Regresión

Verónica E. Arriola-Rios
(Basado en el curso de Andrew NG)

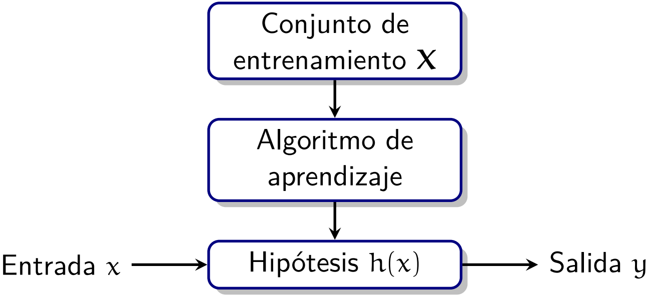
Aprendizaje de máquina

14 de octubre de 2020

Temas

1. Regresión
   - Regresión lineal univariada
   - Regresión lineal multivariada
   - Regresión polinomial
2. Descenso por el grandiente
3. Regularización

Regresión



$$h(x) : \mathbb{R}^n \to \mathbb{R} \tag{1}$$

Temas

1. Regresión
   - Regresión lineal univariada
   - Regresión lineal multivariada
   - Regresión polinomial
2. Descenso por el grandiente
3. Regularización

Regresión lineal

	$x[m^2]$	$y[MX]$
1	90	\$2´000,000
2	200	\$5000,000
3	320	\$4800,000
4	325	\$7700,000
m	...	

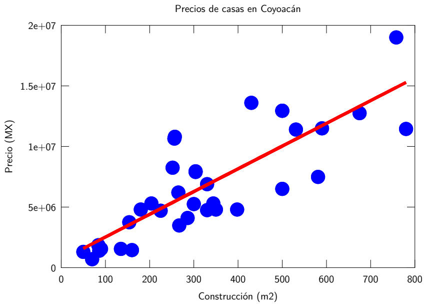


Figura: Se ajusta una recta (*modelo lineal*) a los datos experimentales.

Regresión lineal

$$
\begin{aligned}
m &= \text{número de ejemplares para el entrenamiento.}\\
x &= \text{variable de entrada (características).}\\
y &= \text{variable de salida (objetivo).}\\
(x, y) &= \text{un ejemplar.}\\
(x^{(i)}, y^{(i)}) &= \text{i-ésimo ejemplar.}
\end{aligned}
$$

Hipótesis:

$$h_\theta(x) = \theta_0 + \theta_1 x \tag{2}$$

$$h(x) : \mathbb{R} \to \mathbb{R}$$

$$\theta_i = \text{parámetro del modelo.}$$

El *espacio de hipótesis* es el conjunto de rectas en el espacio $\mathbb{R}^2$.

Aprendizaje

Problema:

- Encontrar los θ_i que permiten que $h(x)$ prediga lo mejor posible los valores de y.

Hipótesis:

- Suponemos que si $h(x^{(i)}) \approx y^{(i)}$ entonces $h(x) \approx y$ para valores de x y y no vistos anteriormente.

Estrategia:

- Minimizar el error que comete $h(x)$ al predecir el valor de y.

Función de costo

$$J(\theta_0, \theta_1) = \frac{1}{2m} \sum_{i=1}^{m} (h(x^{(i)}) - y^{(i)})^2 \tag{3}$$

- Penaliza tanto las sobreestimaciones como las subestimaciones.

Objetivo:

- Minimizar $J(\theta_0, \theta_1)$ variando los valores de θ_0 y θ_1 y eligiendo aquellos que producen el menor valor de $J(\theta_0, \theta_1)$.

Ejemplo

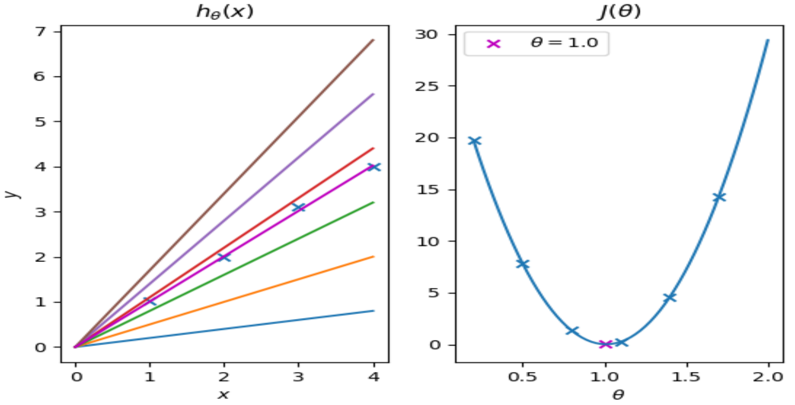
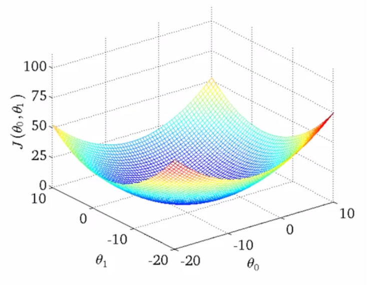


Figura: Izquierda: $h_\theta(x) = \theta x$. Derecha: $J(\theta) = \frac{1}{2m}\sum_{i=1}^{m}(\theta x^{(i)} - y^{(i)})^2$

Ejemplo



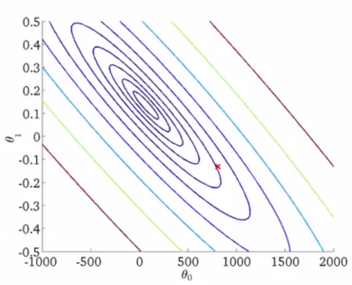


Figura: $J(\theta_0, \theta_1) = \frac{1}{2m} \sum_{i=1}^{m} (h(x^{(i)}) - y^{(i)})^2$

Regresión lineal univariada

También conocida como *ajuste de rectas por mínimos cuadrados*.

$$h_\theta(x) = \theta_0 + \theta_1 x \tag{4}$$

$$\min_{\theta_0,\theta_1} J(\theta_0, \theta_1) = \frac{1}{2m} \sum_{i=1}^{m} (\theta_0 + \theta_1 x^{(i)} - y^{(i)})^2 \tag{5}$$

$$\nabla J_\Theta = \begin{bmatrix} \frac{\partial J}{\partial \theta_0} \\ \frac{\partial J}{\partial \theta_1} \end{bmatrix} \tag{6}$$

$$\frac{\partial J}{\partial \theta_0} = \frac{1}{m} \sum_{i=1}^{m} (\theta_0 + \theta_1 x^{(i)} - y^{(i)}) = 0 \tag{7}$$

$$\frac{\partial J}{\partial \theta_1} = \frac{1}{m} \sum_{i=1}^{m} (\theta_0 + \theta_1 x^{(i)} - y^{(i)}) x^{(i)} = 0 \tag{8}$$

Despejando de (7):

$$\theta_0 \sum_{i=1}^{m} 1 + \theta_1 \sum_{i=1}^{m} x^{(i)} - \sum_{i=1}^{m} y^{(i)} = 0 \tag{9}$$

$$\theta_0 = \frac{1}{m} \left(\sum_{i=1}^{m} y^{(i)} - \theta_1 \sum_{i=1}^{m} x^{(i)} \right) \tag{10}$$

Separando términos en (8) y sustituyendo (10):

$$\theta_0 \sum_{i=1}^{m} x^{(i)} + \theta_1 \sum_{i=1}^{m} (x^{(i)})^2 - \sum_{i=1}^{m} y^{(i)} x^{(i)} = 0 \tag{11}$$

$$\frac{1}{m} \left(\sum_{i=1}^{m} y^{(i)} - \theta_1 \sum_{i=1}^{m} x^{(i)} \right) \sum_{i=1}^{m} x^{(i)} + \theta_1 \sum_{i=1}^{m} (x^{(i)})^2 - \sum_{i=1}^{m} y^{(i)} x^{(i)} = 0 \tag{12}$$

$$\frac{1}{m}\sum_{i=1}^{m} y^{(i)} \sum_{i=1}^{m} x^{(i)} - \theta_1 \frac{1}{m}\sum_{i=1}^{m} x^{(i)} \sum_{i=1}^{m} x^{(i)} + \theta_1 \sum_{i=1}^{m} (x^{(i)})^2 - \sum_{i=1}^{m} y^{(i)} x^{(i)} = 0 \tag{13}$$

$$\theta_1 = \frac{\sum_{i=1}^{m} y^{(i)} x^{(i)} - \frac{1}{m}\sum_{i=1}^{m} y^{(i)} \sum_{i=1}^{m} x^{(i)}}{\sum_{i=1}^{m} (x^{(i)})^2 - \frac{1}{m}\left(\sum_{i=1}^{m} x^{(i)}\right)^2} \tag{14}$$

$$\theta_1 = \frac{m\sum_{i=1}^{m} x^{(i)} y^{(i)} - \sum_{i=1}^{m} x^{(i)} \sum_{i=1}^{m} y^{(i)}}{m\sum_{i=1}^{m} (x^{(i)})^2 - \left(\sum_{i=1}^{m} x^{(i)}\right)^2} \tag{15}$$

Se deja como ejercicio despejar θ_0. Se obtiene:

$$\theta_0 = \frac{\sum_{i=1}^{m} (x^{(i)})^2 \sum_{i=1}^{m} y^{(i)} - \sum_{i=1}^{m} x^{(i)} \sum_{i=1}^{m} x^{(i)} y^{(i)}}{m\sum_{i=1}^{m} (x^{(i)})^2 - \left(\sum_{i=1}^{m} x^{(i)}\right)^2} \tag{16}$$

Temas

1. Regresión
   - Regresión lineal univariada
   - Regresión lineal multivariada
   - Regresión polinomial
2. Descenso por el grandiente
3. Regularización

Regresión lineal multivariada

$$
\begin{aligned}
x_1 &= \text{Tamaño en } m^2 \\
x_2 &= \text{Recámaras} \\
x_3 &= \text{Antigüedad en años} \\
x_4 &= \text{Baños} \\
x_5 &= \text{Estacionamientos} \\
Y &= \text{Precio } (\$ \times 1000)
\end{aligned}
$$

1 (Tam)	2 (Rec)	3 (Ant)	4 (Baños)	5 (Est)	Y
121	3	0	3	2	3748
100.55	2	0	2	0	3112
62.5	2	0	2	1	1918
51.75	1	0	1	1	1580
...					

Regresión lineal multivariada

$$
\begin{aligned}
n &= \text{número de características (variables de entrada).} \\
x^{(i)} &= \text{entradas en el i-ésimo ejemplar, } x^{(i)} \in \mathbb{R}^n. \\
x_j^{(i)} &= \text{valor de la j-ésima variable de entrada en el i-ésimo ejemplar.}
\end{aligned}
$$

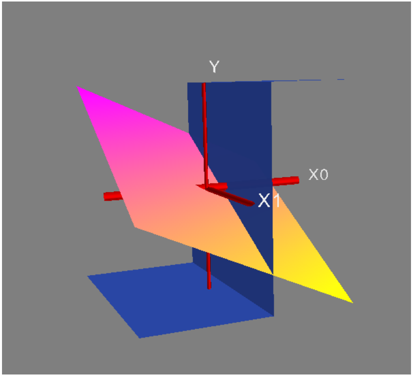
Hipótesis:

$$h_\theta(x) = \theta_0 x_0 + \theta_1 x_1 + \theta_2 x_2 + \ldots + \theta_n x_n \tag{17}$$

con $x_0 \equiv 1$.
El *espacio de hipótesis* es el conjunto de planos $\mathbb{R}^n$ en el espacio $\mathbb{R}^{n+1}$

Interpretación para una hipótesis lineal

$$h(X) = \theta_0 x_0 + \theta_1 x_1$$
$$\mathbf{X} \in \mathbb{R}^2$$



Vectorización

$$h_\theta(x) = \theta_0 x_0 + \theta_1 x_1 + \theta_2 x_2 + \ldots + \theta_n x_n \tag{18}$$

Sea:

$$X = \begin{bmatrix} x_0 \\ x_1 \\ \ldots \\ x_n \end{bmatrix} \qquad \Theta = \begin{bmatrix} \theta_0 \\ \theta_1 \\ \ldots \\ \theta_n \end{bmatrix} \tag{19}$$

Entonces:

$$h_\theta(x) = \Theta^T X = X^T \Theta \tag{20}$$

Vectorización II

Consideremos ahora a todos los ejemplares de entrenamiento, formando a la matriz $\mathbf{X}$, con cada renglón igual a X^T, y las salidas Y para cada ejemplar en un vector.

$$\mathbf{X} = \begin{bmatrix} (X^{(1)})^T \\ (X^{(2)})^T \\ \dots \\ (X^{(m)})^T \end{bmatrix} = \begin{bmatrix} x_0^{(1)} x_1^{(1)} \dots x_n^{(1)} \\ x_0^{(2)} x_1^{(2)} \dots x_n^{(2)} \\ \dots \\ x_0^{(m)} x_1^{(m)} \dots x_n^{(m)} \end{bmatrix} \qquad Y = \begin{bmatrix} y_0 \\ y_1 \\ \dots \\ y_n \end{bmatrix} \tag{21}$$

Entonces la hipótesis, evaluada sobre todos los ejemplares de entrenamiento se puede escribir:

$$H_\Theta(X) = X\Theta \tag{22}$$

Ecuación normal

Consideremos el costo y su derivada en el caso multivariado:

$$J(\theta) = \frac{1}{2m}\sum_{i=1}^{m}(h_\theta(x^{(i)}) - y^{(i)})^2 \tag{23}$$

$$\frac{\partial J}{\partial \theta} = \frac{1}{m}\sum_{i=1}^{m}(h_\theta(x^{(i)}) - y^{(i)})x^{(i)} \tag{24}$$

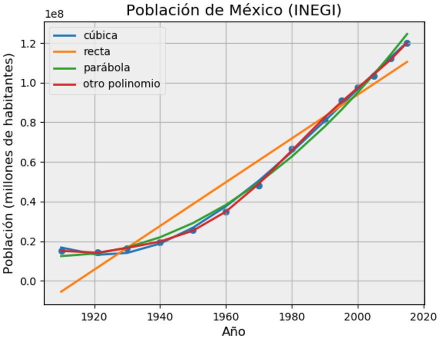
y tomémonos un tiempo para escribir la derivada con vectores y matrices, y buscar Θ tal que J es mínimo:

$$\begin{aligned}
\nabla_\Theta J = \frac{1}{m}((\mathbf{X}\Theta - Y)^T\mathbf{X})^T = \mathbf{X}^T(\mathbf{X}\Theta - Y) &= 0 \\
\mathbf{X}^T\mathbf{X}\Theta - \mathbf{X}^T Y &= 0 \\
(\mathbf{X}^T\mathbf{X})\Theta &= \mathbf{X}^T Y \\
\Theta &= (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T Y
\end{aligned} \tag{25}$$

Temas

1. Regresión
   - Regresión lineal univariada
   - Regresión lineal multivariada
   - Regresión polinomial
2. Descenso por el grandiente
3. Regularización

Regresión polinomial



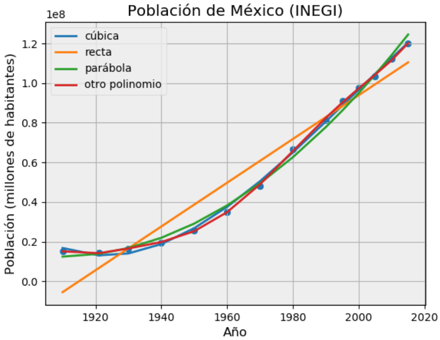
Podemos definir nuevas características:

$$h_\theta(x) = \theta_0 + \theta_1 x_1 + \theta_2 x_2 + \theta_3 x_3$$

$$h_\theta(x) = \theta_0 + \theta_1 size + \theta_2 size^2 + \theta_3 size^3$$

$$x_1 = size \qquad x_2 = size^2 \qquad x_3 = size^3$$

Otras características



Podemos definir nuevas características:

$$h_\theta(x) = \theta_0 + \theta_1 x_1 + \theta_2 x_2$$

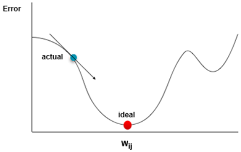
$$h_\theta(x) = \theta_0 + \theta_1 size + \theta_2 \sqrt{size}$$

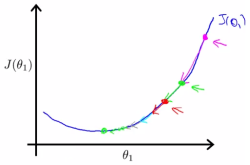
$$x_1 = size \qquad x_2 = \sqrt{size}$$

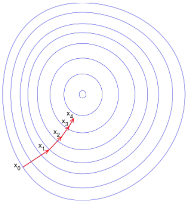
Temas

Descenso por el grandiente

- *Descenso por el gradiente* es un algoritmo de optimization de primer orden[1].
- Para encontrar un **mínimo local** de una función f, a partir de un punto P_0, se avanza un paso proporcional a $-\nabla_P f(P_0)$ (o su aproximado).
- Si se avanza en la dirección $\nabla_P f(P_0)$, se acerca a un **máximo local** y el procedimiento se conoce como *ascenso por el gradiente*.







[1]Depende de la primera derivada, que da la tangente a la curva.

Ejemplo

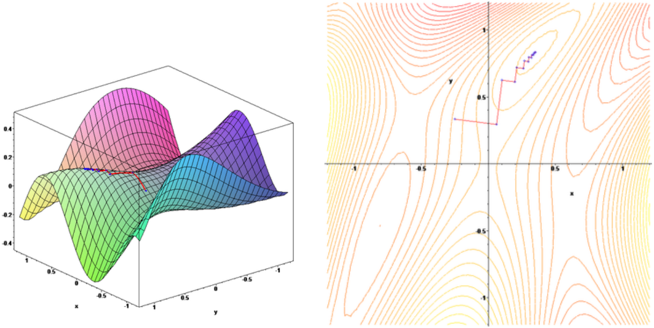


Figura: Centro: $f(x, y) = \sin(\frac{1}{2}x^2 - \frac{1}{4}y^2 + 3)\cos(2x + 1 - e^y)$. Der: Ascenso zigzagueando.

Algoritmo

Algoritmo 1 Descenso por el gradiente

1: **repeat**
2: $\quad P_{t+1} \leftarrow P_t - \alpha \nabla_P f(P_t)$
3: **until** $|f(P_{t+1}) - f(P_t)| < \varepsilon$

donde:

- $P \in \mathbb{R}^n$
- $\alpha \in \mathbb{R}$, determina la velocidad con que se avanza por el gradiente.

Alfa

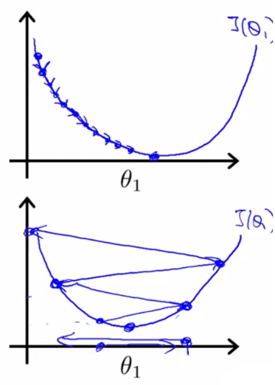


Figura: Si α es demasiado pequeño, tarda mucho en converger. Si es muy grande, diverge.

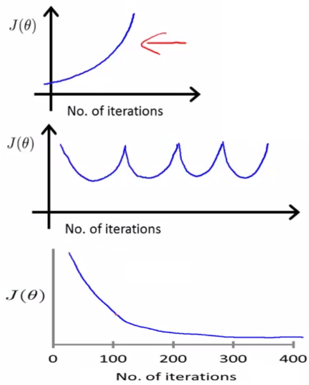


Figura: Graficar $J(\theta)$ *vs.* α permite identificar si el valor de α es adecuado.

Descenso por el gradiente *en línea* aplicado a regresión lineal

Para regresión univariada, sea:

$$
\begin{aligned}
f &= J(\theta_0, \theta_1) = \frac{1}{2}(h(x) - y)^2 \\
P_0 &= (\theta_0 = a_0, \theta_1 = b_0) \\
-\nabla_P f(P_0) &= \begin{bmatrix} \frac{\partial J}{\partial \theta_0} \\ \frac{\partial J}{\partial \theta_1} \end{bmatrix} = \begin{bmatrix} \theta_0 + \theta_1 x^{(i)} - y^{(i)} \\ (\theta_0 + \theta_1 x^{(i)} - y^{(i)}) x^{(i)} \end{bmatrix}
\end{aligned}
$$

Para regresión multivariada, sea:

$$
\begin{aligned}
f &= J(\theta_0, ..., \theta_n) = J(\Theta) = \frac{1}{2}(h(x^{(i)}) - y^{(i)})^2 \\
P_0 &= (\theta_0 = p_{0_0}, .., \theta_n = p_{n_0}) \\
-\nabla_P f(P_0) &= \begin{bmatrix} ... \\ \frac{\partial J}{\partial \theta_j} \\ ... \end{bmatrix} = \begin{bmatrix} ... \\ (\Theta^T X^{(i)} - y^{(i)}) x_j^{(i)} \\ ... \end{bmatrix}
\end{aligned}
$$

Descenso por el gradiente por lotes.

Algoritmo 2 Descenso por el gradiente de $J(\theta_0, \theta_1)$ en lotes

1: **repeat**
2: $\quad \Theta_{t+1} \leftarrow \Theta_t - \alpha \nabla_\Theta J(\theta_{0(t-1)}, \theta_{1(t-1)})$
3: **until** $|J(\theta_0, \theta_1)| < \varepsilon$

- α es un parámetro del *algoritmo de aprendizaje* o un *metaparámetro*.
- El algoritmo se dice que se ejecuta en *lotes*, porque ∇J utiliza todos los ejemplares de entrenamiento para su cálculo.

Descenso por el gradiente por lotes aplicado a regresión lineal

Para regresión univariada, sea:

$$\begin{aligned} f &= J(\theta_0,\theta_1) = \frac{1}{2m}\sum_{i=1}^{m}(h(x^{(i)}) - y^{(i)})^2 \\ P_0 &= (\theta_0 = a_0, \theta_1 = b_0) \\ -\nabla_P f(P_0) &= \begin{bmatrix} \frac{\partial J}{\partial \theta_0} \\ \frac{\partial J}{\partial \theta_1} \end{bmatrix} = \begin{bmatrix} \frac{1}{m}\sum_{i=1}^{m}(\theta_0 + \theta_1 x^{(i)} - y^{(i)}) \\ \frac{1}{m}\sum_{i=1}^{m}(\theta_0 + \theta_1 x^{(i)} - y^{(i)})x^{(i)} \end{bmatrix} \end{aligned}$$

Para regresión multivariada, sea:

$$\begin{aligned} f &= J(\theta_0, ..., \theta_n) = J(\Theta) = \frac{1}{2m}\sum_{i=1}^{m}(h(x^{(i)}) - y^{(i)})^2 \\ P_0 &= (\theta_0 = p_{0\,0}, .., \theta_n = p_{n\,0}) \\ -\nabla_P f(P_0) &= \begin{bmatrix} \dots \\ \frac{\partial J}{\partial \theta_j} \\ \dots \end{bmatrix} = \begin{bmatrix} \dots \\ \frac{1}{m}\sum_{i=1}^{m}(\Theta^T X^{(i)} - y^{(i)})x_j^{(i)} \\ \dots \end{bmatrix} \end{aligned}$$

Normalización o Reescalamiento

- Reescalar la magnitud de cada característica, de tal modo que $-1 \leqslant x_i \leqslant 1$ aproximadamente.

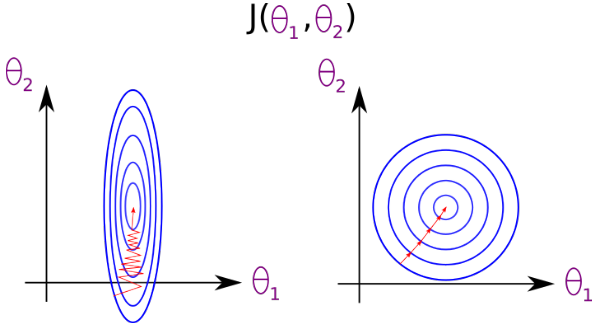


Figura: Comportamiento del descenso por el gradiente en a) datos no normalizador y b) datos normalizados.

Normalización o Reescalamiento

- Por ejemplo, normalizar:

$$x_i' = \frac{x_i - \mu_i}{s_1} \tag{26}$$

  donde $\mu_i = \frac{1}{m}\sum_{j=1}^{m} x_i^{(j)}$ es la media y $s_i = \max(x_i) - \min(x_i)$ es el rango.
- Se puede usar la desviación estándar $s_2 = \sqrt{\frac{1}{m-1}\sum_{j=1}^{m}(x_i^{(j)} - \mu)^2}$ en lugar de s_1.
- *No normalizar x_0.

Alternativas

- Gradiente conjugado.
- BFGS.
- L-BFGS.
- Adam.
- Adagrad.

Se pueden utilizar bibliotecas de cálculo numérico ya implementadas en `Octave`.

```
function[J,GradJ] = costFunction(Theta)
jVal = ...   % J(Θ)
GradJ = ... % ∇J

options = optimset('GradObj', 'on',      % Gradiente
                   'MaxIter', '100');   % 100 iters
thetaZero = [0 0];
[Theta, J, flag] = fminunc(@costFunction,
                   thetaZero, options);
```

Ecuación normal vs. Descenso por el gradiente

Ecuación normal	Descenso por el gradiente
$\Theta = (\mathbf{X}^T\mathbf{X})^1\mathbf{X}^T\mathbf{Y}$	$\Theta_{t+1} = \Theta_t - \alpha\nabla_\Theta J(\Theta_t)$
	Requiere elegir α
Se sugiere reescalar las características	Requiere reescalar las características
Requiere invertir una matriz $n \times n, O(n^3)$	Requiere varias iteraciones
Muy lento si hay muchas característica $n > 10^5$	Funciona igualmente bien si n es grande.

Temas

Regularización

$$J(Theta) = ... + \frac{\lambda}{2m}\sum_{i=1}^{n}\theta_i^2 \tag{27}$$

$$\nabla J = ... + \frac{\lambda}{m}\theta_i \text{ excepto } \theta_0 \tag{28}$$

$$\Theta = (\mathbf{X}^T\mathbf{X} + \lambda\mathbf{K})^{-1}\mathbf{X}^T Y \tag{29}$$

$$K = \begin{bmatrix} 0 & 0 & 0 & ... & 0 \\ 0 & 1 & 0 & ... & 0 \\ 0 & 0 & 1 & ... & 0 \\ ... & & & & \\ 0 & 0 & 0 & ... & 1 \end{bmatrix} \tag{30}$$

$$\lambda > 0 \tag{31}$$

Referencias I

Andrew Ng (2015), *Machine Learning*, Coursera.