida (MSE), obtenida derivando y resolviendo.

Aprendizaje de Máquina (ML – Machine Learning)
Función de Pérdida – Caso de Uso

- Simular el siguiente escenario:
  - En un invernadero el objetivo principal es mantener las condiciones óptimas para el cultivo, y el consumo energético es una variable derivada.
- Tenemos un rango óptimo de temperatura para la planta:
  - $T_{min} \leq T_{ideal} \leq T_{max}$
- El consumo energético depende de cuánta diferencia hay entre la temperatura medida y la temperatura ideal.
- El modelo de ML puede aprender a predecir cuánto tiempo y con qué potencia debe trabajar la climatización para mantener ese rango.

Aprendizaje de Máquina (ML – Machine Learning)
Función de Pérdida – Caso de Uso

- Función objetivo
  - Si la temperatura medida T_m está fuera del rango óptimo, hay que corregirla.
  - Podemos definir la variable objetivo (lo que el modelo debe aprender) como la potencia necesaria para llevar Tm a Tideal:
  - $P_{necesaria} = k \cdot |T_{ideal} - T_m|$
  - k es un factor que representa la capacidad térmica del invernadero y la eficiencia del sistema.
- Función de pérdida
  - El modelo puede ser entrenado para minimizar el error entre la potencia predicha y la potencia real necesaria:

$$\mathcal{L}(\theta) = \frac{1}{N} \sum_{i=1}^{N} \left(P_i - \hat{P}_i \right)^2$$

Aprendizaje de Máquina (ML – Machine Learning)
Función de Pérdida – Caso de Uso

- Aplicación práctica
  - El modelo se entrena con datos reales de sensores y potencia aplicada.
- Una vez entrenado, el sistema puede:
  - Leer temperatura actual.
  - Calcular potencia necesaria para alcanzar la temperatura ideal.
  - Ajustar el calefactor o enfriador automáticamente.
- Esto permite que el criterio principal sea el bienestar de la planta y el consumo energético se optimice de forma secundaria, evitando daños por temperaturas fuera del rango óptimo.

Aprendizaje de Máquina (ML – Machine Learning)
Existe una formula predecida o sólo valores ?

- Fórmula predefinida (modelo físico o empírico)
  - Si conocemos las leyes físicas y parámetros del sistema, podemos usar una ecuación predecida.
  - En climatización de invernaderos, el cálculo de energía necesaria para cambiar la temperatura sigue una fórmula basada en calor específico:

$$Q = m \cdot c_p \cdot \Delta T$$

  - Q = energía necesaria (kJ)
  - m = masa de aire dentro del invernadero (kg)
  - c_p = calor específico del aire (~1.005 kJ/kg·°C)
  - ΔT = diferencia entre temperatura ideal y medida (°C)

- Esto nos da una predicción directa de energía sin necesidad de entrenar un modelo, pero requiere conocer dimensiones, pérdidas de calor, infiltraciones, eficiencia, etc.

Aprendizaje de Máquina (ML – Machine Learning)
Existe una formula predecida o sólo valores ?

- Sólo valores (modelo de ML puro)
  - Si no tenemos la fórmula exacta o el sistema es demasiado complejo, usamos Machine Learning para aprender la relación directamente de los datos:
  - Entradas: temperatura medida, humedad, radiación solar, etc.
  - Salida: potencia o tiempo de calefacción requerido.
  - El modelo aprende una función aproximada $\hat{P}$ = *f(sensores)* que minimiza la función de pérdida.
- Aquí no partimos de una ecuación física, sino que el modelo aprende la función a partir de ejemplos.
- Lo ideal puede ser combinar ambos (modelo híbrido físico–ML) para mejorar precisión y robustez.

Aprendizaje de Máquina (ML – Machine Learning)
Función de Pérdida – Cómo se deriva?

$$\mathcal{L}(\theta) = \frac{1}{N} \sum_{i=1}^{N} \left(P_i - \hat{P}_i \right)^2$$

- Buscamos el conjunto de parámetros θ que minimizan el error medio cuadrático entre la potencia real y la predicha: $\theta^* = \arg\min_{\theta} \mathcal{L}(\theta)$

- Modelo lineal elegido: $\hat{P}_i = \theta_0 + \theta_1 T_i$

  - T_i es la temperatura medida por el sensor.
  - θ_0 y θ_1 son los parámetros a aprender.

Aprendizaje de Máquina (ML – Machine Learning)
Función de Pérdida – Cómo se deriva?

- Sustitución en la función de pérdida:

$$\mathcal{L}(\theta_0, \theta_1) = \frac{1}{N} \sum_{i=1}^{N} [P_i - (\theta_0 + \theta_1 T_i)]^2$$

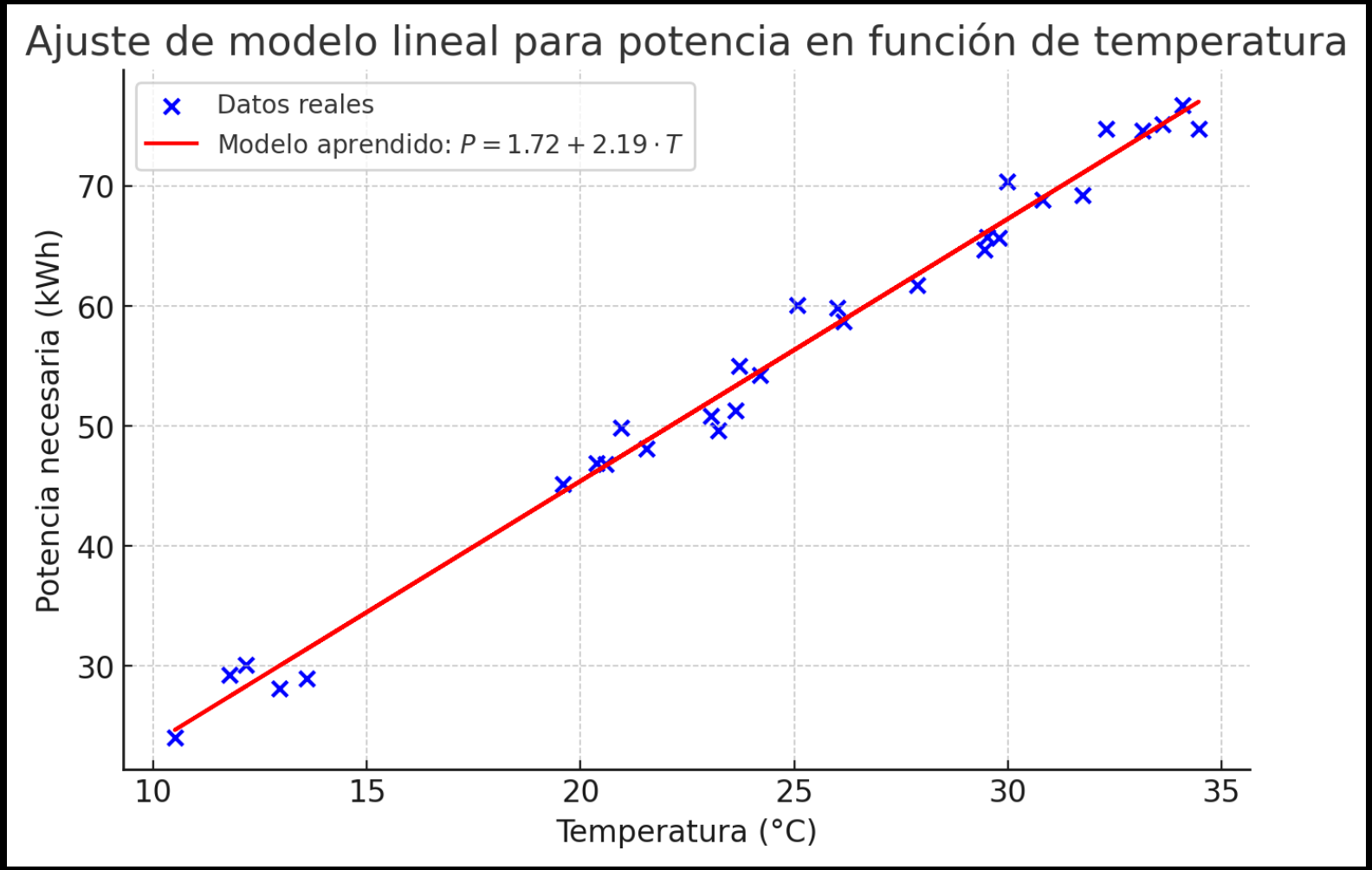
- Derivadas parciales para encontrar el mínimo:
  - Con respecto a θ_0:

$$\frac{\partial \mathcal{L}}{\partial \theta_0} = -\frac{2}{N} \sum_{i=1}^{N} [P_i - (\theta_0 + \theta_1 T_i)] = 0$$

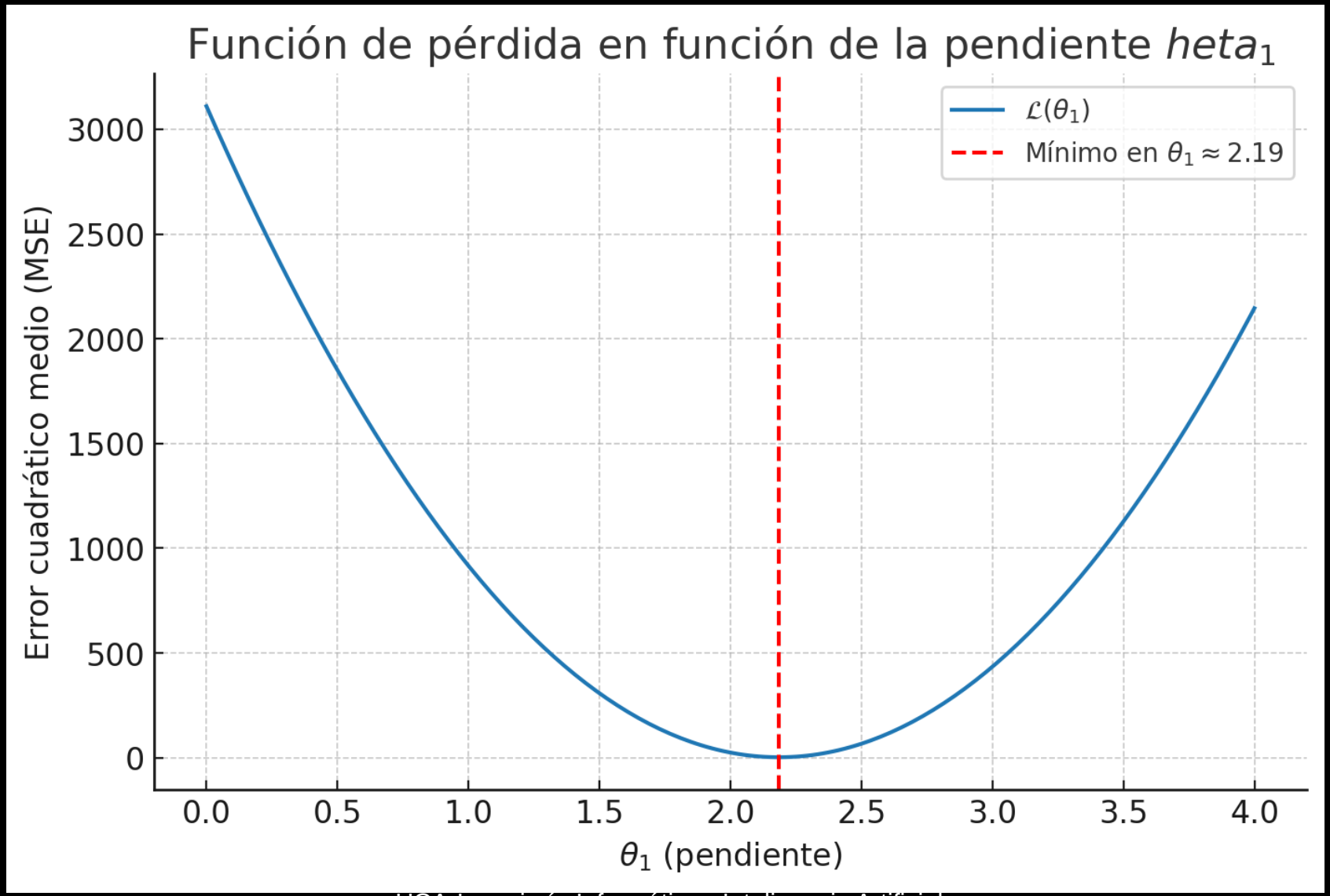
  - Con respecto a θ_1:

$$\frac{\partial \mathcal{L}}{\partial \theta_1} = -\frac{2}{N} \sum_{i=1}^{N} T_i [P_i - (\theta_0 + \theta_1 T_i)] = 0$$

Aprendizaje de Máquina (ML – Machine Learning)
Función de Pérdida – Cómo se deriva?



Aprendizaje de Máquina (ML – Machine Learning)
Función de Pérdida – Cómo se deriva?



Aprendizaje de Máquina
Función de Pérdida – De dónde salen los valores 1.72 y 2.19 ?

- Esos valores son aproximados.
- Salen del proceso de entrenamiento del modelo.
- Calculan la pendiente θ_1 y el intercepto θ_0 de la recta que mejor se ajusta a los datos (T_i,P_i) por mínimos cuadrados.

$$\theta_1 = \frac{\sum_{i=1}^{N}(T_i - \bar{T})(P_i - \bar{P})}{\sum_{i=1}^{N}(T_i - \bar{T})^2}$$

$$\theta_0 = \bar{P} - \theta_1 \cdot \bar{T}$$

Aprendizaje de Máquina

Diferencia entre las fórmulas clásicas y las librerías automáticas de machine learning (ejemplo scikit-learn).

- Método manual (mínimos cuadrados)
  - Se aplican las fórmulas clásicas de estadística para calcular la recta que mejor se ajusta a los datos.
  - Calculamos el promedio de las temperaturas y el promedio de las potencias reales.
  - Medimos cómo varían juntas (covarianza) y cuánto varía la temperatura por sí sola (varianza)
  - Deducimos la pendiente y el punto de inicio de la recta (intercepto).
- Método útil para aprender porque muestra paso a paso cómo se calcula la recta a partir de los datos.
- Sólo funciona bien con una variable (regresión simple). No escala fácilmente a muchos datos o variables.

Aprendizaje de Máquina

Diferencia entre las fórmulas clásicas y las librerías automáticas de machine learning (ejemplo scikit-learn).

- Método scikit-learn (con LinearRegression)
  - Al usar scikit-learn el algoritmo usa herramientas del álgebra lineal para encontrar la mejor recta.

- Este método representa los datos como matrices.
- Técnica matemática llamada ecuación normal para resolver el problema de forma general.
- Puede trabajar con muchas variables, muchos datos y ajustarse a distintos tipos de modelos.
- Valida internamente los datos, detecta errores, optimiza el cálculo, etc.

Aprendizaje de Máquina

Diferencia entre las fórmulas clásicas y las librerías automáticas de machine learning (ejemplo scikit-learn).

- Regresión lineal multivariable (*ecuación normal, permite calcular directamente los parámetros del modelo sin usar iteraciones, como en el descenso por gradiente).*

$$\theta = (\mathbf{X}^\top \mathbf{X})^{-1} \mathbf{X}^\top \mathbf{y}$$

$\mathbf{X}$ es la matriz de entradas de tamaño $N \times (n+1)$

$\mathbf{y}$ es el vector columna de salidas reales, de tamaño $N \times 1$

$\boldsymbol{\theta}$ es el vector columna de parámetros del modelo, de tamaño $(n+1) \times 1$

$\mathbf{X}^\top$ representa la transpuesta de la matriz $\mathbf{X}$

$(\mathbf{X}^\top \mathbf{X})^{-1}$ es la inversa del producto matricial, si $\mathbf{X}^\top \mathbf{X}$ es no singular

Aprendizaje de Máquina

Diferencia entre las fórmulas clásicas y las librerías automáticas de machine learning (ejemplo scikit-learn).

- Resumen:
  - Resultado final:
    - En el caso univariable, los resultados son exactamente iguales (misma pendiente e intercepto).
  - Escalabilidad: La solución manual es limitada a una variable. scikit-learn escala bien a muchas variables.
  - Optimización: Para miles de muestras y muchas variables, scikit-learn puede usar algoritmos más eficientes (ej. QR, SVD o Gradiente).
  - Comodidad: scikit-learn maneja internamente validación, manejo de NaNs, matrices, etc.

Aprendizaje de Máquina (ML – Machine Learning)
Caso de Uso – Steel Frame

- Objetivo:
  - Predecir la geometría óptima de un perfil estructural en steel frame (espesor, sección, etc.) para resistir una determinada presión de viento, sabiendo otras variables del diseño (altura, ancho, separación entre montantes, etc.).

Entrada	Presión + altura + zona
Salida	Espesor e, tipo de perfil C100C100, etc.
Tipo de problema	Ingeniería asistida, Modelo ML Inverso (regresión supervisada)

Aprendizaje de Máquina (ML – Machine Learning)
Caso de Uso – Steel Frame

- Queremos aprender la función que resuelva el problema.
- Queremos un modelo que, dado el vector de condiciones x (carga de viento y geometría del panel), prediga la geometría óptima del perfil: espesor e (regresión) y/o tipo de sección c (clasificación).
  - *f(P,h,w,s,z)→espesor del perfil e*
    - P: presión de viento (N/m^2)
    - h: altura del panel
    - w: ancho del panel
    - s: separación entre montantes (ej. 0.40 m)
    - z: zona o categoría (puede codificarse)
  - Predecir:
    - e: espesor del perfil (en mm)
    - tipo de perfil si planteamos una clasificación (C90, C100, C150, etc.)

Aprendizaje de Máquina (ML – Machine Learning)
Caso de Uso – Steel Frame

- ¿Cómo obtener los datos para entrenar?
  - Hay 3 opciones:
    - Simulación con software de cálculo: generar combinaciones (pressión, altura, ancho) y registrar qué espesor cumple.
    - Normativas (CIRSOC 303, AISI, etc.): usar tablas de diseño estructural como dataset base.
    - Diseños reales existentes.

$$\mathbf{x} = [P,\ h,\ w,\ s,\ z]$$