



deeplearning.ai

Basics of Neural Network Programming

Vectorizing Logistic Regression

Vectorizing Logistic Regression

$$\begin{aligned} \Rightarrow z^{(1)} &= w^T x^{(1)} + b \\ \Rightarrow a^{(1)} &= \sigma(z^{(1)}) \end{aligned}$$

$$\underline{z^{(2)}} = \boxed{w^T x^{(2)} + b}$$

$$\boxed{a^{(2)}} = \sigma(z^{(2)})$$

$$\underline{z^{(3)}} = w^T x^{(3)} + b$$

$$\underline{a^{(3)}} = \sigma(z^{(3)})$$

$$\underline{\underline{X}} = \begin{bmatrix} | & | & & | \\ X^{(1)} & X^{(2)} & \dots & X^{(m)} \\ | & | & & | \end{bmatrix}$$

$$\frac{(n_x, m)}{\mathbb{R}^{n_x \times m}}$$

$$I - \omega^T \begin{bmatrix} x^{(1)} & x^{(2)} & \dots & x^{(n)} \\ 1 & 1 & \dots & 1 \end{bmatrix}$$

$$\underline{Z} = \begin{bmatrix} \underline{z}^{(1)} & \underline{z}^{(2)} & \dots & \underline{z}^{(m)} \end{bmatrix} = \underbrace{w^T X}_{1 \times m} + \underbrace{[b \ b \dots b]}_{1 \times m} = \boxed{w^T x^{(1)} + b} \quad \boxed{w^T x^{(2)} + b} \quad \dots \quad \boxed{w^T x^{(m)} + b}$$

$$\rightarrow \underline{z = np.dot(w.T, x) + \frac{b}{K} \quad (1,1) \quad \mathbb{R}}$$

$$\underline{A} = [\underline{a^{(1)}} \quad \underline{a^{(2)}} \quad \dots \quad \underline{a^{(m)}}] = \underline{\sigma(z)}$$

python automatically take this real number and expands it out to this 1bym row vector.



deeplearning.ai

Basics of Neural Network Programming

Vectorizing Logistic Regression's Gradient Computation

Vectorizing Logistic Regression

$$dz^{(1)} = a^{(1)} - y^{(1)}$$

$$dz^{(2)} = a^{(2)} - y^{(2)}$$

.....

$$dz = [dz^{(1)} \quad dz^{(2)} \quad \dots \quad dz^{(m)}]$$

$1 \times m$

←

$$A = [a^{(1)} \quad \dots \quad a^{(m)}] \quad Y = [y^{(1)} \quad \dots \quad y^{(m)}]$$

$$\rightarrow dz = A - Y = [a^{(1)} - y^{(1)} \quad a^{(2)} - y^{(2)} \quad \dots]$$

$$\rightarrow dw = 0$$

$$dw += \frac{x^{(1)} dz^{(1)}}{m}$$

$$dw += \frac{x^{(2)} dz^{(2)}}{m}$$

⋮

$$dw/ = m$$

$$db = 0$$

$$db += dz^{(1)}$$

$$db += dz^{(2)}$$

$$\vdots$$

$$db += dz^{(m)}$$

$$db/ = m$$

$$db = \frac{1}{m} \sum_{i=1}^m dz^{(i)}$$

$$= \frac{1}{m} \text{np.sum}(dz)$$

$$dw = \frac{1}{m} X dz^T$$

$$= \frac{1}{m} \begin{bmatrix} x^{(1)} & \dots & x^{(m)} \\ 1 & & 1 \end{bmatrix} \begin{bmatrix} dz^{(1)} \\ \vdots \\ dz^{(m)} \end{bmatrix}$$

$$= \frac{1}{m} \left[\underbrace{x^{(1)} dz^{(1)}}_{n \times 1} + \dots + \underbrace{x^{(m)} dz^{(m)}}_{n \times 1} \right]$$

Implementing Logistic Regression

$J = 0, dw_1 = 0, dw_2 = 0, db = 0$

for $i = 1$ to m :

$$z^{(i)} = w^T x^{(i)} + b \leftarrow$$

$$a^{(i)} = \sigma(z^{(i)}) \leftarrow$$

$$J += -[y^{(i)} \log a^{(i)} + (1 - y^{(i)}) \log(1 - a^{(i)})]$$

$$dz^{(i)} = a^{(i)} - y^{(i)} \leftarrow$$

$$\left[\begin{array}{l} dw_1 += x_1^{(i)} dz^{(i)} \\ dw_2 += x_2^{(i)} dz^{(i)} \\ db += dz^{(i)} \end{array} \right] \quad dw += x^{(i)} * dz^{(i)}$$

$$J = J/m, dw_1 = dw_1/m, dw_2 = dw_2/m$$

$$db = db/m$$

for iter in range(1000): \leftarrow

$$Z = w^T X + b$$

$$= \text{np.dot}(w.T, X) + b$$

$$A = \sigma(Z)$$

$$dZ = A - Y$$

$$dw = \frac{1}{m} X dZ^T$$

$$db = \frac{1}{m} \text{np.sum}(dZ)$$

$$w := w - \alpha dw$$

$$b := b - \alpha db$$

If you want to implement multiple iteration as a gradient then you still need a for loop over the number of iterations.