# EL MÉTODO DEL GRADIENTE

Aplicado a regresión lineal y regresión logística

# Índice

# ¿Qué es una función de costo?

**Optimización** es una parte muy **importante** de Machine Learning. La mayoría de algoritmos de Machine Learning tienen una **función** que **optimizar**. Cuando ésta se debe **minimizar**, entonces se le llama **función de costo o de pérdida**.

- Una función de costo nos va a **medir** qué tan **bien** nuestro **algoritmo** está **prediciendo** la **salida** esperada.
- Esta función de costo va a fungir como la **función objetivo** para la cual se **desea hallar** el **argumento** que **minimice** su valor.
- **No hay** una misma **función** que **funcione** para **todos** los **tipos** de **datos**, depende de: tipo de problema, presencia de outliers, eficiencia computacional, facilidad de cálculo de las derivadas, etc.

Funciones de costo para **problemas** de **regresión**:

- MSE, MAE, MAPE, Quantile Loss

Funciones de costo para **problemas** de **clasificación**:

- Log loss, Focal loss, Exponential loss, Hinge loss

# Error Cuadrático Medio (ECM)

1. **Diferencia** entre el valor actual de “y” y el valor predicho: $(y_i - \hat{y}_i)$

2. **Cuadrado** de la diferencia: $(y_i - \hat{y}_i)^2$

3. **Media de la suma** de los cuadrados: $\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2$

( $\hat{y}_i$ depende de la función de nuestro modelo)

# ¿Qué es el gradiente?

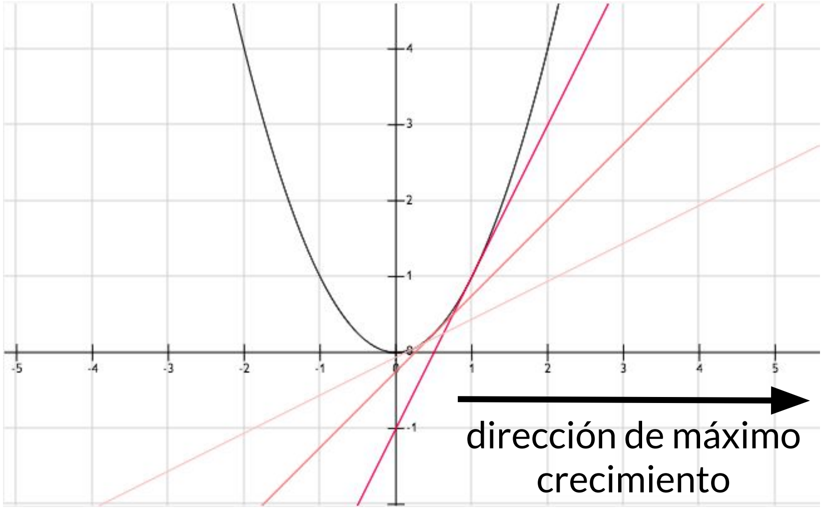
El gradiente es el **vector** de **primeras derivadas** de una función, y denota la **dirección** de **máximo crecimiento** de la función. Es una generalización multivariable de la derivada.

$$\nabla f = \left(\frac{\partial f}{\partial x_1}, \cdots, \frac{\partial f}{\partial x_n}\right)$$

Algunas propiedades:

- Es un operador lineal.
- Se anula en los puntos estacionarios (máximos, mínimos y puntos de silla).

# Idea gráfica




$$f(x) = x^2$$

$$f'(x) = 2x$$

$$f'(1) = 2$$

$$f'(0.5) = 1$$

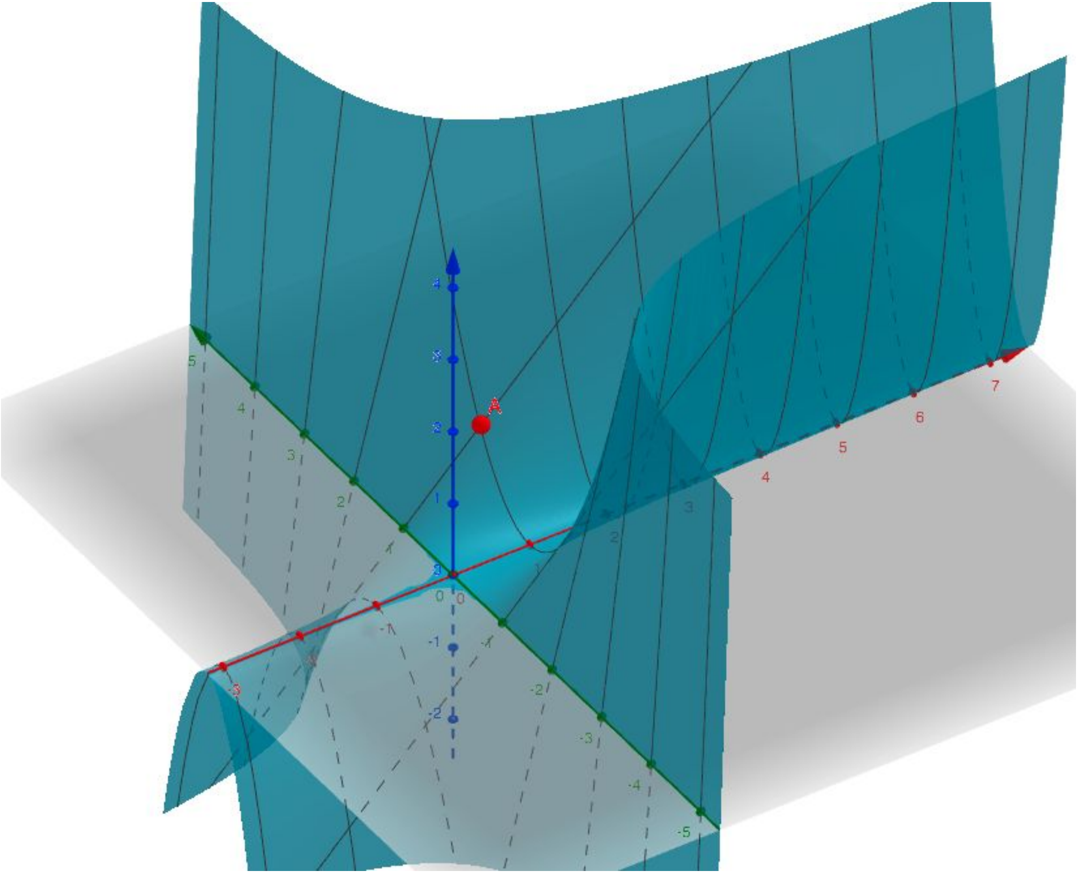
$$f'(0.25) = 0.5$$

$$f(x, y) = xy^2$$

$$\nabla f = (y^2, 2xy)$$

$$f(1, 1) = 1$$

$$\nabla f(1, 1) = (1, 2)$$



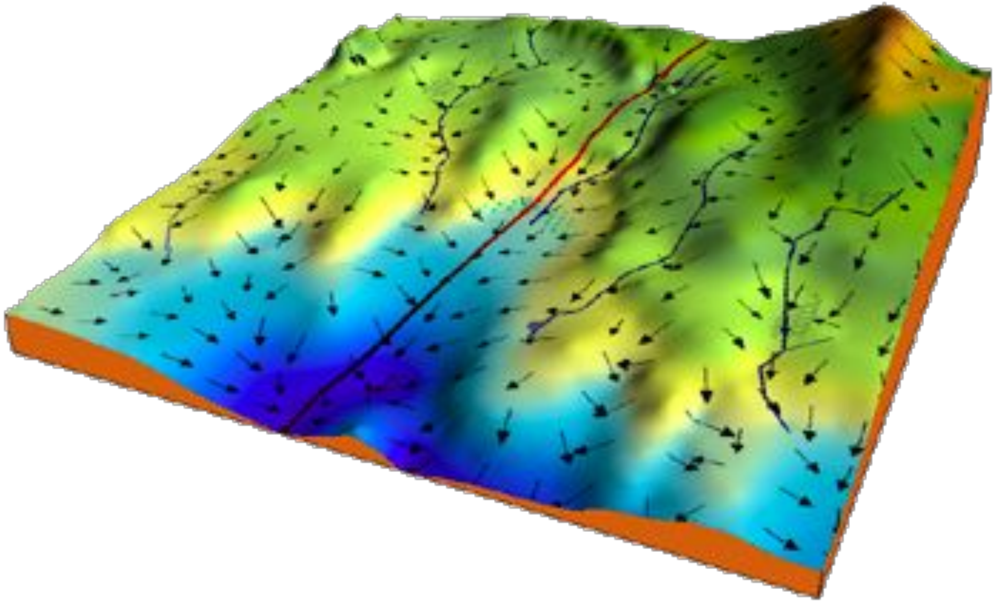
# ¿Qué es el descenso del gradiente?

El descenso del gradiente es un **algoritmo** de optimización utilizado para encontrar, de manera **iterativa**, los **parámetros** que minimizan una función. Lo hace moviéndose en la dirección del **descenso más pronunciado**, la cual se define como el valor **negativo** del **gradiente**.

A tener en cuenta:

- El descenso el gradiente es muy **útil** cuando **no es posible calcular** cuándo la **derivada** de una función **es igual a 0**.
- **Independiente** de la **función** de costo que tomemos, el **descenso** del **gradiente funciona igual**.
- El **gradiente** nos permite tomar **pasos grandes** cuando estamos **lejos** y **pasos pequeños** cuando estamos muy **cerca**.

Representación de una función de costo con dos parámetros.



# Algoritmo de descenso del gradiente

**Paso 1:** Calcular la **derivada** de la **función de coste** para cada **parámetro**, o lo que es lo mismo, calcular el vector gradiente.

**Paso 2:** Elegir un **valor aleatorio** para cada uno de los **parámetros**.

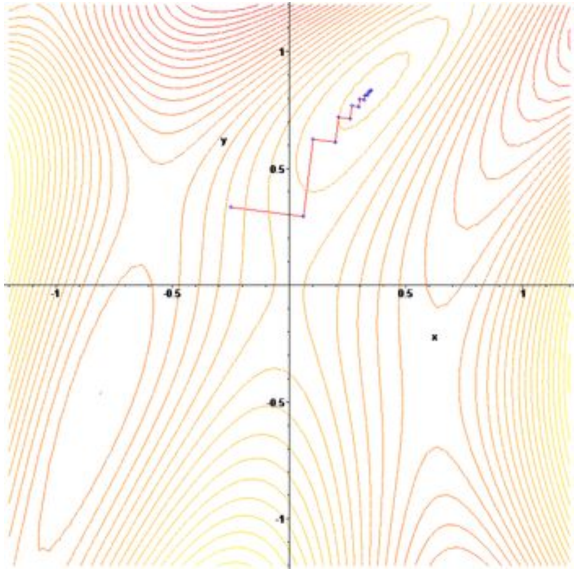
**Paso 3: Meter** los **valores** seleccionados dentro del **gradiente**.

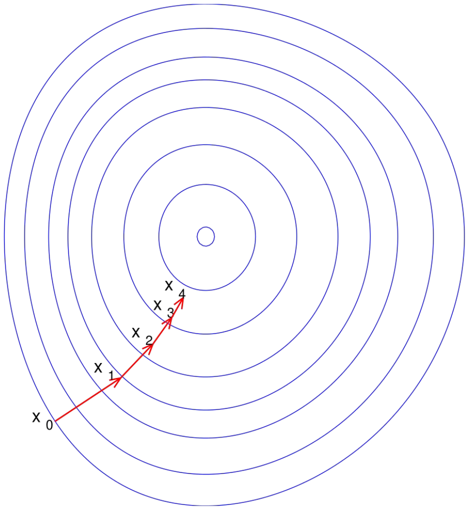
**Paso 4: Calcular** los **pasos**: Paso = Gradiente * Tasa de Aprendizaje.

**Paso 5: Calcular** los **nuevos parámetros**: Nuevos Parámetros = Viejos Parámetros - Paso.

**Paso 6: Repetir** los **pasos 3, 4 y 5** un **número definido** de **iteraciones** o hasta que se cumpla un criterio.

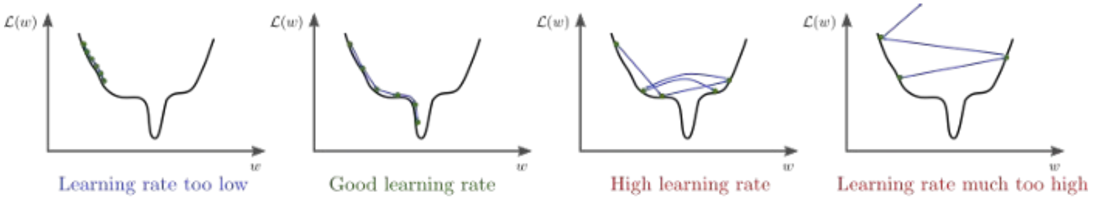
$$a_{n+1} = a_n - \gamma \nabla f(a_n)$$

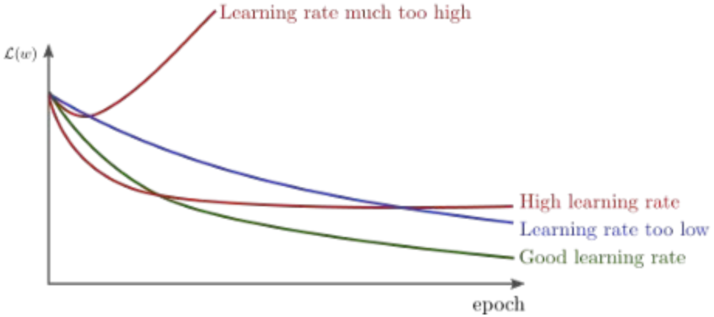


# Selección de la tasa de aprendizaje

La tasa de aprendizaje es un hiperparámetro, y debe ser un valor lo suficientemente pequeño para poder llegar a un mínimo, y lo suficientemente grande para no caer en mínimos locales suboptimos.



# Descenso del Gradiente Estocástico

Problemática:

- Descenso del gradiente es muy costoso computacionalmente cuando se tienen que estimar muchos parámetros.

Solución cuando hay redundancia en los datos (registros muy próximos entre sí) :

- Hacer el descenso del gradiente utilizando solo una observación por iteración.
- Existen muchas adaptaciones: Momentum, RMSProp, Adam
- Otra versión es mini-batch SGD: utilizar un porcentaje de los datos.

# ¿Cómo usarlo en Regresión lineal?

Definimos una función de costo:

$$\left.\begin{aligned} ECM &= \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2 \\ \hat{y}_i &= b_0 + b_1 x1 \end{aligned}\right\} \quad ECM = \frac{1}{n}\sum_{i=1}^{n}(y_i - (b_0 + b_1 x_1))^2$$

Calculamos sus derivadas parciales:

$$\frac{\partial f}{\partial b_0} = \frac{-2}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)$$

$$\frac{\partial f}{\partial b_1} = \frac{-2}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)x_1$$

# ¿Cómo usarlo en regresión logística?

- Definimos una función de costo:

$$ECM = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2$$

$$\hat{y}_i = \frac{1}{1 + e^{-(b_0 + b_1 x_1 + b_2 x_2)}}$$

Calculamos sus derivadas parciales:

$$\frac{\partial f}{\partial b_0} = \frac{-2}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)\hat{y}_i(1 - \hat{y}_i)$$

$$\frac{\partial f}{\partial b_1} = \frac{-2}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)\hat{y}_i(1 - \hat{y}_i)x_1$$

$$\frac{\partial f}{\partial b_2} = \frac{-2}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)\hat{y}_i(1 - \hat{y}_i)x_2$$

# Optimizadores de SKLEARN para regresión logística

https://scikit-learn.org/stable/modules/linear_model.html#logistic-regression

# ¡Gracias!

# Enlaces

- https://github.com/diegoib/newton_logistic/blob/master/main.py (newton-raphson para regresión logística)
- https://machinelearningmastery.com/gradient-descent-for-machine-learning/ (descenos del gradiente)
- https://ruder.io/optimizing-gradient-descent/ (optimizadores)
- https://ml-cheatsheet.readthedocs.io/en/latest/gradient_descent.html (descenso del gradiente)
- https://dphi.tech/blog/tutorial-on-logistic-regression-using-python/#video-logistic-regression (gradiente con regresión logística y lineal)
- https://towardsdatascience.com/a-gentle-introduction-to-maximum-likelihood-estimation-9fbff27ea12f (estimacion de maxima verosimilitud - para regresión logística)
- video de statquest
- https://towardsdatascience.com/common-loss-functions-in-machine-learning-46af0ffc4d23 (funciones de costo)
- https://heartbeat.fritz.ai/5-regression-loss-functions-all-machine-learners-should-know-4fb140e