Regresión lineal

1. Regresión lineal simple
2. Regresión lineal múltiple

Regresión lineal

Obtener una representación lineal de distribución de datos --- Se usa estimación de valor dado un cato de entrada.

(1) Regresión lineal simple

Datos distribuidos son 2D

Una variable independiente → Una variable dependiente

Ejemplos

Obtener relación lineal entre calif. de matemática y calif. de inglés.

Obtener relación lineal entre antigüedad de trabajo y sueldo que recibe

(2) Regresión lineal múltiple

Datos distribuidos son mas de 3D (multidimensional)

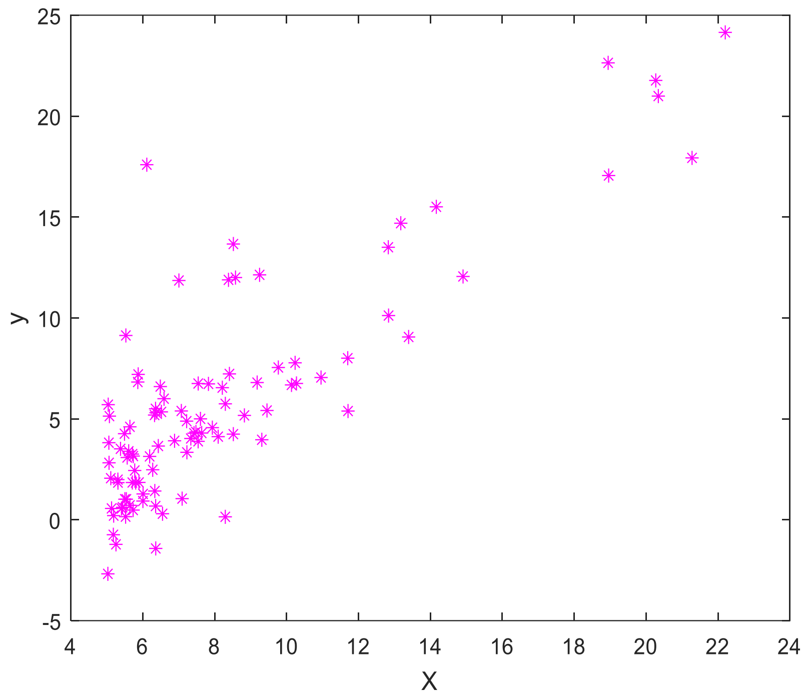
Múltiples variables independientes → una variable dependiente

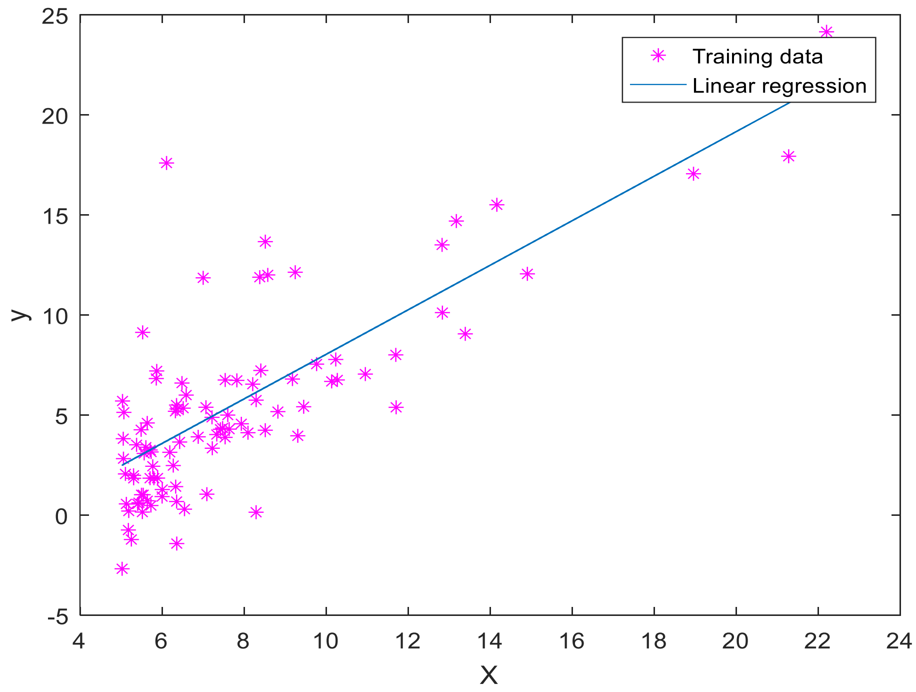
Ejemplos

(1) Predecir el costo de inmueble a partir de condiciones de inmuebles (# de habitaciones, años de construcción, colonia, etc.).
(2) Predecir venta de helado a partir de clima y día de semana (fin de semana o día de trabajo)

Regresión Lineal Simple

Objetivo: Obtener una línea que representa distribución de datos 2D

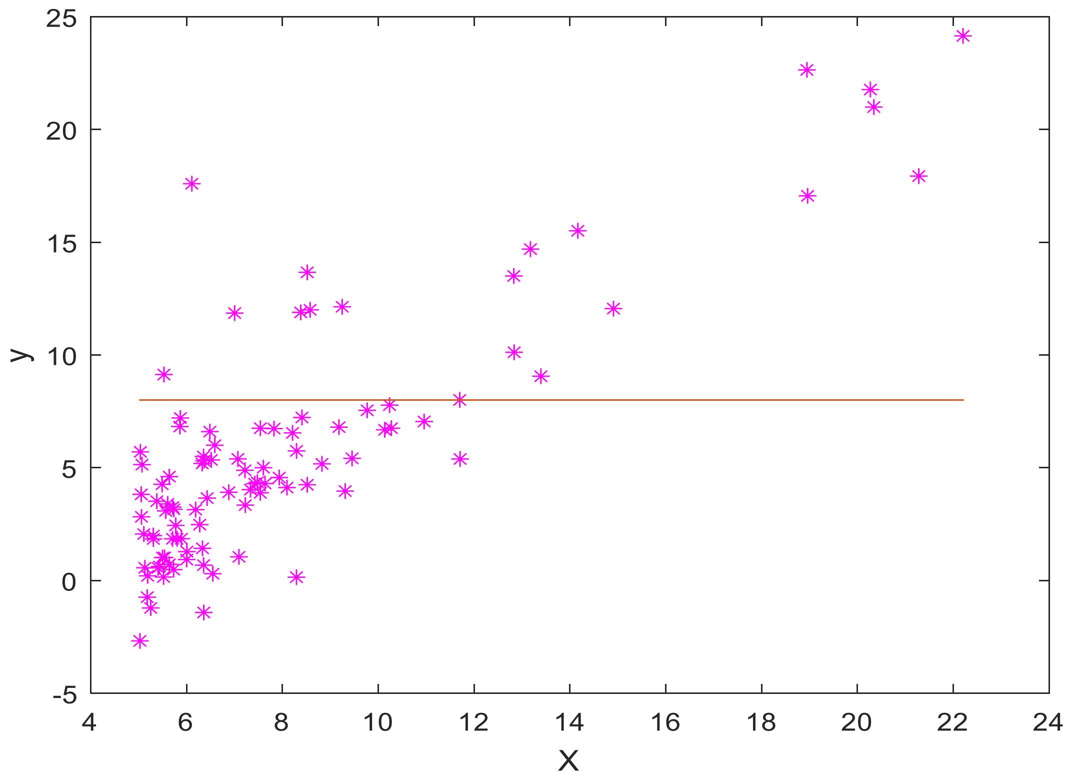




Ecuación de Línea

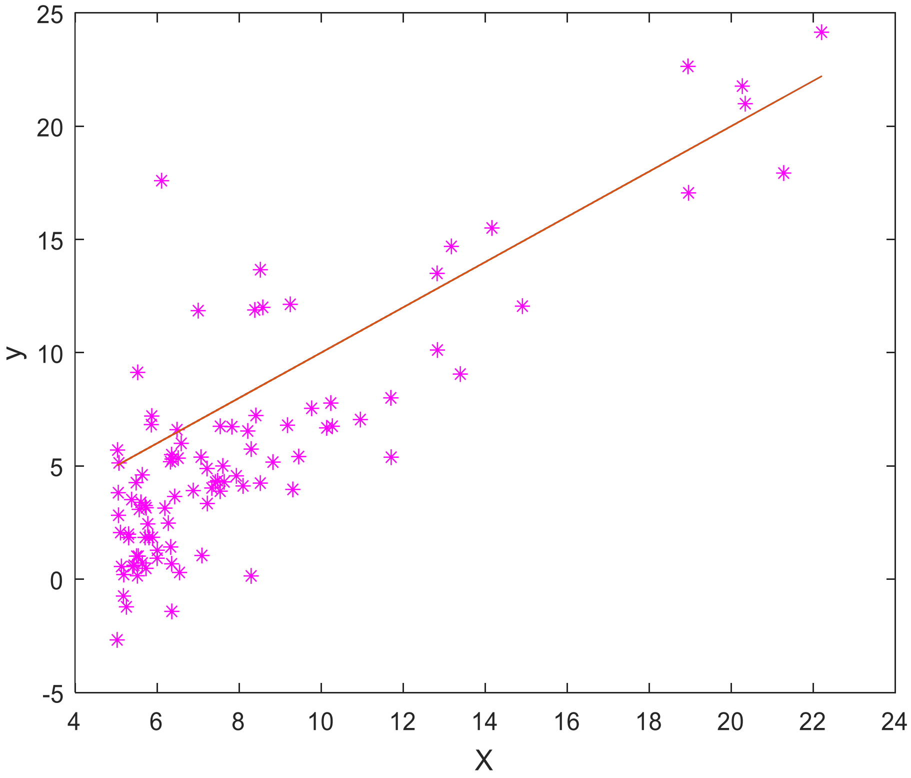
$$y = \theta_1 x + \theta_0$$

$$(\theta_0, \theta_1) = (8,0)$$



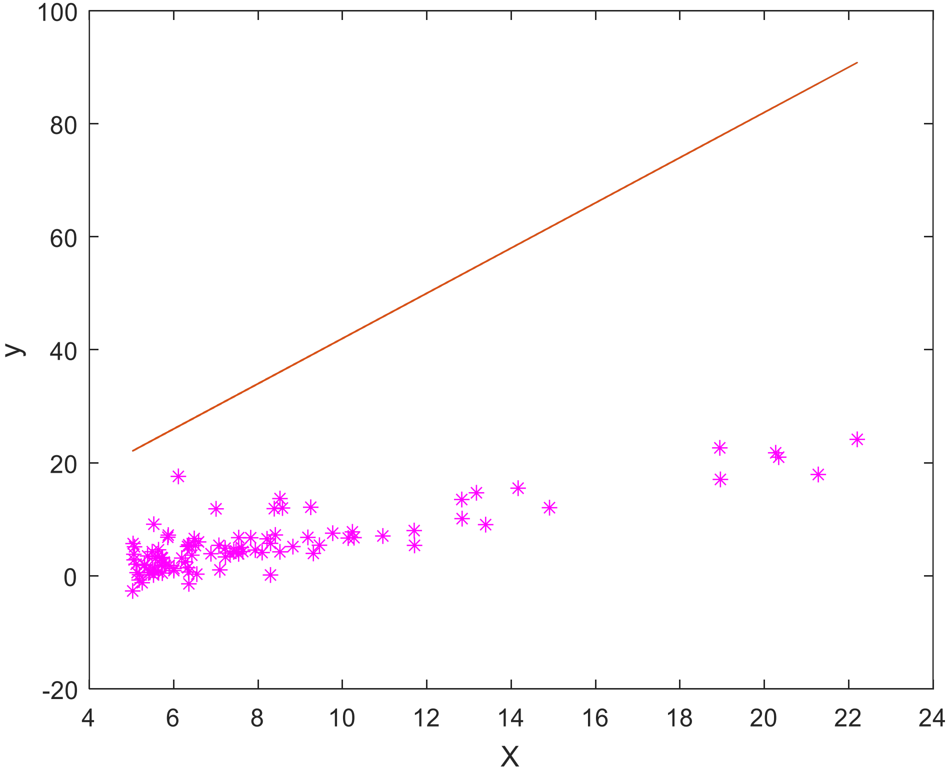
$$y = \theta_1 x + \theta_0$$

$$(\theta_0, \theta_1) = (0,1)$$



$$y = \theta_1 x + \theta_0$$

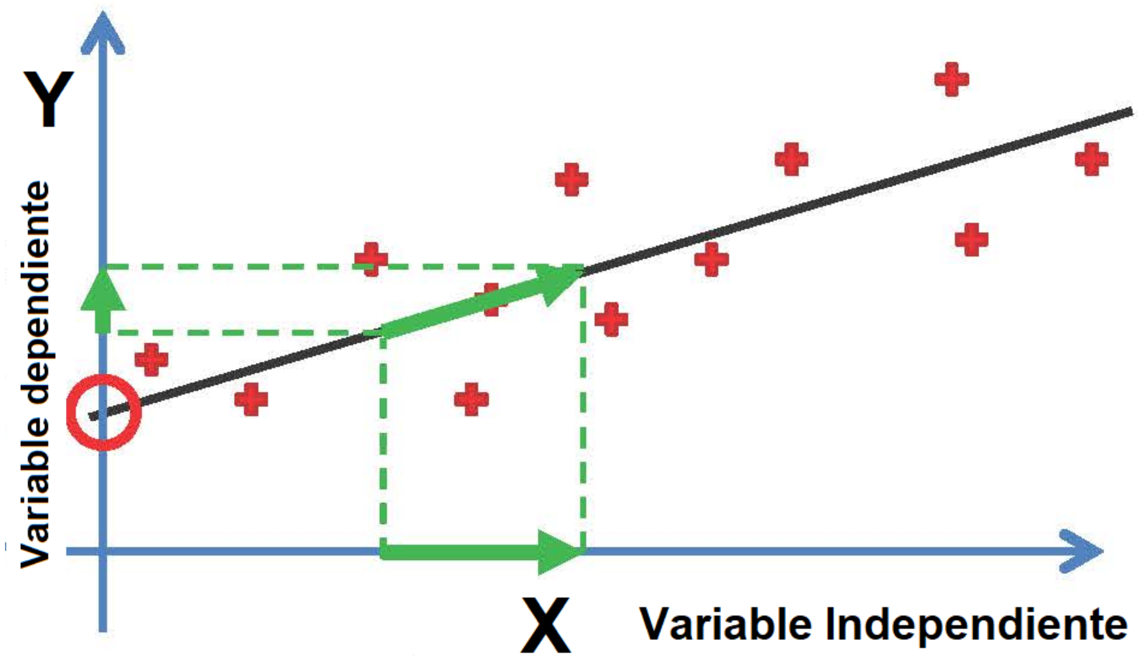
$$(\theta_0, \theta_1) = (2,4)$$



Función de coste

Obtenemos una función (ecuación de línea) óptima $y = \theta_1 x + \theta_0$

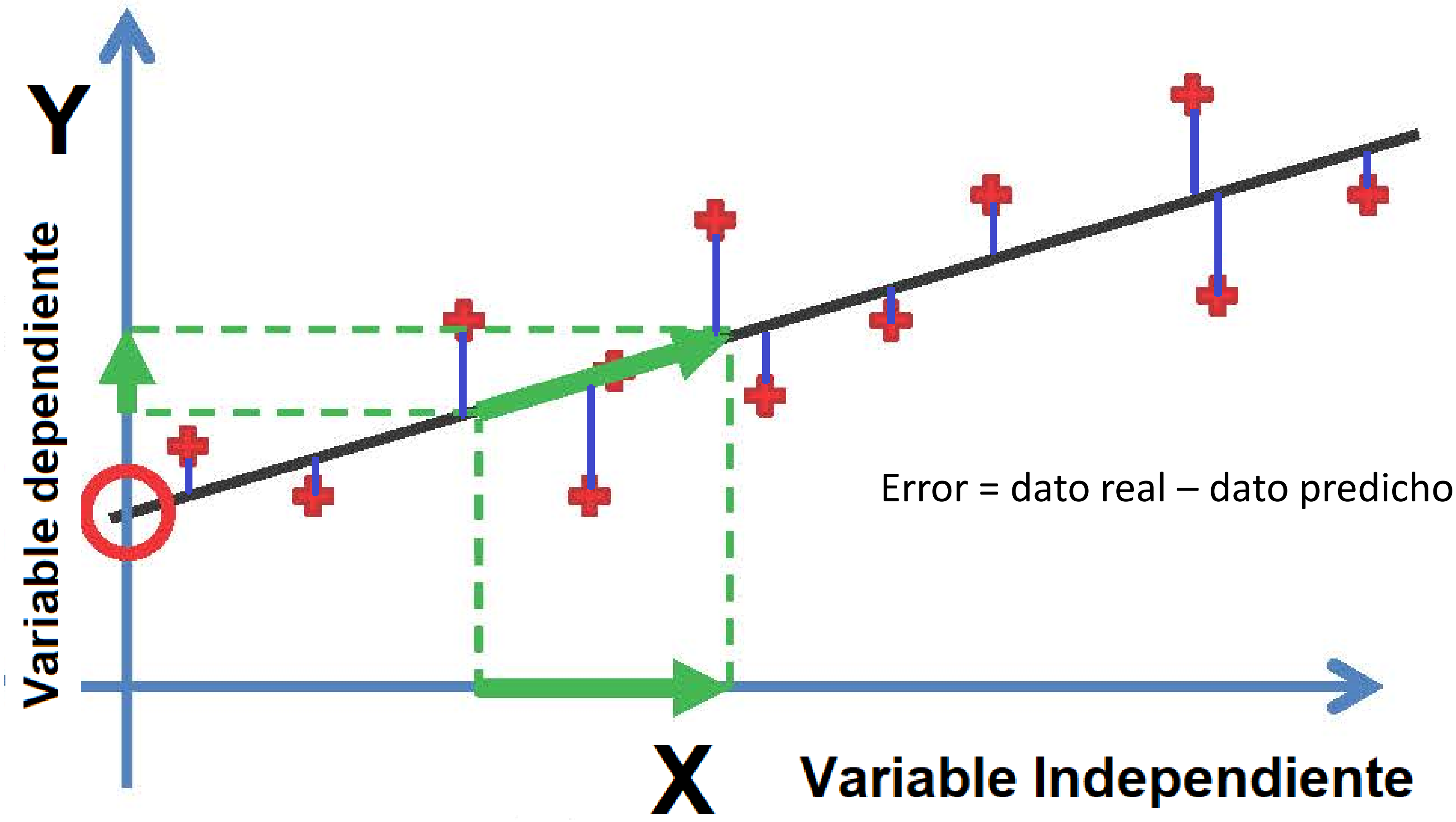
Obtenemos dos parámetros θ_0 y θ_1



θ_0 : intercepto

θ_1: Inclinación

Función de Coste
= El valor cuadrático medio de error



Función de Coste
El valor cuadrático medio de error

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

Donde $h_\theta\left(x^{(i)}\right)$ es variable predicho para i-esimo variable independiente, o sea $y = h_\theta\left(x^{(i)}\right) = \theta_1 x^{(i)} + \theta_0$ y $y^{(i)}$ es variable real.

m es número de datos existentes.

Objetivo

Obtener $\boldsymbol{\theta_0}$ y $\boldsymbol{\theta_1}$ que minimice la función de coste $\boldsymbol{J(\theta_0, \theta_1)}$

Regresión lineal simple

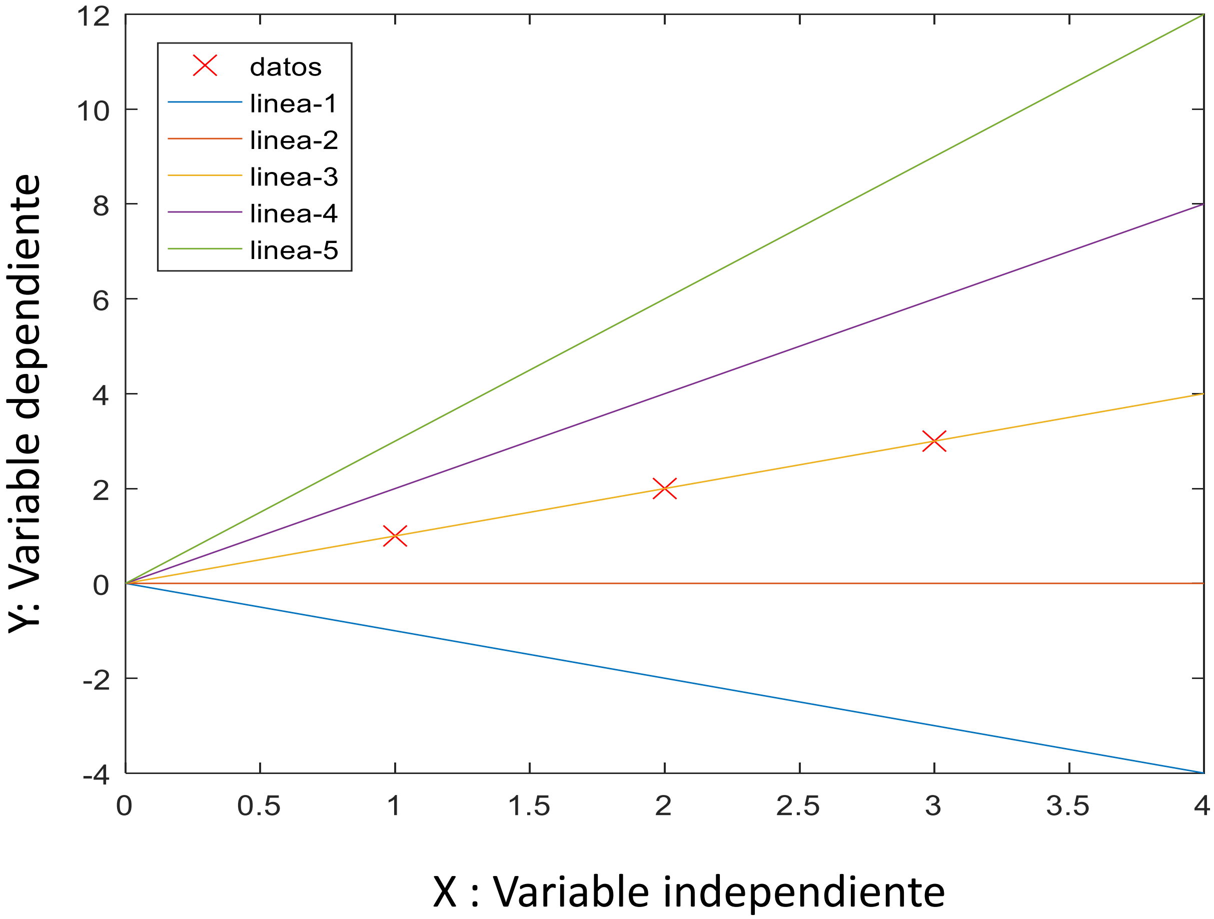
Modelo: $\boldsymbol{y = h_\theta(x) = \theta_1 x + \theta_0}$

Parámetros del modelo : $\boldsymbol{\theta_0,\ \theta_1}$

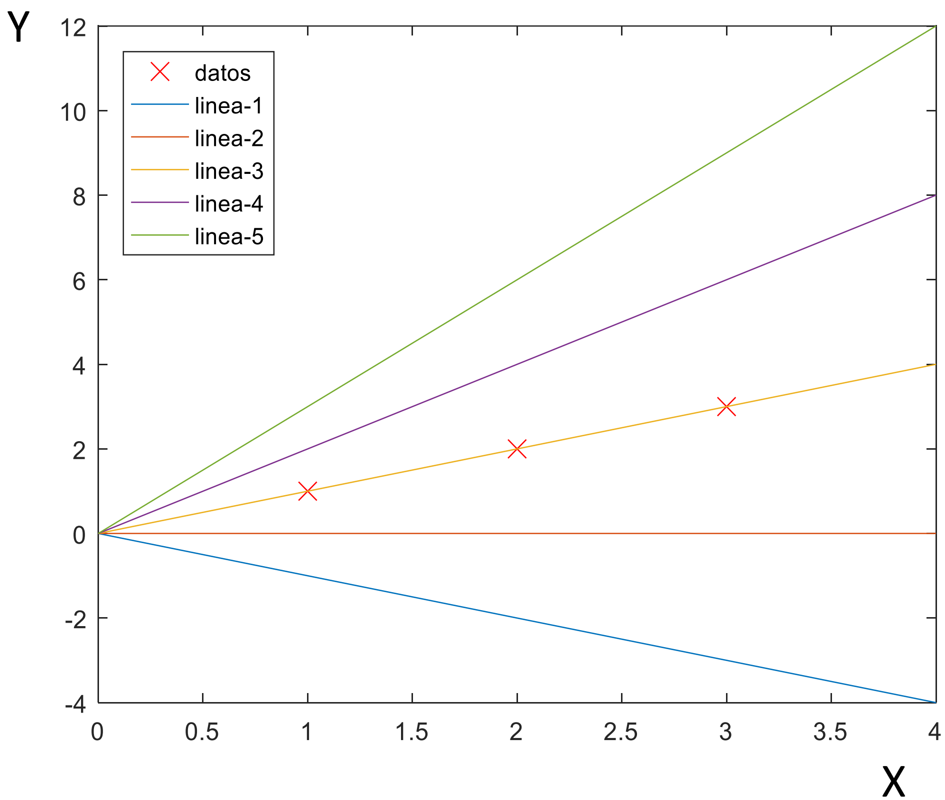
Función de coste: $\boldsymbol{J(\theta_0, \theta_1) = \frac{1}{2m}\sum_{i=1}^{m}\left(h_\theta\left(x^{(i)}\right) - y^{(i)}\right)^2}$

Meta: $\boldsymbol{\min_{\theta_0,\theta_1} J(\theta_0, \theta_1)}$

Ejemplo sencillo para obtener función de coste



Ejercicio 1: Obtener función de coste de cada predicción lineal



Datos reales: (1,1), (2,2), (3,3)

Linea-1: $h_\theta(x) = -x$

Linea-2: $h_\theta(x) = 0$

Linea-3: $h_\theta(x) = x$

Linea-4: $h_\theta(x) = 2x$

Linea-5: $h_\theta(x) = 3x$

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

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

Respuesta

Linea-1: $\boldsymbol{h_\theta(x) = -x}$

$$J(\theta_1) = \frac{1}{6}\sum_{i=1}^{3}\left(h_\theta\left(x^{(i)}\right) - y^{(i)}\right)^2 = \frac{1}{6}\left((-1-1)^2 + (-2-2)^2 + (-3-3)^2\right)$$

$$= \frac{1}{6}(4 + 16 + 36) \approx \mathbf{9.33}$$

Linea-2: $\boldsymbol{h_\theta(x) = 0}$

$$J(\theta_1) = \frac{1}{6}\sum_{i=1}^{3}\left(h_\theta\left(x^{(i)}\right) - y^{(i)}\right)^2 = \frac{1}{6}\left((0-1)^2 + (0-2)^2 + (0-3)^2\right) = \frac{1}{6}(1 + 4 + 9)$$

$$\approx \mathbf{2.33}$$

Linea-3: $\boldsymbol{h_\theta(x) = x}$

$$J(\theta_1) = \frac{1}{6}\sum_{i=1}^{3}\left(h_\theta\left(x^{(i)}\right) - y^{(i)}\right)^2 = \frac{1}{6}\left((1-1)^2 + (2-2)^2 + (3-3)^2\right) = \mathbf{0}$$

Respuesta

Linea-4: $\boldsymbol{h_\theta(x) = 2x}$

$$J(\theta_1) = \frac{1}{6}\sum_{i=1}^{3}\left(h_\theta\left(x^{(i)}\right) - y^{(i)}\right)^2 = \frac{1}{6}\left((2-1)^2 + (4-2)^2 + (6-3)^2\right)$$

$$= \frac{1}{6}(1 + 4 + 9) \approx \mathbf{2.33}$$

Linea-5: $\boldsymbol{h_\theta(x) = 3x}$

$$J(\theta_1) = \frac{1}{6}\sum_{i=1}^{3}\left(h_\theta\left(x^{(i)}\right) - y^{(i)}\right)^2 = \frac{1}{6}\left((3-1)^2 + (6-2)^2 + (9-3)^2\right)$$

$$= \frac{1}{6}(4 + 16 + 36) \approx \mathbf{9.33}$$

Algoritmo de Gradiente descendente

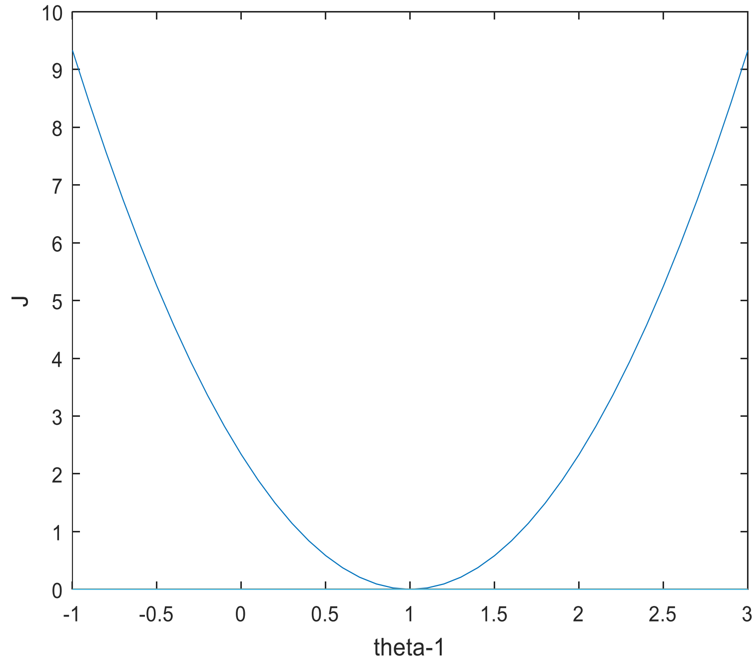
- Un algoritmo iterativo para encontrar los valores de parámetros óptimos (minimiza la función de coste)
- Basado en derivada parcial de función de coste

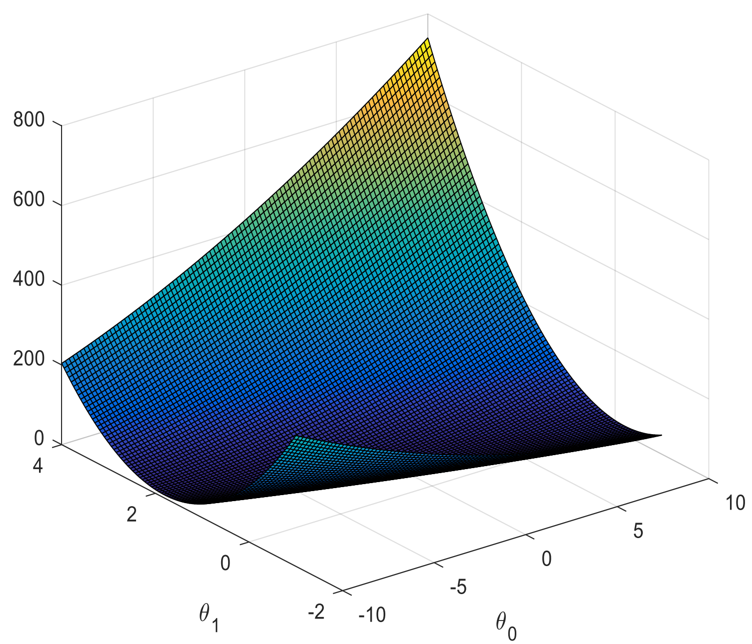
$$J(\theta_0, \theta_1) = \frac{1}{2m} \sum_{i=1}^{m} \left(h_\theta\left(x^{(i)}\right) - y^{(i)}\right)^2 = \frac{1}{2m} \sum_{i=1}^{m} \left(\theta_0 + \theta_1 x^{(i)} - y^{(i)}\right)^2$$

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

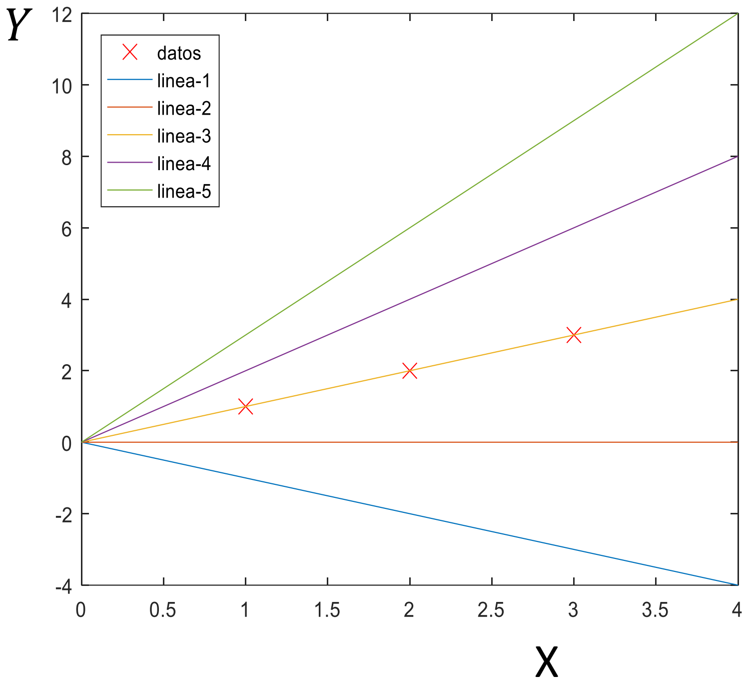
Superficie de función de coste

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





Relación entre θ_1 y el valor de función de coste J



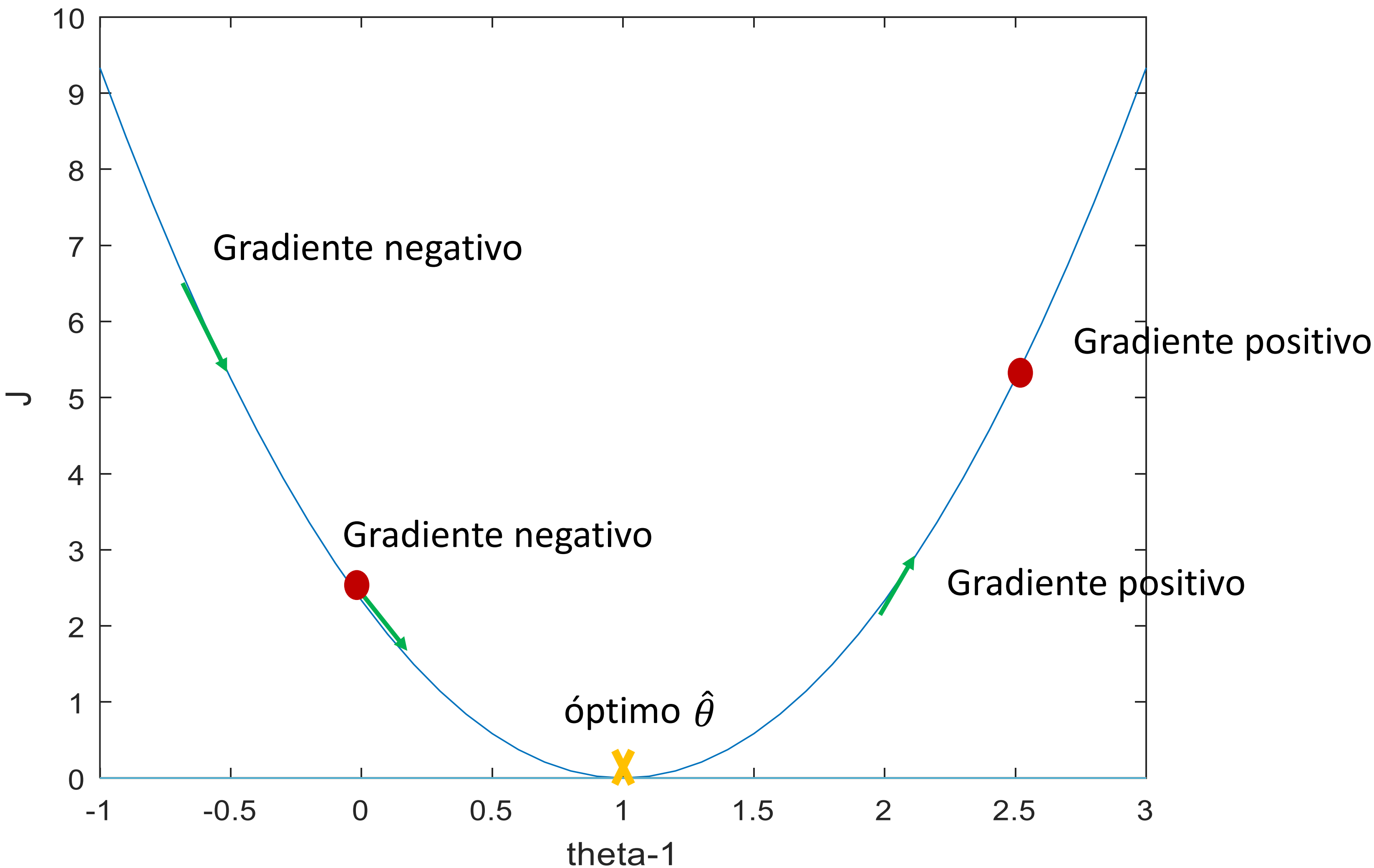
$\theta_0 = 0$

Línea	Valor de θ_1	J
$h_\theta(x) = -x$	-1	$\mathbf{9.33}$
$h_\theta(x) = 0$	0	$\mathbf{2.33}$
$h_\theta(x) = x$	1	$\mathbf{0}$
$h_\theta(x) = 2x$	2	$\mathbf{2.33}$
$h_\theta(x) = 3x$	3	$\mathbf{9.33}$

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

Que es gradiente?

Pendiente de función en un punto



¿Qué es gradiente?

- Pendiente de función en un punto

- Derivada parcial de la función respecto a parámetro

Algoritmo de gradiente descendente

$$\theta_j = \theta_j - \alpha \frac{\partial}{\partial \theta_j} J(\theta_0, \theta_1), j = 0{,}1$$

Donde

$\boldsymbol{\frac{\partial}{\partial \theta_j} J(\theta_0, \theta_1)}$ es la derivada parcial de la función de coste con respecto a θ_j, j=0 o 1.

$\boldsymbol{\alpha}$ es factor de aprendizaje que controla velocidad de aprendizaje y desajuste de error.

Ejercicio 2: Obtener derivada parcial de la función de coste

$$\frac{\partial}{\partial\theta_j} J(\theta_0, \theta_1) = \frac{\partial}{\partial\theta_j}\left[\frac{1}{2m}\sum_{i=1}^{m}\left(\theta_0 + \theta_1 x^{(i)} - y^{(i)}\right)^2\right]$$

Obtener $\frac{\partial}{\partial\theta_0} J(\theta_0, \theta_1)$ y $\frac{\partial}{\partial\theta_1} J(\theta_0, \theta_1)$

Respuesta

$$\frac{\partial}{\partial\theta_0}J(\theta_0,\theta_1) = \frac{\partial}{\partial\theta_0}\left[\frac{1}{2m}\sum_{i=1}^{m}\left(\theta_0+\theta_1 x^{(i)}-y^{(i)}\right)^2\right] = \frac{1}{m}\sum_{i=1}^{m}\left(\theta_0+\theta_1 x^{(i)}-y^{(i)}\right)\times 1$$

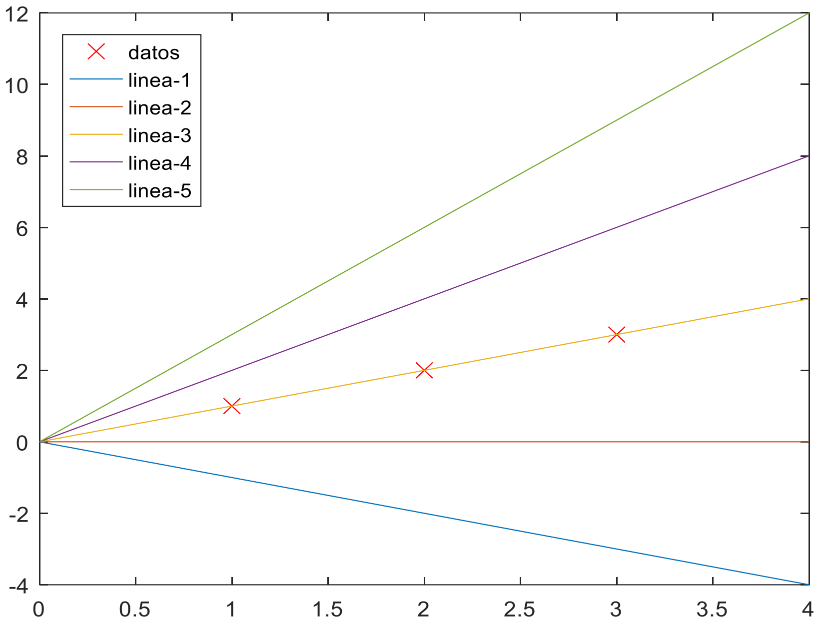
$$= \frac{1}{m}\sum_{i=1}^{m}\left(h_\theta\left(x^{(i)}\right)-y^{(i)}\right)$$

$$\frac{\partial}{\partial\theta_1}J(\theta_0,\theta_1) = \frac{\partial}{\partial\theta_1}\left[\frac{1}{2m}\sum_{i=1}^{m}\left(\theta_0+\theta_1 x^{(i)}-y^{(i)}\right)^2\right] =$$

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

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

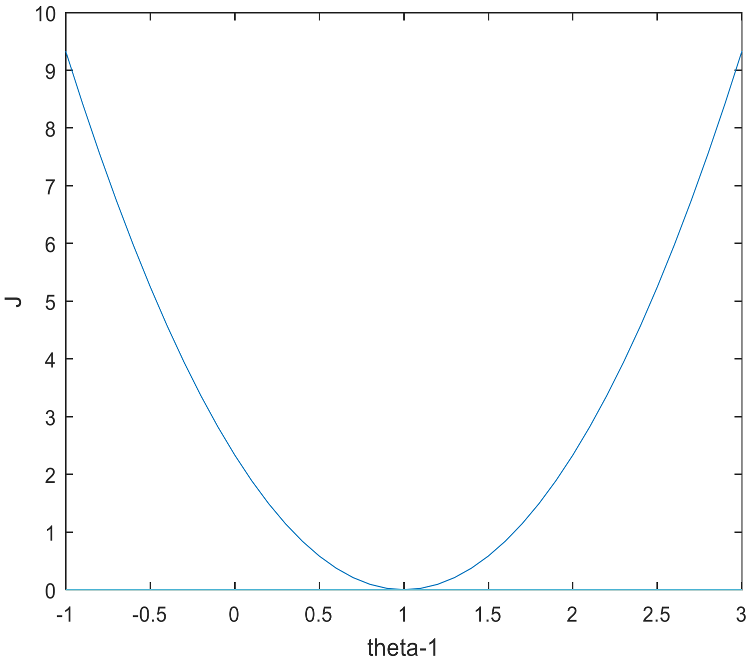
Ejercicio 3: obtener gradiente de función de coste



Los datos dado:
(1,1), (2,2), (3,3)

Óptima línea es línea-3, $h_\theta(x) = x$

$\theta_0 = 0, \theta_1 = 1$ son parámetros óptimos



Linea-1: $h_\theta(x) = -x : \theta_0 = 0, \theta_1 = -1$

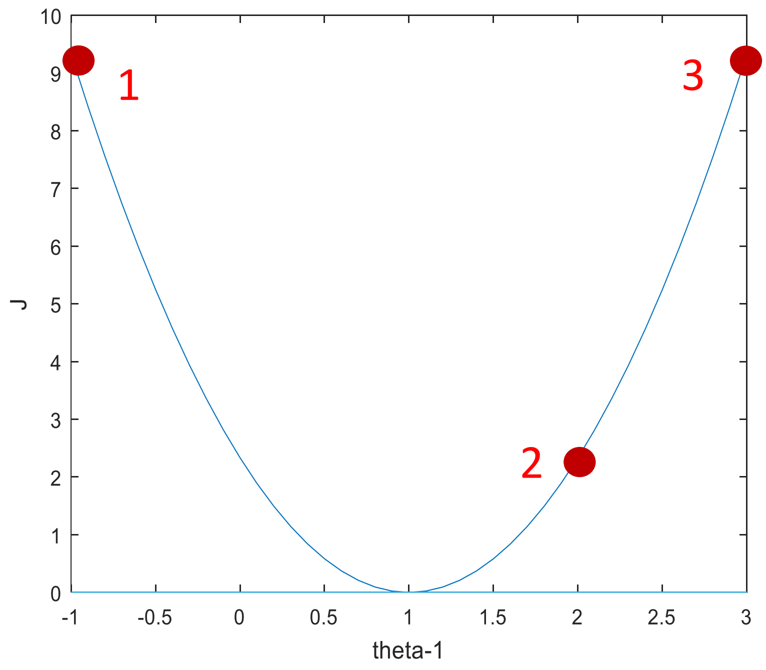
Linea-2: $h_\theta(x) = 0 : \theta_0 = 0, \theta_1 = 0$

Linea-4: $h_\theta(x) = 2x: \theta_0 = 0, \theta_1 = 2$

Linea-5: $h_\theta(x) = 3x: \theta_0 = 0, \theta_1 = 3$

1. Obtener gradiente en el $\theta_1 = -1, \theta_0 = 0$
2. Obtener gradiente en el $\theta_1 = 2, \theta_0 = 0$
3. Obtener gradiente en el $\theta_1 = 3, \theta_0 = 0$

$$\theta_j = \theta_j - \alpha \frac{\partial}{\partial \theta_j} J(\theta_0, \theta_1), j = 0{,}1$$



$$\frac{\partial}{\partial \theta_1} J(\theta_0, \theta_1)$$

$$= \frac{\partial}{\partial \theta_1}\left[\frac{1}{2m}\sum_{i=1}^{m}\left(\theta_0 + \theta_1 x^{(i)} - y^{(i)}\right)^2\right] =$$

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

$x^{(1)}, y^{(1)} = (1{,}1)$
$x^{(2)}, y^{(2)} = (2{,}2)$
$x^{(3)}, y^{(3)} = (3{,}3)$

Respuesta

Primer punto (θ_0, θ_1)=(0,-1)

$$\frac{\partial}{\partial\theta_1}J(\theta_0,\theta_1) = \frac{1}{m}\sum_{i=1}^{m}\left(\theta_0 + \theta_1 x^{(i)} - y^{(i)}\right) \times x^{(i)}$$

$$= \frac{1}{3}\Big(((-1) \times 1 - 1) \times 1 + \big((-1) \times 2 - 2\big) \times 2 + \big((-1) \times 3 - 3\big) \times 3\Big)$$

$$\approx -9.3$$

Segundo punto (θ_0, θ_1)=(0,2)

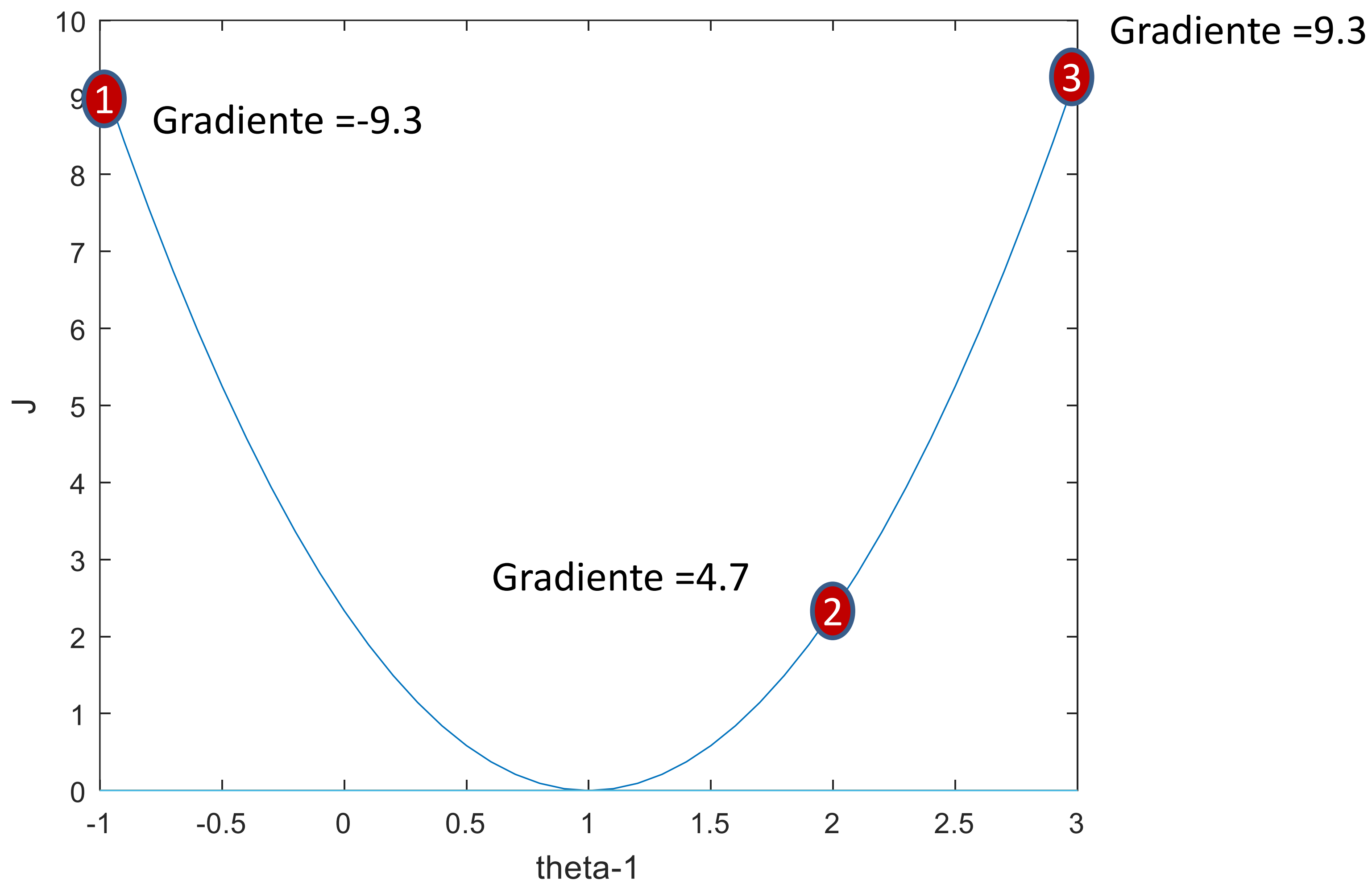
$$\frac{\partial}{\partial\theta_1}J(\theta_0,\theta_1) = \frac{1}{m}\textstyle\sum_{i=1}^{m}\left(\theta_0 + \theta_1 x^{(i)} - y^{(i)}\right) \times x^{(i)} =$$

$$\frac{1}{3}\Big(((2) \times 1 - 1) \times 1 + \big((2) \times 2 - 2\big) \times 2 + \big((2) \times 3 - 3\big) \times 3\Big) \approx 4.7$$

Tercer punto (θ_0, θ_1)=(0,3)

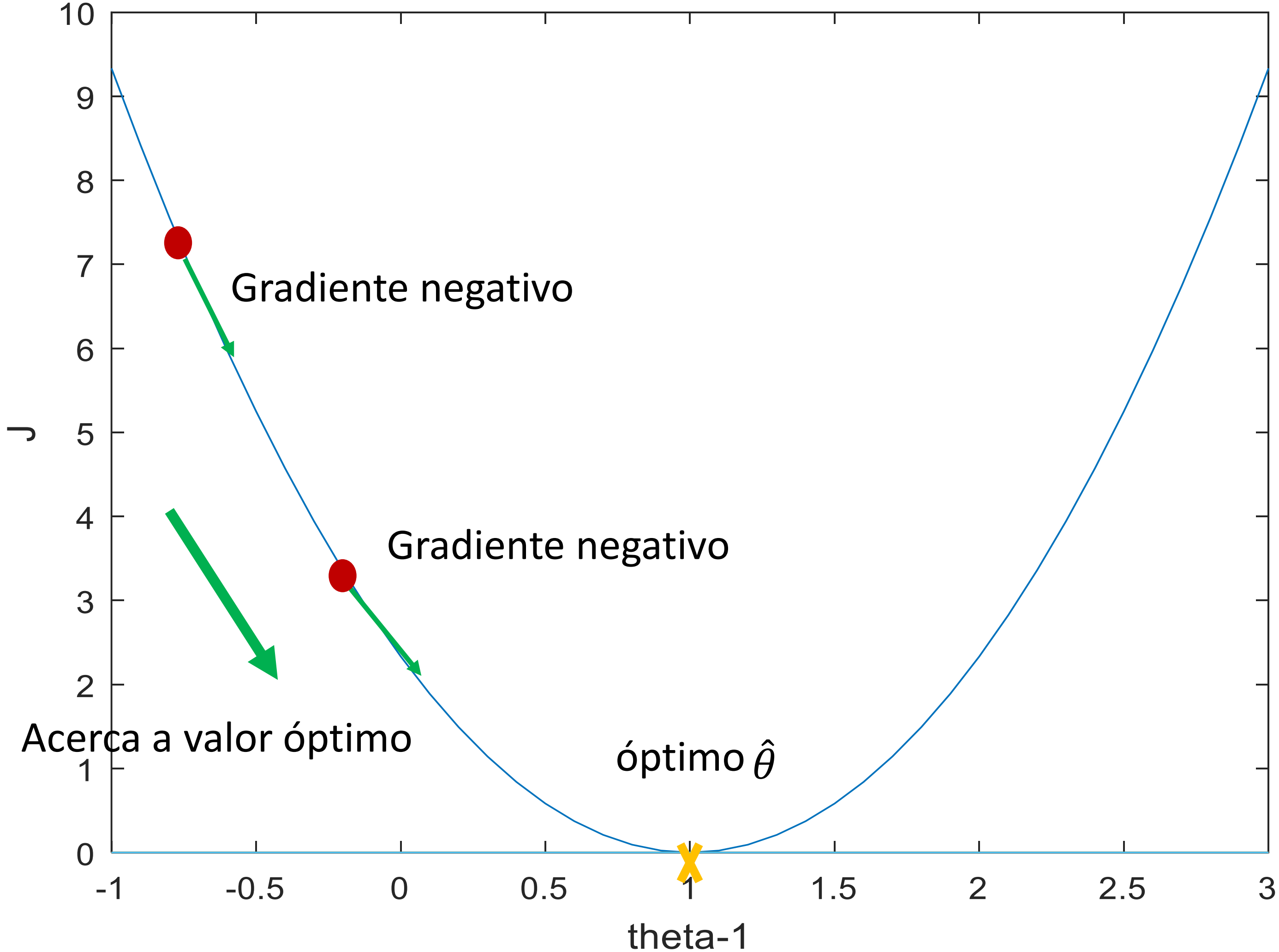
$$\frac{\partial}{\partial\theta_1}J(\theta_0,\theta_1) = \frac{1}{m}\sum_{i=1}^{m}\left(\theta_0 + \theta_1 x^{(i)} - y^{(i)}\right) \times x^{(i)} \approx 9.3$$

Respuesta


Gradiente =9.3
Gradiente =-9.3
Gradiente =4.7
1
2
3
10
9
8
7
6
5
4
3
2
1
0
J
-1
-0.5
0
0.5
1
1.5
2
2.5
3
theta-1

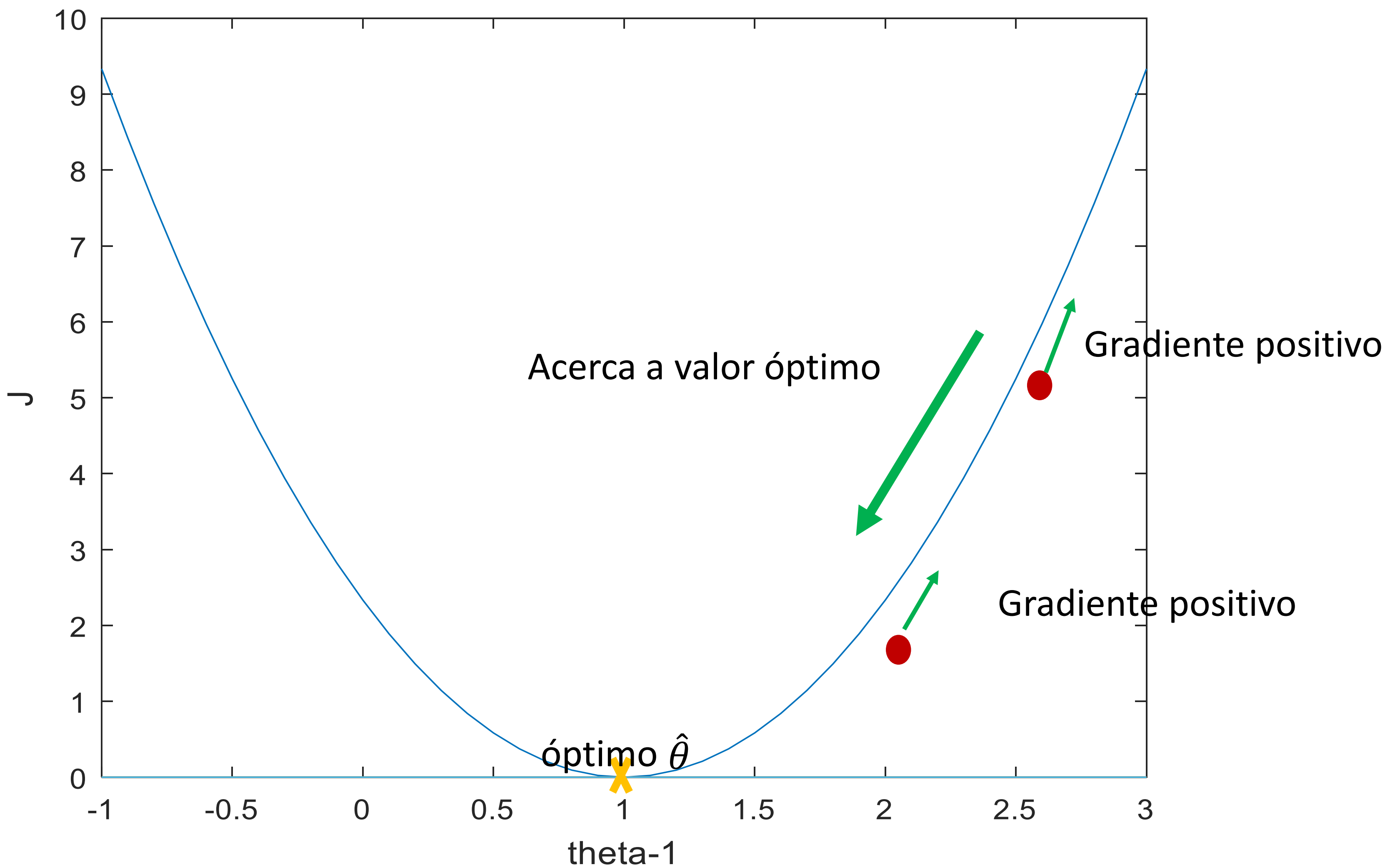
$$\theta_j = \theta_j - \alpha \frac{\partial}{\partial \theta_j} J(\theta_0, \theta_1), j = 0,1$$

Si $\theta_i < \widehat{\theta}_i$, entonces $\theta_i = \theta_i - \alpha \times (un\ valor\ negativo)$ y θ_i incrementa



$$\theta_j = \theta_j - \alpha \frac{\partial}{\partial \theta_j} J(\theta_0, \theta_1), j = 0{,}1$$

Si $\theta_1 > \widehat{\theta_1}$, entonces $\theta_1 = \theta_1 - \alpha \times (un\ valor\ positivo)$ y θ_1 decrece



Análisis de algoritmo “gradiente descendente”

$$\theta_j = \theta_j - \alpha \frac{\partial}{\partial \theta_j} J(\theta_0, \theta_1), j = 0,1$$

Si $\theta_j < \widehat{\theta}_j$, entonces $\theta_j = \theta_j - \alpha \times (un\ valor\ negativo)$ y θ_j incrementa

Si $\theta_j > \widehat{\theta}_j$, entonces $\theta_j = \theta_j - \alpha \times (un\ valor\ positivo)$ y θ_j decrece

En ambos casos, θ_j acerca al valor óptimo $\widehat{\theta}_j$.

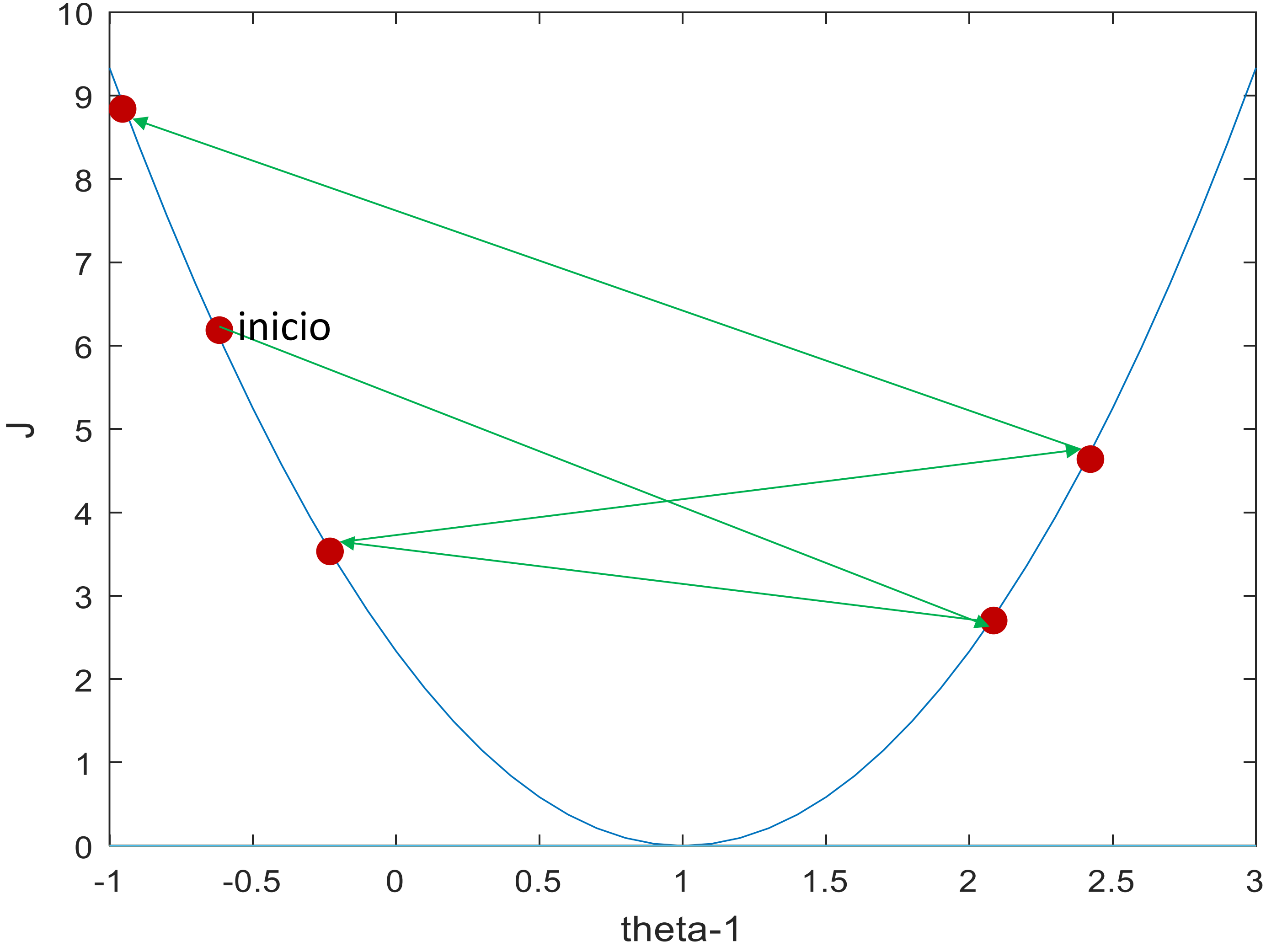
Donde J es función de coste:

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

Efecto de factor de aprendizaje α

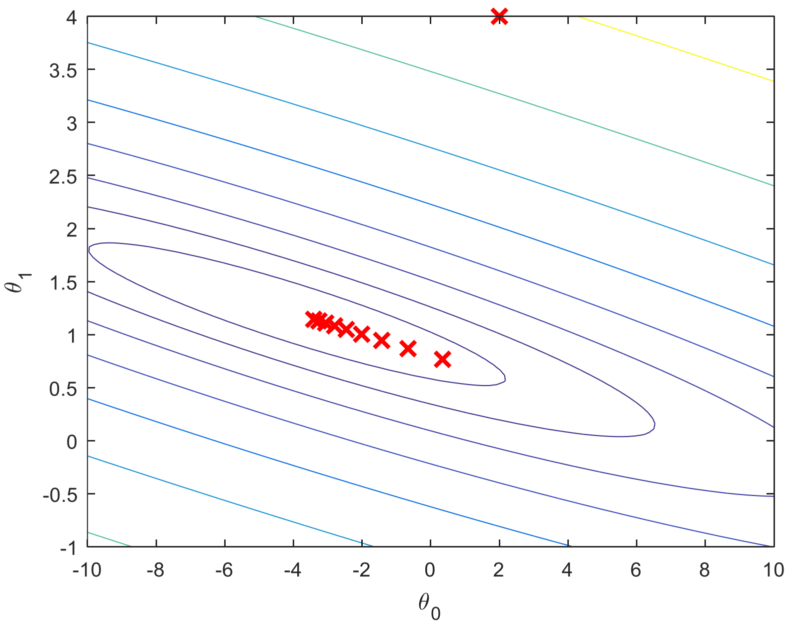
El factor de aprendizaje α define el paso del cambio que realiza a los parámetros del modelo θ_0 y θ_1, si el valor de α grande, el paso de aprendizaje es también grande, sin embargo, si este paso es demasiado grande, el sistema no converge, diverge.

Factor de aprendizaje es demasiado grande

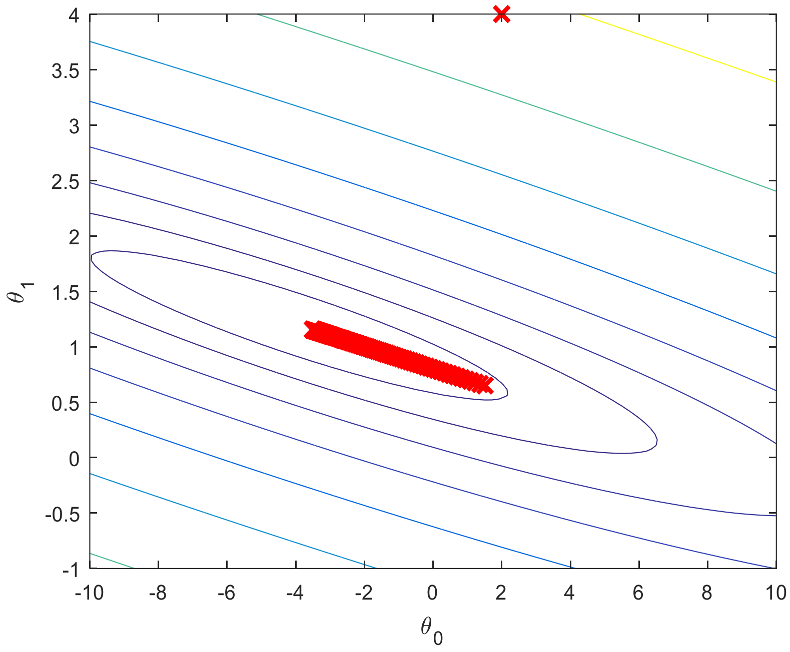


Efecto de factor de aprendizaje

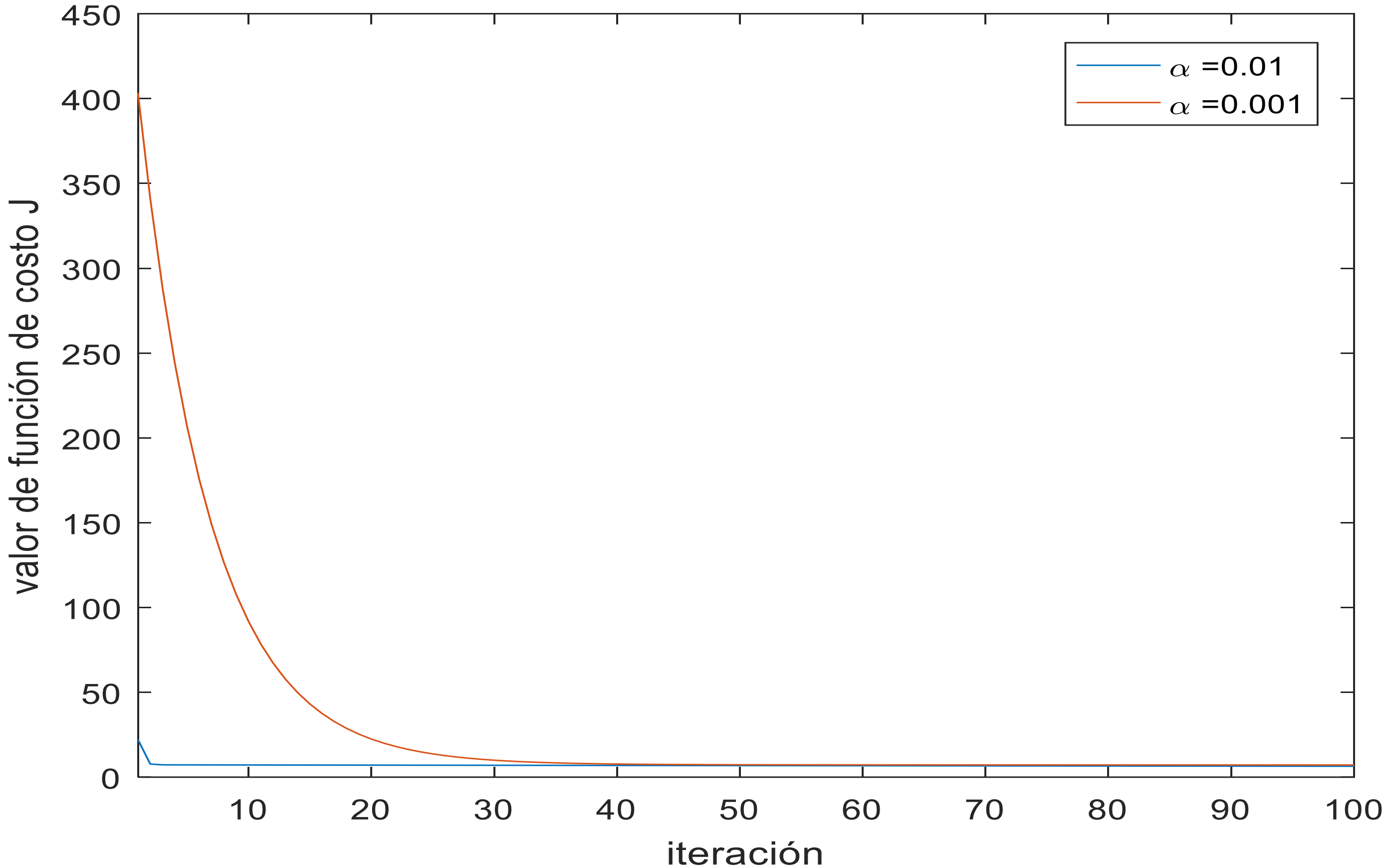
α=0.01



α=0.001



Efecto de Factor de aprendizaje



Algoritmos de gradiente descendente sus variantes

(1) Gradiente descendente

Usar todos los datos de entrenamiento para ajustar parámetros del modelo. **m = numero total patrones de entrenamiento**

(2) **Gradiente descendente estocástico**

Usar un dato de entrenamiento para ajustar parámetros.

m = 1 muestra

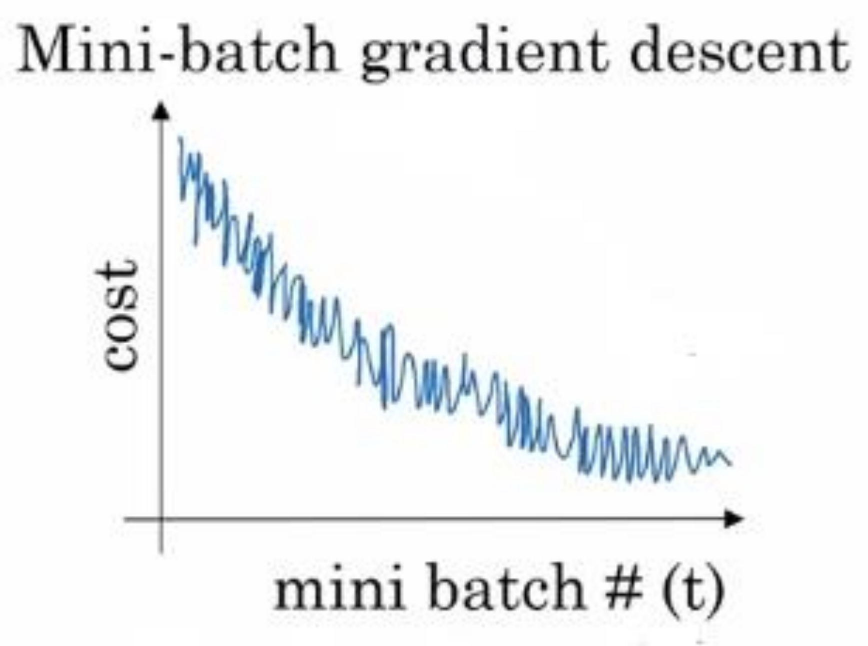
(3) **Gradiente descendente mini-batch**

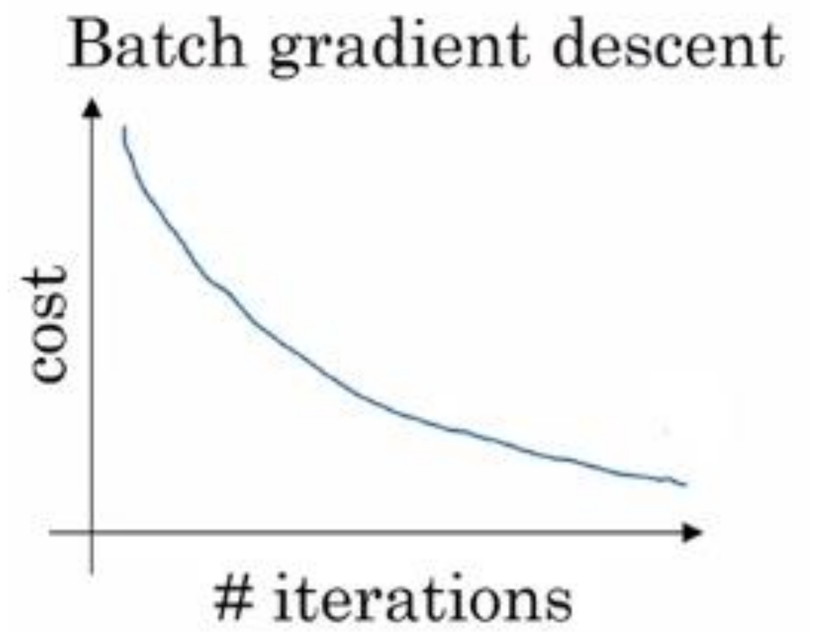
Usar un batch que consiste en algunos datos (no todos) para ajustar parámetros. **m < numero total patrones de entrenamiento**

$$\theta_j = \theta_j - \alpha \frac{\partial}{\partial \theta_j} J(\theta_0, \theta_1), j = 0{,}1$$

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

Comportamiento de diferentes variantes





Implementación

Python

(1) Intentar programar de cero (algunos casos)

(2) Uso de librería Scikit Learn
   - Poderoso librería de Python para aprendizaje automático
   - Combina fácilmente con librería de Deep-learning

Coeficientes de Determinación R^2

Score que entrega la función "LinearRegression"

Usando este valor, se puede evaluar eficiencia de regresión lineal.

$$R^2 = 1 - \frac{SSres}{SStot} \quad R^2 \in [0,1]$$

$$SSres = \sum_i e_i^2 = \sum_i (y_i - \widehat{y}_i)^2$$

$$SStot = \sum_i (y_i - \bar{y})^2$$

Donde $\widehat{y}_i$ es el valor estimado por regresión lineal de i-esimo variable independiente

$\bar{y}$ es el valor promedio de los variables dependientes $Y = [y_1, y_2, \cdots y_N]$

Repaso

Regresión lineal simple – **Estimar linealmente una variable (un valor) dando un variable independiente.**

Modelo de regresión lineal simple

Modelo: $\boldsymbol{y = h_\theta(x) = \theta_1 x + \theta_0}$

Parámetros del modelo : $\boldsymbol{\theta_0,\ \theta_1}$

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

Meta: $\boldsymbol{\min_{\theta_0, \theta_1} J(\theta_0, \theta_1)}$

Algoritmo para obtener parámetros óptimos: Gradiente descendente

$$\theta_j = \theta_j - \alpha \frac{\partial}{\partial \theta_j} J(\theta_0, \theta_1), j = 0,1 \qquad \alpha: tasa\ de\ aprendizaje$$

Métrica de evaluación: $\boldsymbol{R^2} \in [0,1]$ $\qquad R^2 = 1 - \frac{\sum_i (y_i - \widehat{y_i})^2}{\sum_i (y_i - \bar{y})^2}$

$\mathbf{y}$: datos reales , $\boldsymbol{\hat{y}}$: datos estimados, $\boldsymbol{\bar{y}}$: promedio de datos reales

Regresión Lineal por Sklearn

***class* sklearn.linear_model.LinearRegression(**, fit_intercept*=*True*, *copy_X*=*True*, *tol*=*1e-06*, *n_jobs*=*None*, *positive*=*False*)**

(1) Importar función

```
from sklearn.linear_model  import  LinearRegression
```

(2) Generar modelo

```
modelo = LinearRegression()   # Estan usando los parametros predeterminados
```

(3) Aplicar el modelo a los datos -- Ajustar parametros

```
modelo.fit(X,y)  # X: variable independiente, y variable dependiente
```

(4) Obtener valores estimados

```
y_pred =modelo.predict(X)  # predict es método, X: variable independiente
```

(5) Obtener score (R^2)

```
score =modelo.score(X, y)  # score es método, X: variable independiente, y: variable dependiente (GT)
```

(6) Obtener parámetros (intercepto e inclinación) de regresión

```
th0 =modelo.intercept_  # intercept_ es  atributo
th1 =modelo.coef_       # coef_ es atributo
```

Linear_regression_1.py

(1) Generar DataFrame desde un archivo

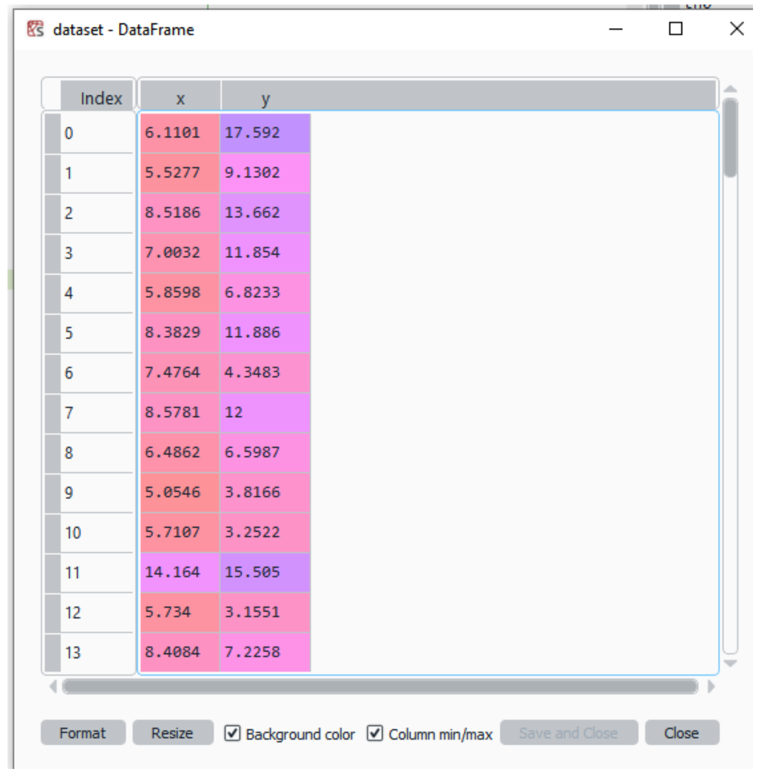
```
dataset = pd.read_csv('Data1.csv') # usando pandas, generar dataframe "dataset"
```

Data1.csv es un archivo de texto (Coma Separated Value)

Data1.csv

```
x,y
6.1101,17.592
5.5277,9.1302
8.5186,13.662
7.0032,11.854
5.8598,6.8233
8.3829,11.886
7.4764,4.3483
8.5781,12
6.4862,6.5987
:
```

pd.read_csv



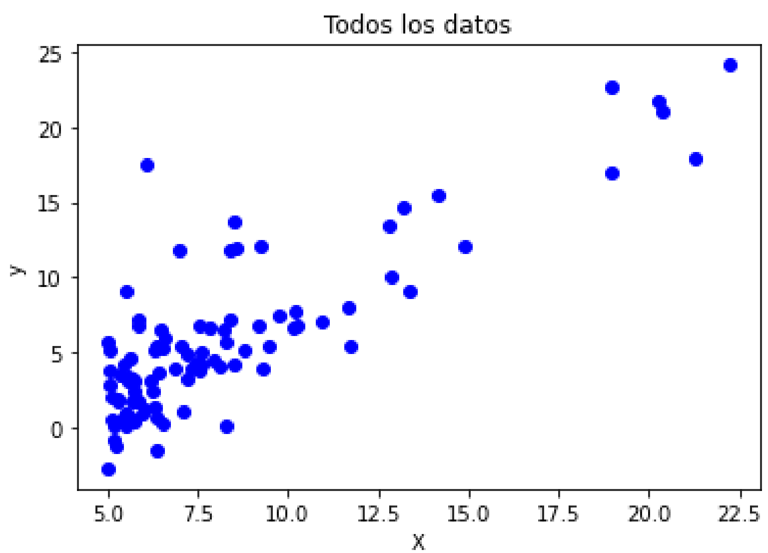
Index	x	y
0	6.1101	17.592
1	5.5277	9.1302
2	8.5186	13.662
3	7.0032	11.854
4	5.8598	6.8233
5	8.3829	11.886
6	7.4764	4.3483
7	8.5781	12
8	6.4862	6.5987
9	5.0546	3.8166
10	5.7107	3.2522
11	14.164	15.505
12	5.734	3.1551
13	8.4084	7.2258

(2) Obtener la variable independiente X y la variable dependiente y desde dataframe.

```
X = dataset.iloc[:, :-1].values     # Todos los rengulones, desde la primer columna hasta la penultima columna.  -1 significa última, a:b significa desde a hasta b-1
y = dataset.iloc[:, 1].values   # Todos los rengulones, la segunda columna
```

(3) Graficar los datos usando pyplot

```
plt.scatter(X,y,color="blue")
plt.xlabel('X')
plt.ylabel('y')
plt.title("Todos los datos")
plt.show()
```

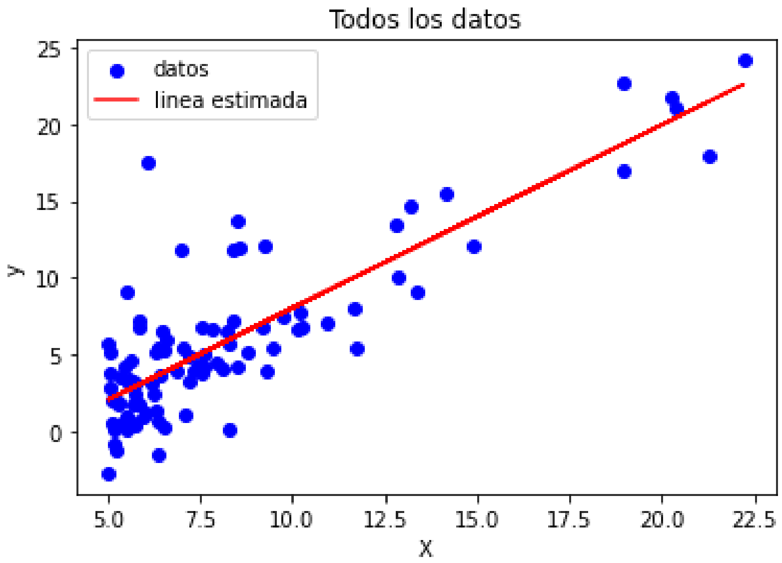


(4) Aplicar modelo de regresión linear a los datos (X y y) para obtener parámetros que representa una línea, y score.

```
from sklearn.linear_model import LinearRegression
modelo = LinearRegression()

modelo.fit(X,y)
y_pred = modelo.predict(X)

th0=modelo.intercept_
th1=modelo.coef_

print("Intercepto: ", th0)
print("Inclinación: ", th1[0])
print("score", modelo.score(X,y))
```

(5) Dibujar la línea que se representa los parámetros óptimos

```
plt.scatter(X,y,color="blue", label="datos")
plt.plot(X,th1*X+th0,color="red",label="linea estimada")
plt.xlabel('X')
plt.ylabel('y')
plt.title("Todos los datos")
plt.legend()
plt.show()
```



Linear_regression_2.py

- Básicamente el mismo que Linear_regression_1.py
- Única diferencia es la partición de datos en entrenamiento y prueba.

Separar datos en datos de entrenamiento y datos de prueba

```
from sklearn.model_selection import  train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 1/3,
                                                    random_state = 0)
```

Genera el model con los datos de entrenamiento

Ejercicios – Tarea 1

- **Obtener relación entre total_phenole y flavanoids**

Aplicar Regresión lineal de sklearn
(1) obtener función de línea
(2) dibujar la línea, junto con los datos
(3) obtener Score -- Coeficientes de Determinación R^2

- **Obtener relación entre hue y color_intensity**

(1) obtener función de línea
(2) dibujar la linea, junto con los datos
(3) obtener Score -- Coeficientes de Determinación R^2

- **Analizar relaciones entre dos factores (total_phenole y flavenoids) y (hue y color_intensity)**

- **Seleccionar dos factores de características de vinos que mejor relacionado linealmente.**

Wine dataset

Key	Type	Size	Value
data	Array of float64	(178, 13)	[[1.423e+01 1.710e+00 2.430e+00 ... 1.040e+00 3.920e+00 1.065e+03] [1 ...
DESCR	str	3449	.. _wine_dataset:
feature_names	list	13	['alcohol', 'malic_acid', 'ash', 'alcalinity_of_ash', 'magnesium', 'to ...
frame	NoneType	1	NoneType object
target	Array of int32	(178,)	[0 0 0 ... 2 2 2]
target_names	Array of str224	(3,)	ndarray object of numpy module

Inde	Type	Size	Value
0	str	7	alcohol
1	str	10	malic_acid
2	str	3	ash
3	str	17	alcalinity_of_ash
4	str	9	magnesium
5	str	13	total_phenols
6	str	10	flavanoids
7	str	20	nonflavanoid_phenols
8	str	15	proanthocyanins
9	str	15	color_intensity
10	str	3	hue
11	str	28	od280/od315_of_diluted_wines
12	str	7	proline

total_phenols (variable independiente) X

Flavanoids (variable dependiente) y

hue (variable independiente) X

color_intensity (variable dependiente) y