deeplearning.ai

Basics of Neural Network Programming

Vectorizing Logistic Regression

Vectorizing Logistic Regression

$z^{(1)} = w^T x^{(1)} + b$ $z^{(2)} = w^T x^{(2)} + b$ $z^{(3)} = w^T x^{(3)} + b$

$a^{(1)} = \sigma(z^{(1)})$ $a^{(2)} = \sigma(z^{(2)})$ $a^{(3)} = \sigma(z^{(3)})$

Aim: Remove All For loops in Gradient descent, there is still 1 for loop

$$X = \begin{bmatrix} \vdots & \vdots & \vdots & & \vdots \\ x^{(1)} & x^{(2)} & x^{(3)} & \cdots & x^{(m)} \\ \vdots & \vdots & \vdots & & \vdots \end{bmatrix} \in \mathbb{R}^{n_x \times m}$$

n_x features per sample

m samples

$$\underset{1\times m}{Z} = [z^{(1)}, z^{(2)} \dots z^{(m)}]_{1\times m} = w^T \cdot X + \underbrace{[b, b, b, \dots b]}_{1 \times m} = [\underbrace{w^T x^{(1)} + b}_{z^{(1)}}, \underbrace{w^T x^{(2)} + b}_{z^{(2)}}, \underbrace{w^T x^{(3)} + b}_{z^{(3)}}, \dots]$$

$z^{(3)} \dots z^{(m)}$

Here w^T is a row vector $1 \times n_x$ (weight for all features)

$$= (1 \times n_x)(n_x \times m) + (1 \times m)$$
$$= 1 \times m$$

In python

$\hookrightarrow$ $Z = np.dot(w.T, X) + b$ → technically b is defined as a (1,1) vector, so how can you add it to a m sized vector?

Broadcasting!

Andrew Ng

deeplearning.ai

Basics of Neural Network Programming

Vectorizing Logistic Regression's Gradient Computation

Vectorizing Logistic Regression

$dz^{(1)} = a^{(1)} - y^{(1)}$, $dz^{(2)} = a^{(2)} - y^{(2)}$

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

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

$dZ = A - Y = [a^{(1)} - y^{(1)}, a^{(2)} - y^{(2)} \dots a^{(m)} - y^{(m)}]$

So we'd gotten rid of 1 loop (The inner loop $1 \dots n_x$)

What about the outer loop?

Implementing Logistic Regression

Here, we can have

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

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

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

$$= \frac{1}{m} \begin{bmatrix} | & | & & | \\ x^{(1)} & x^{(2)} & \dots & X^{(m)} \\ | & | & & | \end{bmatrix} \cdot \begin{bmatrix} dz^{(1)} \\ \vdots \\ dz^{(m)} \end{bmatrix}$$

$$= \frac{1}{m} \left[X^{(1)} \cdot dz^{(1)} + X^{(2)} \cdot dz^{(2)} + \dots X^{(m)} \cdot dz^{(m)} \right]$$

$$= \frac{1}{m} np.dot(X, dz.T)$$

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

for i = 1 to m:

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

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

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

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

$$dw \mathrel{+}= X^{(i)} \cdot dz^{(i)}$$

$$db \mathrel{+}= dz^{(i)}$$

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

db = db/m

Gone

Final code

$$Z = np.dot(w.T, X) + b$$

$$A = \sigma(Z)$$

$$dz = A - Y$$

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

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

update weights

$$w = w - \alpha \cdot dw$$

$$b = b - \alpha \cdot db$$

Remember this was 1 iteration of m samples, this outer most for loop cant be removed

Andrew Ng