3) Vectorization (Hızı çok artırır)

$z = w^T x + b$ $\quad w = \begin{bmatrix} \vdots \end{bmatrix}$ $\quad x = \begin{bmatrix} \vdots \end{bmatrix}$

Non-vektorized

$z = 0$

for i in range (n-x):

$\quad z += w^{(i)} * x^{(i)}$

$z += b$

Vektorized

In numpy;

$z = \underbrace{np.dot(w, x)}_{w^T x} + b$

numpy, paralel hesaplamada çok iyidir.

For loop kullanma, numpy ile derleme hızını artır; yani Numpy kullan.

! Whenever possible, avoid explicit for-loops.

* Vektörizasyon, for loop kullanmak yerine vektörler oluşturarak problemi çözmektir.

Vectorising Logistic Regression's Gradient Computation

$$dz^{(1)} = a^{(1)} - y^{(1)} \qquad dz^{(2)} = a^{(2)} - y^{(2)} \dots$$

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

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

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

$\rightarrow$ $dw = 0$ $\qquad$ $db = 0$

$dw \mathrel{+}= x^{(1)} dz^{(1)}$ $\qquad$ $db \mathrel{+}= dz^{(1)}$

$dw \mathrel{+}= x^{(2)} dz^{(2)}$ $\qquad$ $db \mathrel{+}= dz^{(2)}$

$\vdots$ $\qquad$ $\vdots$

$dw \mathrel{/}= m$ $\qquad$ $db \mathrel{/}= m$

* $db = \frac{1}{m} \sum_{i=1}^{m} dz^{(i)} \Rightarrow \frac{1}{m}$ np.sum(dZ)

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

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

$$= \frac{1}{m} [x^{(1)} dz^{(1)} + \dots + x^{(m)} dz^{(m)}]$$

Result:

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

$A = \sigma(Z)$

$dZ = A - Y$

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

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

$$\begin{pmatrix} w := w - \alpha dw \\ b := b - \alpha db \end{pmatrix}$$