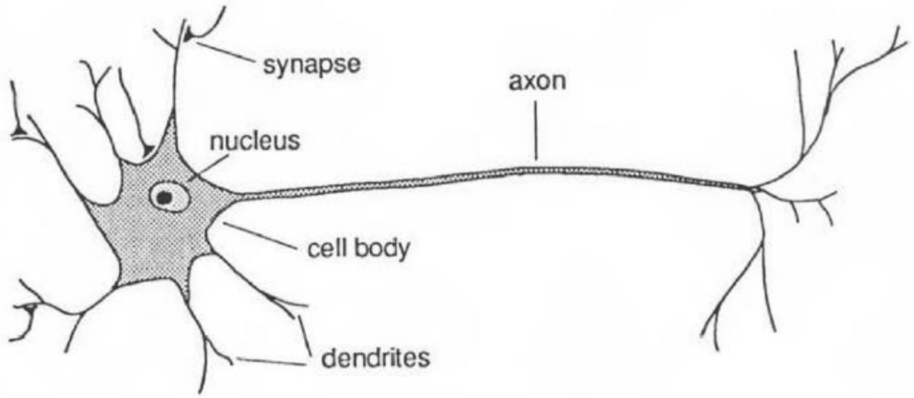
INTRODUCCIÓN AL APRENDIZAJE AUTOMÁTICO

APRENDIZAJE PARA LA REGRESIÓN – EL DESCENSO POR EL GRADIENTE

LAURA DIAZ DÁVILA – FRANCISCO TAMARIT



REPRESENTACIÓN DE LA INFORMACIÓN

REGRESIÓN

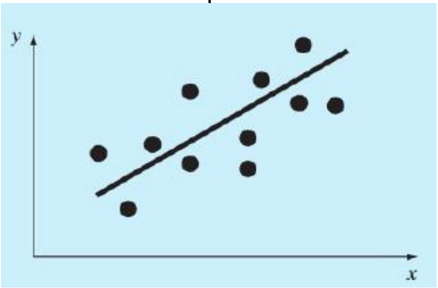
CLASIFICACIÓN

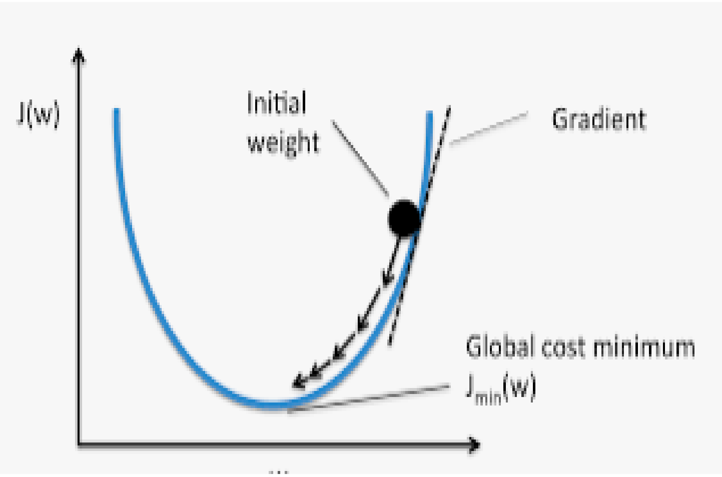
DE REGRESIÓN LINEAL A "DESCENSO POR EL GRADIENTE" EN IA

- **Análisis de la tendencia**: Predecir valores de la variable dependiente –interpolar o extrapolar-.
- **Prueba de hipótesis**: Validar un modelo matemático existente con los resultados experimentales o adecuar el modelo a los datos.

Regresión por mínimos cuadrados

$$\boldsymbol{y = a_0 + a_1\ x}$$





Minimizar la suma de los cuadrados de los errores:

$$S_r = \sum_{i=1}^{n} e_i^2 = \sum_{i=1}^{n} (y_{i,\text{medida}} - y_{i,\text{modelo}})^2 = \sum_{i=1}^{n} (y_i - a_0 - a_1 x_i)^2$$

$$\frac{\partial S_r}{\partial a_0} = -2\sum (y_i - a_0 - a_1 x_i)$$

$$\frac{\partial S_r}{\partial a_1} = -2\sum [(y_i - a_0 - a_1 x_i)x_i]$$

$$n \quad a_0 + \left(\sum x_i\right) a_1 = \sum y_i$$

$$\left(\sum x_i\right) a_0 + \left(\sum x_i^2\right) a_1 = \sum x_i y_i$$

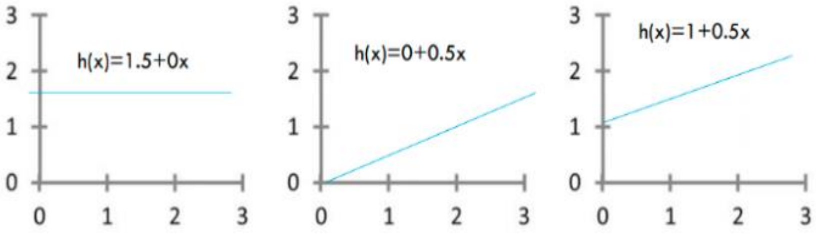
PARÁMETROS

$$a_1 = \frac{n\sum x_i y_i - \sum x_i \sum y_i}{n\sum x_i^2 - \left(\sum x_i\right)^2}$$

$$a_0 = \bar{y} - a_1 \bar{x}$$

EL DESCENSO POR EL GRADIENTE

$h_\theta(x) = \theta_0 + \theta_1 x$



$\theta_0 = 1.5$
$\theta_1 = 0$

$\theta_0 = 0$
$\theta_1 = 0.5$

$\theta_0 = 1$
$\theta_1 = 0.5$

Hypothesis:

$$h_\theta(x) = \theta_0 + \theta_1 x$$

Parameters:

$$\theta_0, \theta_1$$

Cost Function:

$$J(\theta_0, \theta_1) = \frac{1}{2m} \sum_{i=1}^{m} \left(h_\theta(x^{(i)}) - y^{(i)}\right)^2$$

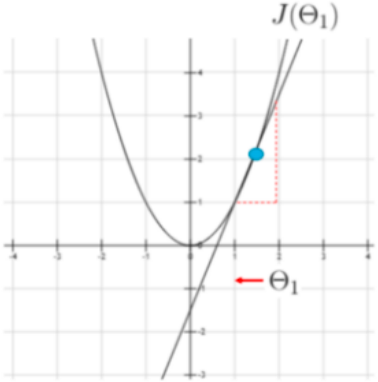
Goal: $\underset{\theta_0, \theta_1}{\text{minimize}}\ J(\theta_0, \theta_1)$

$J(\Theta_1) \quad \Theta_1 \in \mathbb{R}$

$$min\ J(\Theta_1) := \Theta_1 - \alpha \frac{\partial}{\partial \Theta_1} J(\Theta_1)$$

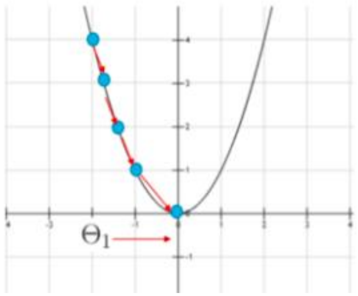
Positive Slope $\frac{\partial}{\partial \Theta_1} J(\Theta_1) \geqslant 0$

$\Theta_1 - \alpha(positive, number)$

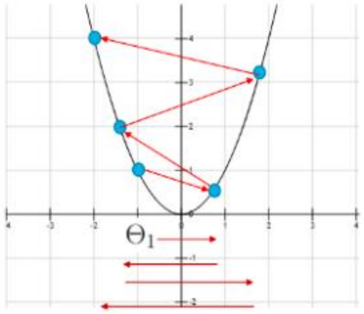


$$\theta_1 := \theta_1 - \alpha \frac{\partial}{\partial \theta_1} J(\theta_1)$$

If α is too small, gradient descent can be slow.



If α is too large, gradient descent can overshoot the minimum. It may fail to converge, or even diverge.



EL ALGORITMO DEL DESCENSO POR EL GRADIENTE

$$\frac{\partial}{\partial \Theta_j} J(\Theta_0, \Theta_1) := \frac{\partial}{\partial \Theta_j} \cdot \frac{1}{2m} \sum_{i=1}^{n} (h_\Theta(x^{(i)}) - y^{(i)})^2$$

$$\frac{\partial}{\partial \Theta_j} J(\Theta_0, \Theta_1) := \frac{\partial}{\partial \Theta_j} \cdot \frac{1}{2m} \sum_{i=1}^{n} (\Theta_0 + \Theta_1 x^{(i)} - y^{(i)})^2$$

$$\frac{\partial}{\partial \Theta_0} J(\Theta_0, \Theta_1) = \frac{1}{m} \sum_{i=1}^{n} (h_\Theta(x^{(i)}) - y^{(i)})^2$$

$$\frac{\partial}{\partial \Theta_1} J(\Theta_0, \Theta_1) = \frac{1}{m} \sum_{i=1}^{n} (h_\Theta(x^{(i)}) - y^{(i)})^2 \cdot\ x^{(i)}$$

Gradient descent algorithm

repeat until convergence {

$$\theta_0 := \theta_0 - \alpha \boxed{\frac{1}{m} \sum_{i=1}^{m} \left(h_\theta(x^{(i)}) - y^{(i)}\right)}$$

$$\theta_1 := \theta_1 - \alpha \boxed{\frac{1}{m} \sum_{i=1}^{m} \left(h_\theta(x^{(i)}) - y^{(i)}\right) \cdot x^{(i)}}$$

}

update θ_0 and θ_1 simultaneously

$$\frac{\partial}{\partial \Theta_0} J(\Theta_0, \Theta_1)$$

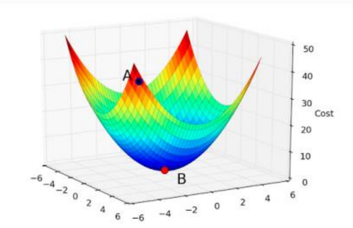
$$\frac{\partial}{\partial \Theta_1} J(\Theta_0, \Theta_1)$$

DE REGRESIÓN LINEAL A “DESCENSO POR EL GRADIENTE” EN IA

- **Análisis de la tendencia**: Predecir valores de la variable dependiente –interpolar o extrapolar-.
- **Prueba de hipótesis**: Validar un modelo matemático existente con los resultados experimentales o adecuar el modelo a los datos.

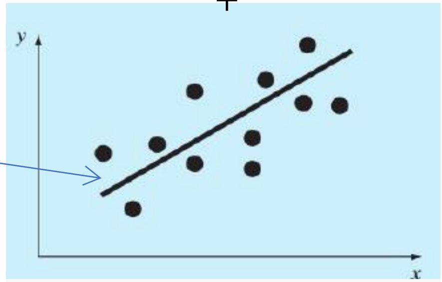
Regresión por mínimos cuadrados

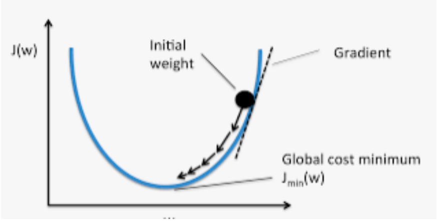
FUNCIÓN DE COSTE (LOSS FUNCTION)



$$\boldsymbol{y = a_0 + a_1 \, x}$$

FUNCIÓN DE ACTIVACIÓN





Minimizar la suma de los cuadrados de los errores:

$$S_r = \sum_{i=1}^{n} e_i^2 = \sum_{i=1}^{n} (y_{i,\text{medida}} - y_{i,\text{modelo}})^2 = \sum_{i=1}^{n} (y_i - a_0 - a_1 x_i)^2$$

$$\frac{\partial S_r}{\partial a_0} = -2 \sum (y_i - a_0 - a_1 x_i)$$

$$\frac{\partial S_r}{\partial a_1} = -2 \sum [(y_i - a_0 - a_1 x_i) x_i]$$

DESCENSO POR EL GRADIENTE

$$n \quad a_0 + \left(\sum x_i\right) a_1 = \sum y_i$$

$$\left(\sum x_i\right) a_0 + \left(\sum x_i^2\right) a_1 = \sum x_i y_i$$

PARÁMETROS O PESOS SINÁPTICOS

$$a_1 = \frac{n \sum x_i y_i - \sum x_i \sum y_i}{n \sum x_i^2 - \left(\sum x_i\right)^2}$$

$$a_0 = \bar{y} - a_1 \bar{x}$$

EL DESCENSO POR EL GRADIENTE EN PYTHON

LIBRERÍAS DE PYTHON:

NUMPY

SCIPY

MATPLOTLIB

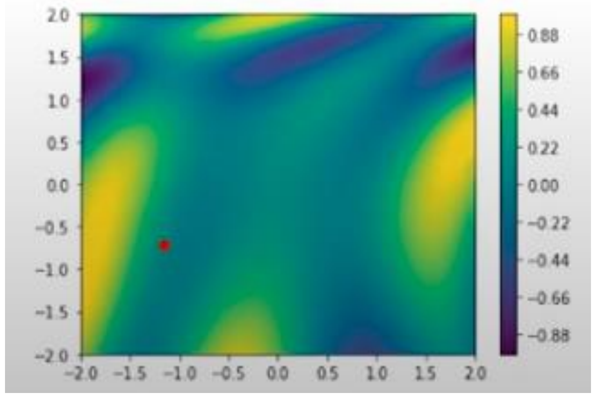
```python
def coste(x, y, a, b):
    m = len(x)
    error = 0.0
    for i in range(m):
        hipotesis = a+b*x[i]
        error +=  (y[i] - hipotesis) ** 2
    return error / (2*m)
```

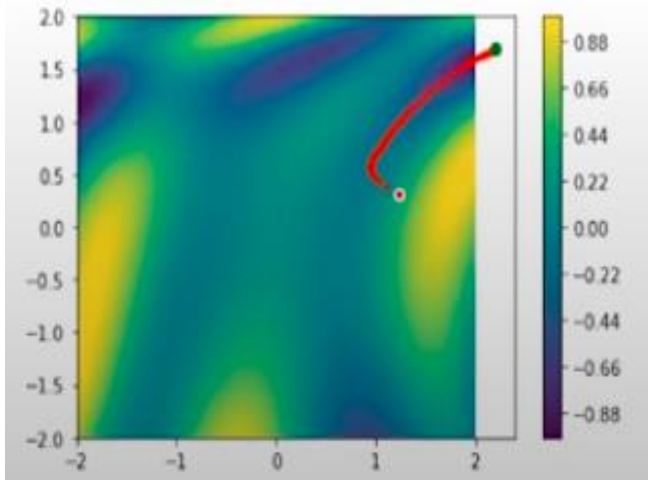
FUNCIÓN DE COSTE (LOSS FUNCTION)

```python
def descenso_gradiente(x, y, a, b, alpha, epochs):
    m = len(x)
    hist_coste = []
    for ep in range(epochs):
        b_deriv = 0
        a_deriv = 0
        for i in range(m):
            hipotesis = a+b*x[i]
            a_deriv += hipotesis - y[i]
            b_deriv += (hipotesis - y[i]) * x[i]
            hist_coste.append(coste(x, y, a, b))
        a -= (a_deriv / m) * alpha
        b -= (b_deriv / m) * alpha

    return a, b, hist_coste
```

DESCENSO POR EL GRADIENTE

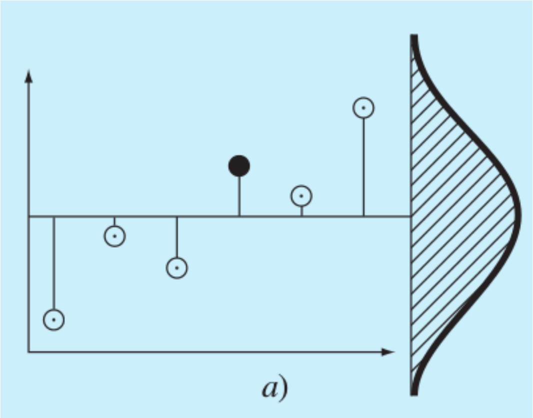


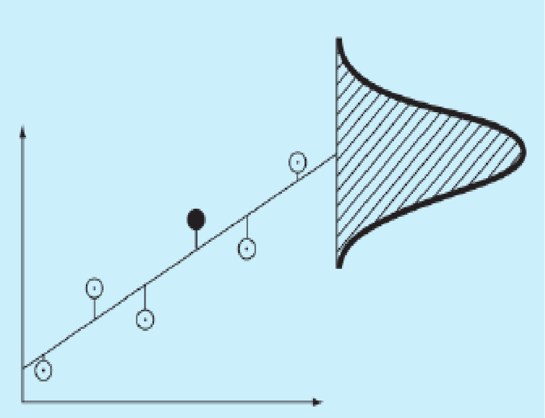


https://www.youtube.com/watch?v=-_A_AAxqzCg

REGRESIÓN LINEAL MÚLTIPLE

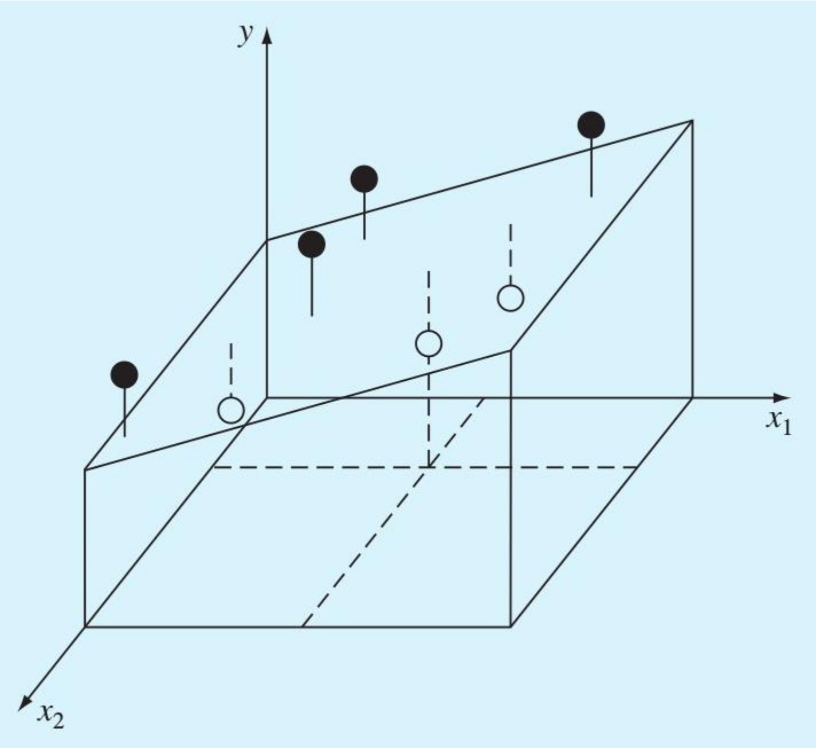
$S_{y/x}$





REGRESIÓN LINEAL MÚLTIPLE:

* VARIABLES PREDICTORAS
* MULTICOLINEALIDAD

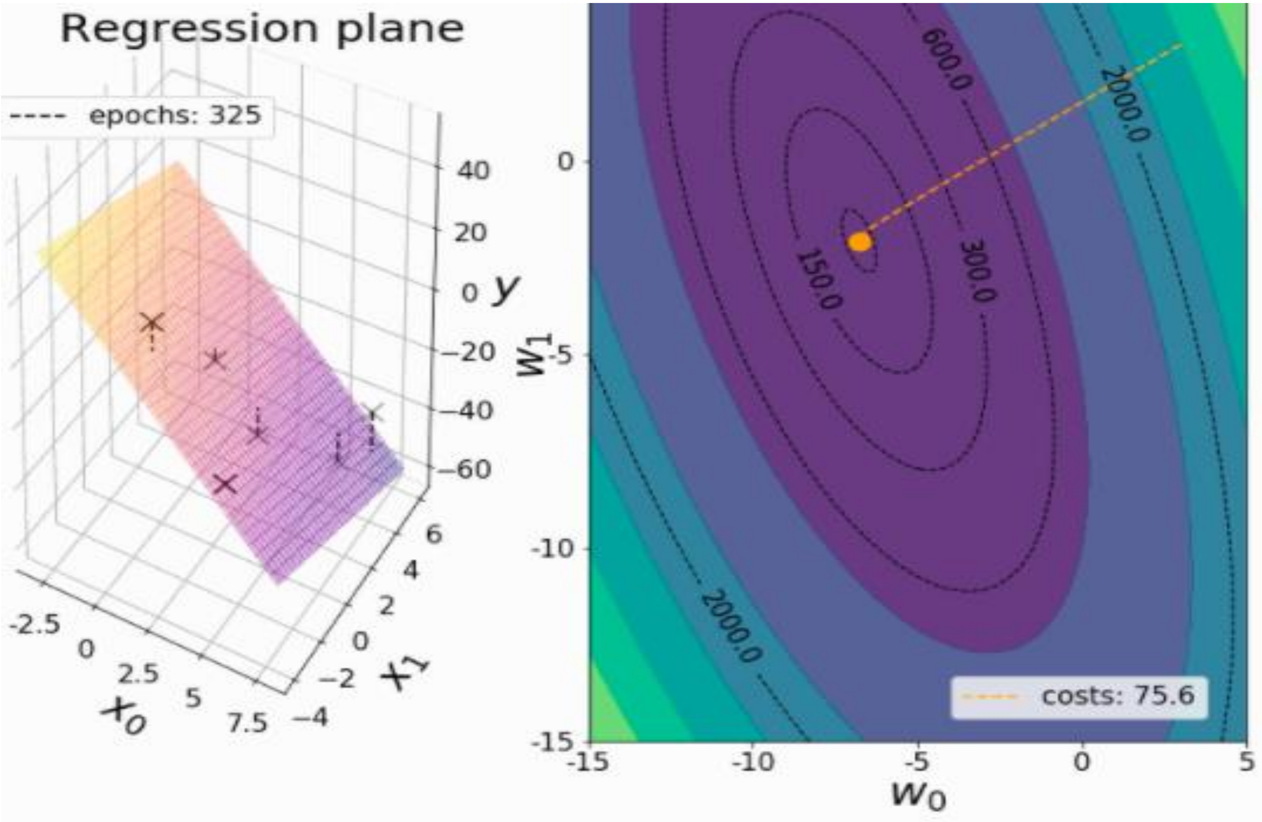


FUNCIÓN DE COSTE PARA LA REGRESIÓN MULTILINEAL:

$$S_r = \sum_{i=1}^{n} (y_i - a_0 - a_1 x_{1i} - a_2 x_{2i})^2$$

REGRESIÓN LINEAL MÚLTIPLE

EL DESCENSO POR EL GRADIENTE



https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.LinearRegression.html#

¡Gracias!