Regresión lineal univariada. Parte 2

Pontificia Universidad Javeriana
Francisco Carlos Calderon Ph.D
2020

Facultad de ingeniería
Depto de electrónica
Pontificia Universidad JAVERIANA Bogotá

Objetivos

Tomar la función de costo ya implementada y obtener un mejor algoritmo que permita encontrar su mínimo de manera más precisa y sin requerir una iteración de parámetros extensa.

Método de descenso de gradiente

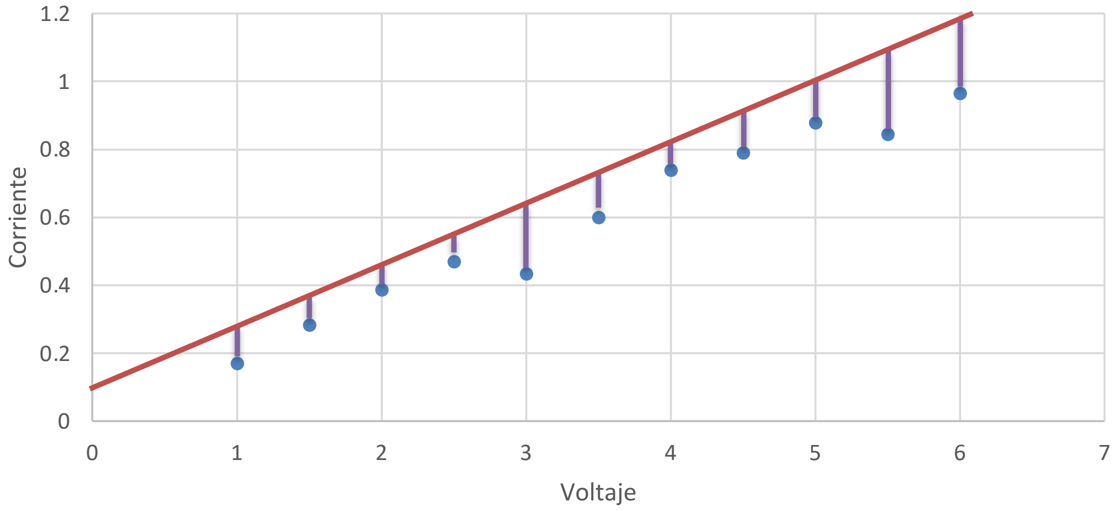
- En inglés Gradient descent.
- <u>*No es gradiente descendiente*</u>.
- Puede ser aplicable a problemas de múltiples variables.
- Es un método iterativo.
- permite encontrar un mínimo local.
- Se le atribuye a Augustin-Louis Cauchy en 1847

Contexto:

Definimos una función de costo $\mathrm{J}(\theta_0, \theta_1)$.

IDEA: Definir y evaluar los (θ_0, θ_1) de tal manera que se minimize alguna métrica de error que ***tenga sentido en el problema***.

Gráfico de Voltaje contra corriente



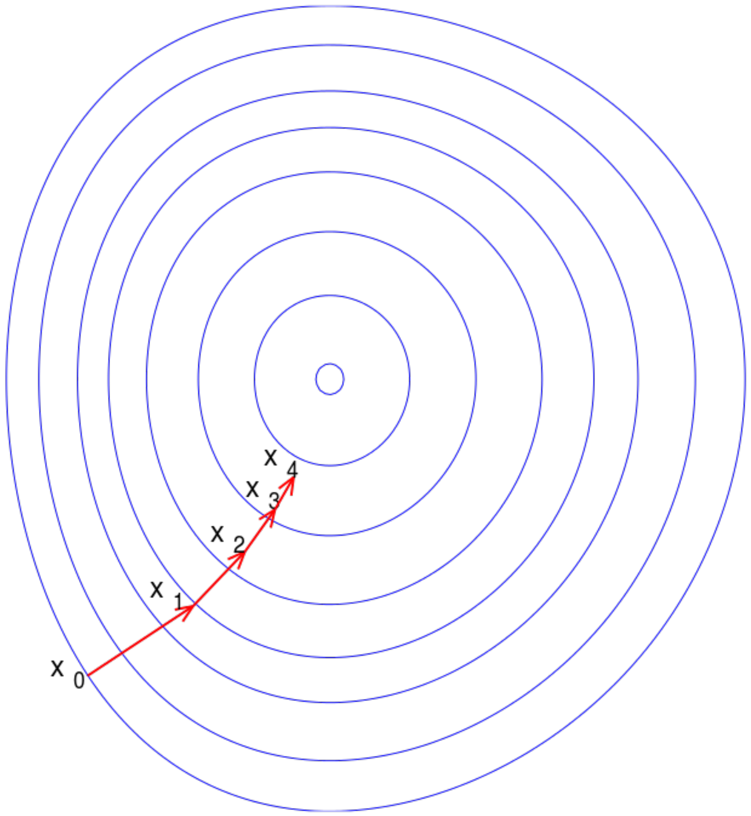
$$h_\theta(x) = \theta_0 + \theta_1 x = y$$

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

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

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

Contexto:



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

Partir de un punto cualquiera *y encontrar una dirección en la que se garantice el descenso*.

Evaluar un nuevo punto y repetir hasta encontrar:

1. Un número de iteraciones máximo.
2. Un cambio en J menor a un mínimo de error aceptable.
    - No hay cambios en J.
    - El cambio es "*muy*" pequeño.

Definición formal:

Si la función de múltiples variables $F(\boldsymbol{x})$ es diferenciable en la vecindad del punto $\boldsymbol{a}$, entonces $F(\boldsymbol{x})$ decrece más rápidamente si se parte desde $\boldsymbol{a}$ hacia una dirección dada por el gradiente negativo de F en $\boldsymbol{a}$, $-\boldsymbol{\nabla} F(\boldsymbol{a})$.

$$\boldsymbol{a}_{n+1} = \boldsymbol{a}_n - \alpha \boldsymbol{\nabla} F(\boldsymbol{a}_n)$$

Para algunos $\alpha \in \mathbb{R}$ y pequeños, $F(\boldsymbol{a}_n) \geq F(\boldsymbol{a}_{n+1})$.

Para nuestro problema

Para minimizar:

$$\min_{\theta_0,\theta_1} \mathrm{J}(\theta_0, \theta_1)$$

Repetir hasta encontrar convergencia:

$$\theta_j^{(n+1)} = \theta_j^{(n)} - \alpha \frac{\partial}{\partial \theta_j} \mathrm{J}(\theta_0, \theta_1)$$

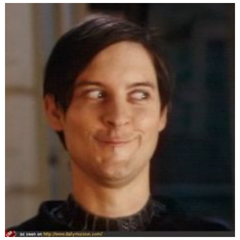
Consideraciones en la evaluación

No todas las implementaciones son iguales:

Los dos ejes al tiempo:

$$\theta_0^{(n+1)} = \theta_0^{(n)} - \alpha \frac{\partial}{\partial \theta_0} \mathrm{J}(\theta_0^{(n)}, \theta_1^{(n)})$$

$$\theta_1^{(n+1)} = \theta_1^{(n)} - \alpha \frac{\partial}{\partial \theta_1} \mathrm{J}(\theta_0^{(n)}, \theta_1^{(n)})$$

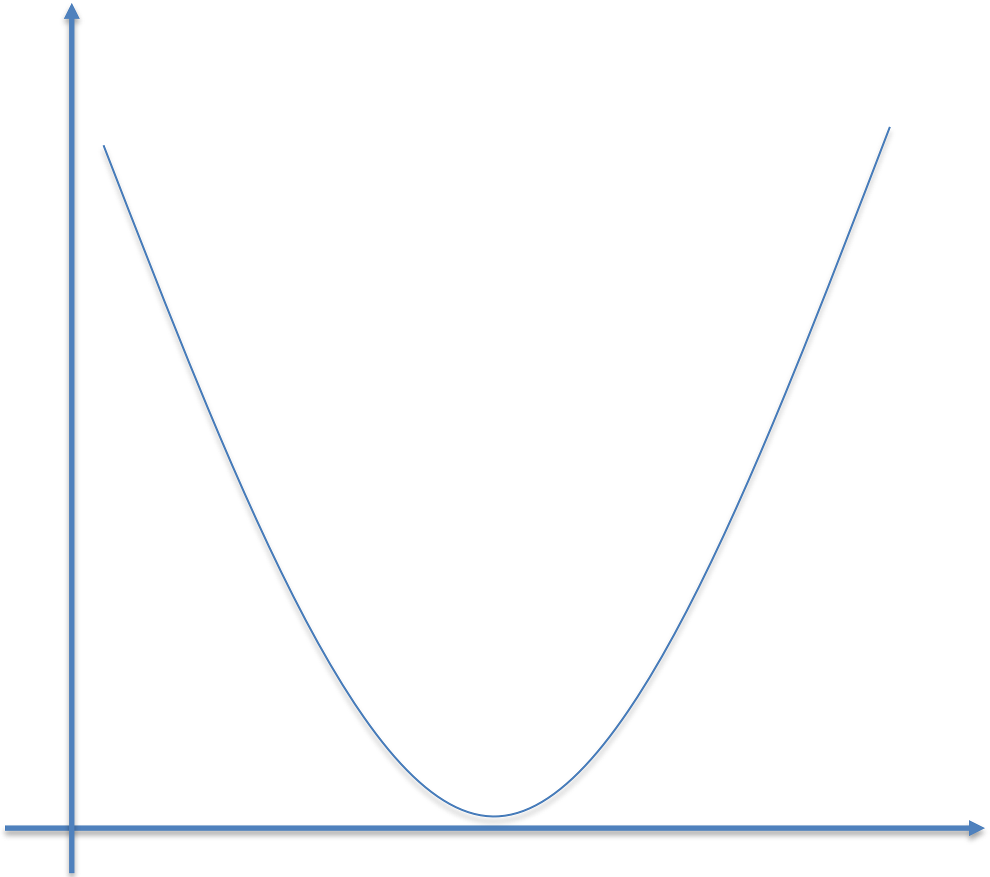


Primero uno y luego el otro:

$$\theta_0^{(n+1)} = \theta_0^{(n)} - \alpha \frac{\partial}{\partial \theta_0} \mathrm{J}(\theta_0^{(n)}, \theta_1^{(n)})$$

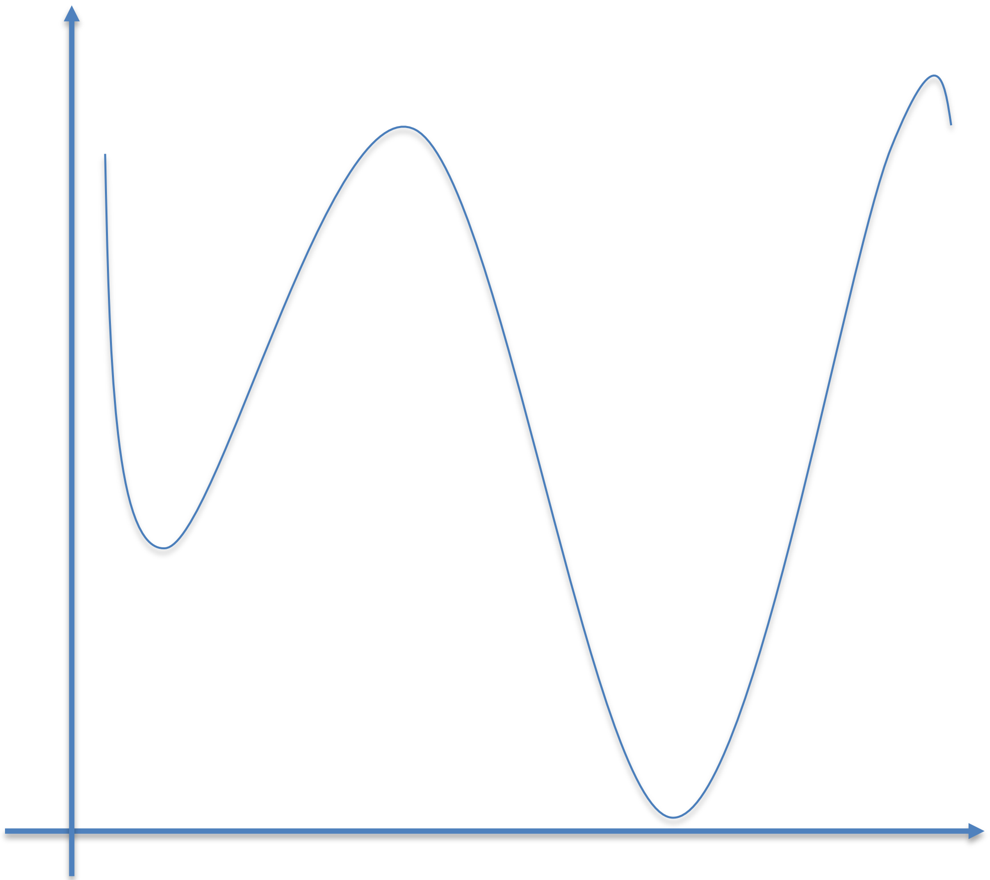
$$\theta_1^{(n+1)} = \theta_1^{(n)} - \alpha \frac{\partial}{\partial \theta_1} \mathrm{J}(\theta_0^{(n+1)}, \theta_1^{(n)})$$

Selección del α



- Si es muy pequeño va a ser lento.
- Si es muy grande va a divergir.
- Se puede evaluar el α en la marcha y dejarse fijo.

Mínimos locales o globales



- No se garantiza el mínimo global, solo uno local pero solo si se tienen α adecuados.
- Que pasa si se inicia en un máximo!

Para nuestro problema:

$$\theta_0^{(k+1)} = \theta_0^{(k)} - \alpha \frac{\partial}{\partial \theta_0} \mathrm{J}(\theta_0^{(k)}, \theta_1^{(k)})$$

$$\theta_1^{(k+1)} = \theta_1^{(k)} - \alpha \frac{\partial}{\partial \theta_1} \mathrm{J}(\theta_0^{(k)}, \theta_1^{(k)})$$

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

$$h_{\theta^k}(x) = \theta_0^k + \theta_1^k x = y$$

$$\theta_0^{(k+1)} = \theta_0^{(k)} - \alpha \frac{1}{n} \sum_{i=1}^{n} (h_{\theta^k}\left(x^{(i)}\right) - y^{(i)})$$

$$\theta_1^{(k+1)} = \theta_1^{(k)} - \alpha \frac{1}{n} \sum_{i=1}^{n} (h_{\theta^k}\left(x^{(i)}\right) - y^{(i)})\ x^{(i)}$$

Ejercicio para el resto de la clase

1. Implemente el par de funciones de gradiente del costo anterior teniendo 3 entradas cada una:
    1. Un vector “nparray” para los parámetros actuales (θ_0^k, θ_1^k)
    2. Un vector “nparray” para las características “x”
    3. Un vector “nparray” para las etiquetas “y”

   La salida de cada una es un flotante.

https://numpy.org/doc/stable/reference/generated/numpy.array.html

Ejercicio para el resto de la clase

2. Use el par de funciones de gradiente de costo, para implementar otra función que cree un paso del algoritmo de descenso del gradiente

$$\theta_0^{(k+1)} = \theta_0^{(k)} - \alpha \frac{1}{n} \sum_{i=1}^{n} (h_{\theta^k}(x^{(i)}) - y^{(i)})$$

$$\theta_1^{(k+1)} = \theta_1^{(k)} - \alpha \frac{1}{n} \sum_{i=1}^{n} (h_{\theta^k}(x^{(i)}) - y^{(i)})\ x^{(i)}$$

$$h_{\theta^k}(x) = \theta_0^k + \theta_1^k x$$