



Regresión Lineal Multivariable



Temas del día

1. Álgebra lineal
2. Regresión Lineal Multivariable
 - a. Regresión polinomial
 - b. Parámetros analíticamente
3. Clasificación
 - a. Regresión Logística
 - b. Clasificación multiclase
4. Overfitting y regularización



Álgebra Lineal



Matriz

- Arreglo rectangular de números

$$\begin{bmatrix} 14 & 7 \\ 3 & 2 \\ 8 & 14 \end{bmatrix}$$

$$\begin{bmatrix} 1 & 2 & 3 \\ 4 & 5 & 6 \end{bmatrix}$$

- Dimensiones: número de filas x número de columnas

Elementos de una matriz

A_{ij} = entrada de la fila i y la columna j

$$\begin{bmatrix} A_{11} & A_{12} & A_{13} \\ A_{21} & A_{22} & A_{23} \\ A_{31} & A_{32} & A_{33} \end{bmatrix}$$

Vector

- Matrix de $n \times 1$
- y_i = elemento i
- A veces usamos índice 1 y a veces índice 0
 - En ML se suele usar vectores con índice 0
- Matrices se declaran en mayúscula: A, B
- Vectores y escalares en minúscula: a, b, x

$$\begin{bmatrix} 460 \\ 232 \\ 315 \\ 178 \end{bmatrix}$$

Suma de matrices

- Deben tener las mismas dimensiones
- Se suma por elemento

$$\begin{bmatrix} 1 & 0 \\ 2 & 3 \\ 1 & 2 \end{bmatrix} + \begin{bmatrix} 0 & 2 \\ 2 & 1 \\ -5 & 0 \end{bmatrix} = \begin{bmatrix} 1 & 2 \\ 4 & 4 \\ -4 & 2 \end{bmatrix}$$

Multiplicación con escalar

$$3 \times \begin{bmatrix} 1 & 0 \\ 2 & 5 \end{bmatrix} = \begin{bmatrix} 3 & 0 \\ 6 & 15 \end{bmatrix}$$

Multiplicación Vector con Matriz

$$\begin{bmatrix} 1 & 3 \\ 4 & 0 \\ 2 & 1 \end{bmatrix} \times \begin{bmatrix} 1 \\ 5 \end{bmatrix} = \begin{bmatrix} (1)(1) + (3)(5) \\ (4)(1) + (0)(5) \\ (2)(1) + (1)(5) \end{bmatrix} = \begin{bmatrix} 16 \\ 4 \\ 7 \end{bmatrix}$$

- Para conseguir y_i , multiplicar la fila i de A con los elementos del vector x , y agregar esos resultados.
- Si A tiene dimensiones $m \times n$, y x tiene dimensiones $n \times 1$, y tiene dimensiones $m \times 1$.

Aplicación

- Tamaños de las casas: [2104, 1416, 1534, 852]
- Hipótesis: $h_{\theta}(x) = -40 + 0.25x$
- Predicción: Matriz de Datos x parámetros

$$\begin{bmatrix} 1 & 2104 \\ 1 & 1416 \\ 1 & 1534 \\ 1 & 852 \end{bmatrix} \times \begin{bmatrix} -40 \\ 0.25 \end{bmatrix} = \begin{bmatrix} h_{\theta}(2104) \\ h_{\theta}(1416) \\ h_{\theta}(1539) \\ h_{\theta}(852) \end{bmatrix}$$

Multiplicación Matriz por Matriz ($A \times B = C$)

- La columna i de la matriz C se obtiene por multiplicar A con la columna i de B .
- Se puede ver como descomponer B en sus columnas.

$$\begin{bmatrix} 1 & 3 & 2 \\ 4 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 3 \\ 0 & 1 \\ 5 & 2 \end{bmatrix} = \left[\begin{bmatrix} 1 & 3 & 2 \\ 4 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 \\ 0 \\ 5 \end{bmatrix} \text{ append } \begin{bmatrix} 1 & 3 & 2 \\ 4 & 0 & 1 \end{bmatrix} \begin{bmatrix} 3 \\ 1 \\ 2 \end{bmatrix} \right] = \begin{bmatrix} 11 & 10 \\ 9 & 14 \end{bmatrix}$$

Aplicación

- Tamaños de las casas: [2104, 1416, 1534, 852]
- Varias hipótesis
 - Hipótesis: $h_{\theta}(x) = -40 + 0.25x$
 - Hipótesis: $h_{\theta}(x) = 200 + 0.1x$
 - Hipótesis: $h_{\theta}(x) = -150 + 0.4x$

$$\begin{bmatrix} 1 & 2104 \\ 1 & 1416 \\ 1 & 1534 \\ 1 & 852 \end{bmatrix} \times \begin{bmatrix} -40 & 200 & -150 \\ 0.25 & 0.1 & 0.4 \end{bmatrix} = \begin{bmatrix} 486 & 410 & 692 \\ 314 & 342 & 416 \\ 344 & 353 & 464 \\ 173 & 285 & 191 \end{bmatrix}$$

Propiedades de multiplicación

- No es conmutativa $A \times B \neq B \times A$
- Es asociativa $A \times B \times C = (A \times B) \times C = A \times (B \times C)$
- Matriz identidad
 - 1 es la identidad de los reales: $1 \times 7 = 7$
 - I o $I_{n \times n}$ es la de matrices
- $A \times I = I \times A = A$

$$\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

Matriz inversa


- En los reales $3 * (1/3) = 1$
 - No todos tienen inverso (0)
- Si A es una matriz cuadrada y tiene inversa $AA^{-1} = A^{-1}A = I$

$$\begin{bmatrix} 3 & 4 \\ 2 & 10 \end{bmatrix} \begin{bmatrix} 0.4 & -0.1 \\ -0.05 & 0.075 \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$


Matriz transpuesta

- Si A es una matriz $\mathbf{m \times n}$, y definimos $B = A^T$, entonces B es una matriz $\mathbf{m \times m}$ y $B_{ij} = A_{ji}$

$$A = \begin{bmatrix} 1 & 2 & 0 \\ 3 & 5 & 9 \end{bmatrix} A^T = \begin{bmatrix} 1 & 3 \\ 2 & 5 \\ 0 & 9 \end{bmatrix}$$



Regresión Lineal con multiples variables



Regresión Lineal

Supervised Learning: Regression

training set

Observation #	Years of Higher Education (X)	Income (Y)
1	4	\$80,000
2	5	\$91,500
3	0	\$42,000
4	2	\$55,000
...
N	6	\$100,000

test set

1	4	???
2	6	???

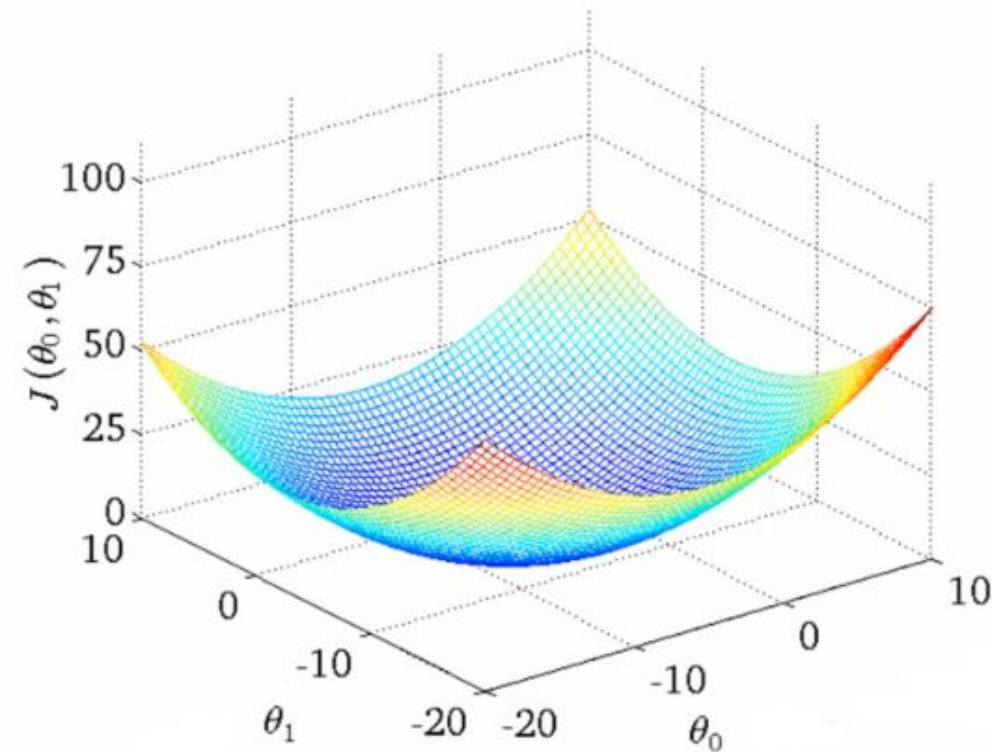
Loss Function (1)

$$J(\theta_0, \theta_1) = \frac{1}{2m} \sum_{i=1}^m (\hat{y}_i - y_i)^2 = \frac{1}{2m} \sum_{i=1}^m (h_{\theta}(x_i) - y_i)^2$$

- Objetivo: Elegir θ_0 y θ_1 de manera que $h_{\theta}(x)$ se acerque a la y real en nuestros ejemplos de entrenamiento (x,y) , es decir, que minimicen la función de pérdida J .
- En este caso usamos Mean Squared Error, pero hay otras.

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

Gradient Descent



Repeat until convergence {

$$\theta_j \leftarrow \theta_j - \alpha \frac{\partial}{\partial \theta_j} J(\theta)$$

}

Gradient Descent para MSE

$$J(\theta_0, \theta_1) = \frac{1}{2m} \sum_{i=1}^m (\hat{y}_i - y_i)^2 = \frac{1}{2m} \sum_{i=1}^m (h_{\theta}(x_i) - y_i)^2$$

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

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

Algoritmo de Gradient Descent

$$\text{temp0} := \theta_0 - \alpha \frac{\partial}{\partial \theta_0} J(\theta_0, \theta_1)$$

$$\text{temp1} := \theta_1 - \alpha \frac{\partial}{\partial \theta_1} J(\theta_0, \theta_1)$$

$$\theta_0 := \text{temp0}$$

$$\theta_1 := \text{temp1}$$

Antes

Tamaño (x)	Costo (y)
2100	400
1416	232
...	...

Ahora

Tamaño x1	# cuartos x2	pisos x3	Edad x4	Costo (y)
2104	5	1	45	460
1416	3	2	40	232

Conceptos

- De antes
 - m = # de training samples
 - x = variables de entrada o features
 - y = variable de salida
 - (x, y) un ejemplo de entrenamiento
 - $(x^{(i)}, y^{(i)})$ - i^{th} training sample
- Adicionalmente
 - n = # de features
 - $x^{(i)}$ = entradas (features) del sample o ejemplo i
 - $x_j^{(i)}$ = valor del feature j en el sample o ejemplo i

Tamaño x1	# cuartos x2	pisos x3	Edad x4	Costo (y)
2104	5	1	45	460
1416	3	2	40	232

$$x^{(2)} = \begin{bmatrix} 1416 \\ 3 \\ 2 \\ 40 \end{bmatrix}$$

$$x_3^{(2)} = 2$$

Antes

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

Ahora

$$h_{\theta}(x) = \theta_0 + \theta_1 x_1 + \theta_2 x_2 + \dots + \theta_n x_n$$

Hipótesis

$$h_{\theta}(x) = \theta_0 + \theta_0 x_1 + \theta_2 x_2 + \dots + \theta_n x_n$$

- Definimos x_0 como 1 ($x_0 = 1$)

$$x = \begin{bmatrix} x_0 \\ x_1 \\ \dots \\ x_n \end{bmatrix} \quad \theta = \begin{bmatrix} \theta_0 \\ \theta_1 \\ \dots \\ \theta_n \end{bmatrix}$$

- Por lo tanto, usando multiplicación de matrices

$$h_{\theta} = \theta^T x$$

Hipótesis

$$h_{\theta}(x) = \theta_0 + \theta_0 x_1 + \theta_2 x_2 + \dots + \theta_n x_n$$

- Definimos x_0 como 1 ($x_0 = 1$)

$$x = \begin{bmatrix} x_0 \\ x_1 \\ \dots \\ x_n \end{bmatrix} \quad \theta = \begin{bmatrix} \theta_0 \\ \theta_1 \\ \dots \\ \theta_n \end{bmatrix}$$

- Por lo tanto, usando multiplicación de matrices

$$h_{\theta}(x) = \theta^T x \qquad \begin{bmatrix} \theta_0 & \dots & \theta_n \end{bmatrix} \begin{bmatrix} x_0 \\ \dots \\ x_n \end{bmatrix}$$

Gradient Descent

- Hipótesis $h_{\theta}(x) = \theta^T x$
- Parámetros Θ vector $n+1$
- Función de costo?
- Gradient Descent?

Antes

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

Ahora

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

Antes

$$J(\theta_0, \theta_1) = \frac{1}{2m} \sum_{i=1}^m (\hat{y}_i - y_i)^2 = \frac{1}{2m} \sum_{i=1}^m (h_{\theta}(x_i) - y_i)^2$$

Repeat {

$$\theta_0 := \theta_0 - \alpha \underbrace{\frac{1}{m} \sum_{i=1}^m (h_{\theta}(x^{(i)}) - y^{(i)})}_{\frac{\partial}{\partial \theta_0} J(\theta)}$$

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

(simultaneously update θ_0, θ_1)

}

Ahora

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

Repeat {

$$\theta_j := \theta_j - \alpha \frac{1}{m} \sum_{i=1}^m (h_{\theta}(x^{(i)}) - y^{(i)}) x_j^{(i)}$$

(simultaneously update θ_j for
 $j = 0, \dots, n$)

}

jListo!



Feature Scaling



Feature Scaling

- Idea: Asegurarse que los features tengan la misma escala

x_1 = tamaño (0-500 metros)

x_2 = # de cuartos (máximo 5)

- Esto hace que Gradient Descent se tarde más tiempo en encontrar un mínimo
- ¿Qué pueden hacer?

Feature Scaling

x_1 = tamaño (0-500 metros)

x_2 = # de cuartos

- Los convertimos a la misma escala (0 a 1)

$$x_1 = \frac{\text{tamano}}{2000} \quad x_2 = \frac{\text{numcuartos}}{5}$$

Feature Scaling

- No tienen que tener misma escala, pero se sugiere que sea similar
- El rango más aceptado es de -1 a 1
 - Pero que los números no sean demasiado pequeños tampoco (0.00001)

Mean Normalisation

- Idea: Hacer que el promedio sea 0
- No se aplica a X_0 porque su valor es 1
- Reemplazamos los valores de los features

$$\bar{X}_i = X_i - \mu_i$$

Técnicas de hacer feature scaling

- Reescalar

$$x' = \frac{x - \min(x)}{\max(x) - \min(x)}$$

- Estandarización

$$x' = \frac{x - \bar{x}}{\sigma}$$

- Mean Normalisation

$$x' = \frac{x - \text{mean}(x)}{\max(x) - \min(x)}$$

¿Por qué?

- Los parámetros descienden rápidamente en rangos bajos y lentamente en rangos altos
 - Oscila ineficientemente para llegar al óptimo si las variables no están en escalas similares

Learning Rate

Repeat until convergence {

$$\theta_j \leftarrow \theta_j - \alpha \frac{\partial}{\partial \theta_j} J(\theta)$$

}

Learning Rate

Repeat until convergence {

$$\theta_j \leftarrow \theta_j - \alpha \frac{\partial}{\partial \theta_j} J(\theta)$$

}

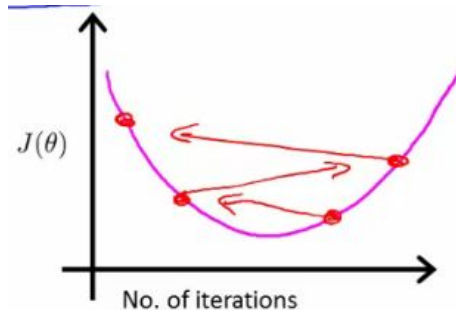
- Nosotros definimos el learning rate.
- Debe ser pequeño
- ¿Cómo graficamos el error vs número de iteraciones?

Learning Rate

- ¿Cómo graficamos el error vs número de iteraciones?
- El error $J(\theta)$ debe bajar en cada iteración
- Podemos agregar pruebas de convergencia automática
 - Si $J(\theta)$ disminuye menos que ϵ en una iteración
- ¿Qué pasa si el error aumenta en las iteraciones?

Learning Rate

- ¿Cómo graficamos el error vs número de iteraciones?
- El error $J(\theta)$ debe bajar en cada iteración
- Podemos agregar pruebas de convergencia automática
 - Si $J(\theta)$ disminuye menos que ϵ en una iteración
- ¿Qué pasa si el error aumenta en las iteraciones?



Learning Rate

- ¿Cómo graficamos el error vs número de iteraciones?
- El error $J(\theta)$ debe bajar en cada iteración
- Podemos agregar pruebas de convergencia automática
 - Si $J(\theta)$ disminuye menos que ϵ en una iteración
- Esto es lo que nos asegura un learning rate pequeño
 - Si es muy pequeño, el algoritmo es lento
 - Converge lento
-

Para definir Learning Rate

- Prueba un rango de valores: 0.001, 0.003, 0.01, 0.03, 0.1, 0.3
- Grafica la pérdida vs learning rate, y revisa hasta iteración 100 como máximo en problemas sencillos.
- Agrega pruebas de convergencia
- Nota en cuanto a número de iteraciones: varía mucho y no hay reglas fijas. Algunos problemas requieren 10 y otros 3,000,000.



Regresión Polinomial



Regresión Polinomial

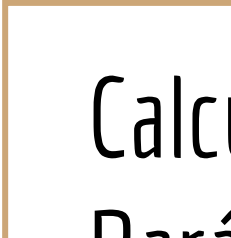
- ¿Cómo hacemos si nuestros datos no se comportan de manera lineal?
- Hay trucos para que funcione
- El más sencillo y común es el siguiente.

Feature 1 - tamaño


Feature 2 - $(\text{tamaño})^2$

Feature 3 - $(\text{tamaño})^3$

- Es necesario hacer feature scaling



Calculando los Parámetros Analíticamente



Ecuación normal

- Método para resolver θ analíticamente.

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

- ¿Cómo podemos calcular el mínimo de una función?

Ecuación normal

- Método para resolver θ analíticamente.

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

- ¿Cómo podemos calcular el mínimo de una función?
 - Cuando su derivada es 0 para cada j

$$\frac{\delta}{\delta \theta_j} J(\Theta) = \dots = 0$$

Un ejemplo donde $m=4$

$$X = \begin{bmatrix} 1 & 2104 & 5 & 1 & 45 \\ 1 & 1416 & 3 & 2 & 40 \\ 1 & 1534 & 3 & 2 & 30 \\ 1 & 852 & 2 & 1 & 36 \end{bmatrix}$$

$$y = \begin{bmatrix} 460 \\ 232 \\ 315 \\ 178 \end{bmatrix}$$

- X se llama matriz de diseño en este caso
- ¿Cómo se crea?

Matriz de diseño

- $x^{(i)}$ =entradas (features) del sample o ejemplo i
- $x_j^{(i)}$ = valor del feature j en el sample o ejemplo i

$$x^{(i)} = \begin{bmatrix} x_0^{(i)} \\ \dots \\ x_n^{(i)} \end{bmatrix} \quad X = \begin{bmatrix} \dots (x^{(1)})^T \dots \\ \dots \\ \dots (x^{(m)})^T \dots \end{bmatrix} \quad X = \begin{bmatrix} 1 & 2104 & 5 & 1 & 45 \\ 1 & 1416 & 3 & 2 & 40 \\ 1 & 1534 & 3 & 2 & 30 \\ 1 & 852 & 2 & 1 & 36 \end{bmatrix}$$

Calculamos analíticamente

$$\Theta = (X^T X)^{-1} X^T y$$

Gradient Descent

- Necesitas elegir learning rate
- Necesita iterar

Ecuación Normal

- No necesitas learning rate
- No se necesita iterar

Gradient Descent

- Necesitas elegir learning rate
- Necesita iterar
- Funciona muy bien cuando n es largo (muchos features)
 - Algoritmo $O(n^2)$
- Funciona bien para muchas técnicas de ML
 - Regresión Logística
 - Redes Neuronales

Ecuación Normal

- No necesitas learning rate
- No se necesita iterar
- Funciona mal cuando n es largo (muchos features)
 - Algoritmo $O(n^3)$
 - Calcular inversas es muy costoso computacionalmente

Ecuación normal $\Theta = (X^T X)^{-1} X^T y$

- ¿Qué pasa si $X^T X$ no es invertible?
 1. Features redundantes (crea dependencia linear)
 - a. Si un feature es el tamaño en metros y el otro en pies
 2. Si hay demasiados features ($m < n$)
 - a. Significa que no hay suficientes datos.
 - b. Podemos eliminar features o utilizar regularización