Inteligencia Artificial

Apuntes del 28 de agosto de 2025 - Semana 4

Juan Diego Jiménez Valverde - 2019199111
juand0908@estudiantec.cr

***Abstract*—Estos apuntes resumen la clase del 28 de agosto de 2025, en la que se analizaron noticias recientes sobre modelos de lenguaje y sus impactos socioeconómicos, así como conceptos matemáticos clave para algoritmos de aprendizaje supervisado. Se cubrieron temas como KNN, regresión lineal, funciones de pérdida, derivadas, descenso del gradiente y sus variantes, epochs y batches, y la diferencia entre MSE y MAE. El resumen enfatiza la relación entre teoría y práctica en la optimización de modelos predictivos.**

***Index Terms*—Modelos de lenguaje, aprendizaje supervisado, KNN, regresión lineal, MSE, descenso del gradiente, optimización, epochs, batches.**

I. Introducción

La clase del 28 de agosto de 2025 se centró primero en discutir cómo los avances en modelos de lenguaje, desde Small Language Models hasta LLMs, están afectando el empleo y la economía, especialmente en ocupaciones susceptibles de automatización. Luego, se abordaron los fundamentos matemáticos que sustentan algoritmos de aprendizaje supervisado, incluyendo KNN y regresión lineal, así como las herramientas necesarias para evaluar y optimizar modelos, como funciones de pérdida, derivadas, descenso del gradiente y sus variantes (batch, stochastic y mini-batch), epochs y batches, y la comparación entre MSE y MAE. Lo que sigue son mis apuntes y reflexiones personales sobre estos temas, explicando cómo los entendí y cómo se aplican en la práctica de la modelación predictiva.

II. Noticias del día

Se mencionaron dos trabajos importantes: primero, un artículo que argumenta que *Small Language Models* (SLMs) pueden ser más prácticos que LLMs en muchos despliegues por costo y adaptabilidad; y segundo, un estudio que muestra efectos tempranos de la IA generativa en el empleo, especialmente perjudicando a jóvenes en ocupaciones altamente automatizables. Estas observaciones nos ayudaron a contextualizar por qué la eficiencia computacional y la interpretabilidad son temas relevantes hoy. [1], [2]

III. Repaso: conceptos clave

A. KNN - K nearest neighbor

KNN es un algoritmo **lazy** (perezoso): no aprende un modelo global, simplemente guarda los datos y en tiempo de consulta busca los K vecinos más cercanos.
K**:** Es el hiperparámetro a seleccionar.
Ventajas: sencillo, interpretable, sin entrenamiento costoso.
Desventajas: costoso en memoria y consulta; sensible a la escala de las features.

B. Regresión Lineal

Concepto básico

- Busca construir un modelo estadístico lineal.
- La relación entre variables debe representarse como una recta.
- Si no es lineal, cae en otra categoría de modelos.

Variables

- **Variable dependiente (y):** valor que queremos predecir.
- **Variable independiente (x):** valor usado para explicar/predicir.
- x: vector D-dimensional (características o *features*).
- w: vector D-dimensional (pendientes/pesos).
- b: número real (intersección con el eje y).

Modelo

$$f_{w,b}(x) = w \cdot x + b$$

- $f(x)$: predicción del modelo.
- $w \cdot x$: producto punto $\rightarrow$ asegura que el resultado sea un escalar.
- Interpretación: combinación lineal de características.

Parámetros del modelo

- F: vector de variables independientes.
- W: pendientes.
- B: intersección con el eje y.
- Modelo parametrizado por w y b.
- Objetivo: encontrar valores óptimos de w y b que permitan predicciones más acertadas.
- Óptimo $\neq$ perfecto, siempre existe error.
- Restricción: solo se pueden modificar w y b, el x es fijo (sample).

Función de pérdida

- Mide qué tan bien o mal está funcionando el modelo (qué tan lejos están las predicciones de los valores reales).

Plot residual

- Un residual es la diferencia entre el valor real y la predicción.
- El *plot residual* muestra gráficamente esas diferencias para analizar la calidad del ajuste.

C. Función de Costo: Error Cuadrático Medio (MSE)

Definición

$$L = \frac{1}{N} \sum_{i=1}^{N} (f_{w,b}(x_i) - y_i)^2$$

Conceptos Clave

- **Loss function:** $(f_{w,b}(x_i) - y_i)^2$ mide la penalidad o error de cada ejemplo individual.
- **Error cuadrático:** penaliza más los errores grandes.
- **Cost function:** promedio de la loss function en todo el dataset; es una medida global del desempeño del modelo.
- **Objetivo:** minimizar L ajustando los parámetros w y b.

Interpretación

- L pequeño $\rightarrow$ mejor modelo.
- L grande $\rightarrow$ peor modelo.
- Reducir L implica mejorar la capacidad predictiva del modelo.

Funciones convexas vs. no convexas

- **Convexa:** garantiza un único mínimo global.
- **No convexa:** pueden aparecer mínimos locales y globales.

D. Repaso de Derivadas

Reglas básicas

- $f(x) = k \quad \Rightarrow \quad f'(x) = 0$
  Ejemplo: $f(x) = 2 \quad \Rightarrow \quad f'(x) = 0$
- $f(x) = x \quad \Rightarrow \quad f'(x) = 1$
- $f(x) = kx \quad \Rightarrow \quad f'(x) = k$
  Ejemplo: $f(x) = 2x \quad \Rightarrow \quad f'(x) = 2$

Potencias

- $f(x) = x^n \quad \Rightarrow \quad f'(x) = nx^{n-1}$
  Ejemplo: $f(x) = x^2 \quad \Rightarrow \quad f'(x) = 2x$

Suma

- $f(x) = u(x) + v(x) \quad \Rightarrow \quad f'(x) = u'(x) + v'(x)$
  Ejemplo: $u(x) = 2x,\ v(x) = 3x \quad \Rightarrow \quad f(x) = 5x,\ f'(x) = 5$

Producto

- $f(x) = u(x)v(x) \quad \Rightarrow \quad f'(x) = u'(x)v(x) + u(x)v'(x)$

Constante sumada

- $f(x) = u(x) + z \quad \Rightarrow \quad f'(x) = u'(x)$
  Ejemplo: $f(x) = 2x + 5 \quad \Rightarrow \quad f'(x) = 2$

Derivadas parciales

- Sea $f(x, y) = 2x + 3y$

$$\frac{\partial f}{\partial x} = 2, \qquad \frac{\partial f}{\partial y} = 3$$

E. Descenso del Gradiente

Concepto básico

- La cantidad de pasos se calcula como: *pendiente* $\times$ α (learning rate).
- Ejemplo: si $x = 1$, el gradiente es $\frac{dy}{dx} = 2x = 2$.
- Para acercarnos al mínimo, nos movemos en la dirección del gradiente negativo con un paso de tamaño α.

Regla de actualización

$$x_{\text{nuevo}} = x_{\text{antiguo}} - \alpha \cdot (2x)$$

- Donde $2x$ es el gradiente.
- El proceso se repite hasta que el gradiente sea 0 (punto de mínimo).

Importancia del α (learning rate)

- El tamaño del paso α debe ser pequeño (ejemplo: 0.1) para no sobrepasar el mínimo.
- Al acercarnos al mínimo, los saltos se reducen porque el gradiente disminuye.
- Un α muy grande puede provocar oscilaciones o incluso alejarse del mínimo.
- Un α demasiado pequeño ralentiza la convergencia.

Nota

- El *learning rate* (α) es un hiperparámetro que debe seleccionarse cuidadosamente.

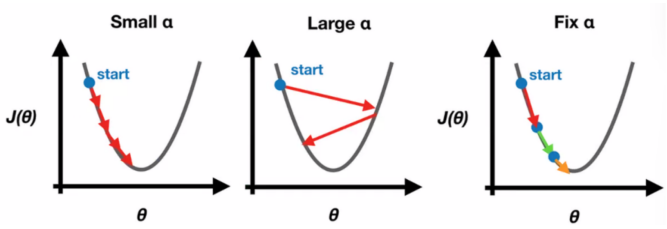

Fig. 1. Impacto del learning rate en gradient descent

F. ¿Por qué usar MSE y no MAE?

El *Mean Squared Error* (MSE) es más utilizado que el *Mean Absolute Error* (MAE) en optimización con descenso del gradiente porque:

- La función MSE es *suave* (diferenciable en todos sus puntos), lo que permite calcular derivadas de forma sencilla y aplicar métodos basados en gradiente.
- En contraste, la función MAE no es diferenciable en 0 (presenta una esquina), lo que complica el uso de derivadas directas y hace más difícil la optimización con gradiente puro.
- Gracias a su naturaleza cuadrática, el MSE penaliza más fuertemente los errores grandes, favoreciendo un ajuste más preciso en esos casos.

Error Cuadrático Medio (MSE):

$$MSE = \frac{1}{N} \sum_{i=1}^{N} \left(f_{w,b}(x_i) - y_i\right)^2$$

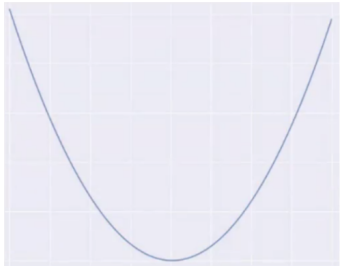

Fig. 2. MSE

Error Absoluto Medio (MAE):

$$MAE = \frac{1}{N} \sum_{i=1}^{N} |f_{w,b}(x_i) - y_i|$$

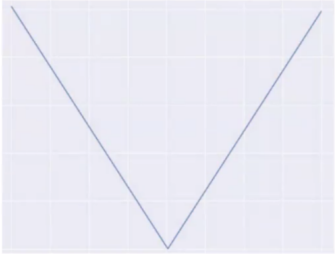

Fig. 3. MAE

IV. Material de clase

A. Derivadas de la función de pérdida (MSE) — notación con ∂/∂

Recordemos la función:

$$L = \frac{1}{N}\sum_{i=1}^{N}\big((wx_i + b) - y_i\big)^2.$$

Derivada por muestra (desglose) — respecto a w: Para la contribución de la muestra i:

$$\ell_i = \big((wx_i + b) - y_i\big)^2.$$

Aplicando la regla de la cadena con derivadas parciales:

$$\frac{\partial \ell_i}{\partial w} = 2\big((wx_i + b) - y_i\big) \cdot \frac{\partial}{\partial w}\big(wx_i + b - y_i\big).$$

Desglose término a término en la parte interior:

$$\frac{\partial}{\partial w}(wx_i + b - y_i) = \frac{\partial}{\partial w}(wx_i) + \frac{\partial}{\partial w}(b) - \frac{\partial}{\partial w}(y_i),$$

$$\frac{\partial}{\partial w}(wx_i) = x_i, \qquad \frac{\partial}{\partial w}(b) = 0, \qquad \frac{\partial}{\partial w}(y_i) = 0,$$

$$\frac{\partial}{\partial w}(wx_i + b - y_i) = x_i.$$

Sustituyendo:

$$\frac{\partial \ell_i}{\partial w} = 2\big((wx_i + b) - y_i\big)\, x_i.$$

Sumando sobre las muestras y normalizando:

$$\boxed{\frac{\partial L}{\partial w} = \frac{1}{N}\sum_{i=1}^{N}\frac{\partial \ell_i}{\partial w} = \frac{2}{N}\sum_{i=1}^{N}\big((wx_i + b) - y_i\big)\, x_i.}$$

Derivada por muestra (desglose) — respecto a b: Para la contribución de la muestra i:

$$\ell_i = \big((wx_i + b) - y_i\big)^2.$$

Regla de la cadena (forma no simplificada):

$$\frac{\partial \ell_i}{\partial b} = 2\big((wx_i + b) - y_i\big) \cdot \frac{\partial}{\partial b}\big(wx_i + b - y_i\big).$$

Desglose término a término en la parte interior:

$$\frac{\partial}{\partial b}(wx_i + b - y_i) = \frac{\partial}{\partial b}(wx_i) + \frac{\partial}{\partial b}(b) - \frac{\partial}{\partial b}(y_i),$$

$$\frac{\partial}{\partial b}(wx_i) = 0, \qquad \frac{\partial}{\partial b}(b) = 1, \qquad \frac{\partial}{\partial b}(y_i) = 0,$$

$$\frac{\partial}{\partial b}(wx_i + b - y_i) = 0 + 1 - 0 = 1.$$

Sustituyendo (forma idéntica a la de tu imagen, antes de simplificar):

$$\frac{\partial \ell_i}{\partial b} = 2\big((wx_i + b) - y_i\big) \cdot 1.$$

Forma final por muestra:

$$\boxed{\frac{\partial \ell_i}{\partial b} = 2\big((wx_i + b) - y_i\big).}$$

Sumando y normalizando para el MSE completo:

$$\boxed{\frac{\partial L}{\partial b} = \frac{1}{N}\sum_{i=1}^{N}\frac{\partial \ell_i}{\partial b} = \frac{2}{N}\sum_{i=1}^{N}\big((wx_i + b) - y_i\big).}$$

Nota:

- Observa que las expresiones por muestra coinciden con lo que aparece en tu imagen: para w aparece el factor extra x_i (porque $(wx_i)' = x_i$), y para b queda solo $2((wx_i + b) - y_i)$ (porque la derivada de b es 1).
- Estas son las cantidades que se usan en la regla de actualización por descenso de gradiente:

$$w \leftarrow w - \alpha\,\frac{\partial L}{\partial w}, \qquad b \leftarrow b - \alpha\,\frac{\partial L}{\partial b}.$$

B. Epochs, Batches y tipos de descenso por gradiente

Epoch:

- Una **epoch** es una iteración completa sobre todo el conjunto de entrenamiento.
- Es un *hiperparámetro* (p. ej. `epochs = 5`).
- Ejemplo: si hay 10 000 samples y ejecutamos 5 epochs, recorremos los 10 000 samples 5 veces en total.
- Podemos aplicar el descenso del gradiente al finalizar un epoch (actualizaciones por epoch) o antes (por batches).

Batch:

- Un **batch** es un subconjunto del conjunto de entrenamiento usado para calcular la gradiente y actualizar parámetros.
- Ejemplo: 10 000 samples y batch size = 1 000 $\Rightarrow$ se necesitan 10 batches para completar 1 epoch.
- No se espera a procesar todo el dataset: cada partición (batch) sirve para calcular la gradiente y actualizar los parámetros.
- Cada vez que procesamos un batch actualizamos los parámetros (o acumulamos gradientes según la estrategia).

Tipos de descenso por gradiente:

a) Batch Gradient Descent (Vanilla):

- Calcula la gradiente usando *todo* el dataset: $\nabla L = \frac{1}{N}\sum_{i=1}^{N}\ldots$.
- Actualización cuando se ha procesado el conjunto completo.
- Ventajas: gradiente estable, pasos consistentes.
- Desventajas:

- Requiere tener todo el dataset en memoria.
- En datasets grandes, las actualizaciones son lentas (cada paso es costoso).
- Gradiente muy estable puede ocultar señales útiles y hacer que el proceso converja a parámetros no deseados según la topología (según el problema).

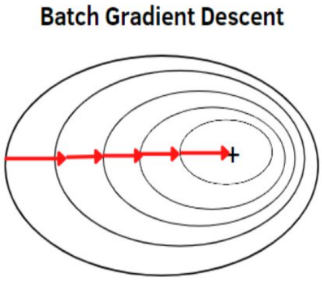

Fig. 4. Batch gradient descent

b) Stochastic Gradient Descent (SGD):

- Actualiza los parámetros por cada sample del training set (o mezcla aleatoria de samples).
- Ventajas: detecta rápidamente si el algoritmo puede converger; útil para datasets muy grandes.
- Desventajas:
  - Señales de gradiente ruidosas (alto ruido en las actualizaciones).
  - Muchas actualizaciones (computacionalmente costoso si no se optimiza).
  - La trayectoria del parámetro es muy oscilatoria:

$$w \leftarrow w - \alpha \frac{\partial L}{\partial w}$$

ejecutado por muestra puede producir movimientos muy erráticos.

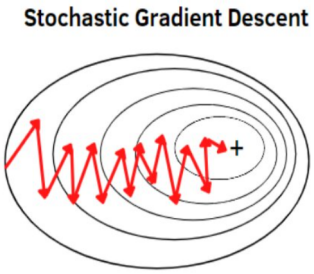

Fig. 5. Stochastic gradient descent

c) Mini-batch Gradient Descent:

- Combina ambas estrategias: se calcula la gradiente sobre batches de tamaño intermedio.
- Ventajas:
  - Reduce el ruido respecto a SGD (más estable) y es más eficiente que Batch GD.
  - Mejora la explotación de hardware (vectorización, GPUs).
  - Ayuda a evitar mínimos locales y aporta robustez en la optimización.
- Desventajas:
  - Introduce un hiperparámetro adicional: *batch size*.
  - Hay que elegir el tamaño del batch cuidadosamente (trade-off entre estabilidad y velocidad).

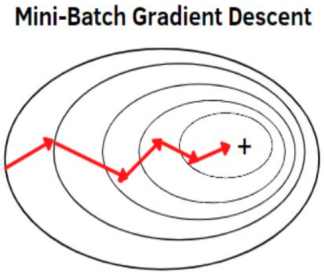

Fig. 6. Mini-batch gradient descent

V. Comentarios prácticos y tareas

Se mencionó que se asignará una tarea práctica: implementar (solo con NumPy) un pipeline de regresión lineal que incluya:

1) Exploración visual del dataset.
2) Ingeniería simple de features (transformaciones no lineales cuando aplique).
3) Implementación de MSE y pasos de descenso (batch / mini-batch).

Eso ayuda a entender por qué algunas funciones no son *smooth* y cómo afecta a las derivadas y la optimización.

VI. Conclusión

En esta clase se consolidó la comprensión de conceptos fundamentales para implementar algoritmos de aprendizaje supervisado de manera eficiente y correcta. Se destacó la relevancia de elegir adecuadamente funciones de pérdida, hiperparámetros como el learning rate y la estrategia de actualización de gradientes, así como la importancia de comprender la teoría detrás de KNN y regresión lineal. Asimismo, los apuntes reflejan la relación entre teoría y práctica, preparando al estudiante para aplicar estos conceptos en tareas concretas y proyectos de programación.

References

[1] Belcak et al., "Small Language Models are the Future of Agentic AI", NVIDIA Research, 2025. https://arxiv.org/abs/2506.02153

[2] E. Brynjolfsson, A. Chandar, y Z. Chen, "Canaries in the Coal Mine? Six Facts about the Recent Employment Effects of Artificial Intelligence", Stanford Digital Economy Lab, 2025. https://digitaleconomy.stanford.edu/publications/canaries-in-the-coal-mine/