

Vectorization is useful in getting rid of for loop to run algorithm faster.

$$z = w^T x + b \quad w = \begin{pmatrix} \vdots \\ \vdots \\ \vdots \end{pmatrix}_{\mathbb{R}^{n \times 1}} \quad x = \begin{pmatrix} \vdots \\ \vdots \\ \vdots \end{pmatrix}_{\mathbb{R}^{n \times 1}}$$

Non Vectorized:

```
z = 0
for i in range(n-x):
    z += w[i] * x[i]
z += b
```

Vectorized: (numpy)

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

Avoid for loops whenever possible

$$u = A \cdot v$$
$$u_i = \sum_j A_{ij} v_j$$

$$u = \text{np.dot}(A, v)$$

$$u = \text{np.zeros}(n, 1)$$

for i ---


```
for j - ...  
    u_i += A_ij * v_j
```

Vectors & Matrix valued Function:

e.g. Need to apply exponential operation,
on every element of matrix.

$$v = \begin{bmatrix} v_1 \\ v_2 \\ \vdots \\ v_n \end{bmatrix} \quad u = \begin{bmatrix} e^{v_1} \\ e^{v_2} \\ \vdots \\ e^{v_n} \end{bmatrix}$$

```
u = np.zeros((n,1))  
for i in range(n):  
    u[i] = math.exp(v[i])
```

Using Numpy:

```
import numpy as np.  
u = np.exp(v)
```

e.g. np.log , maximum
abs

Logistic Regression Derivatives:

$$dw = np.zeros((n-x, 1))$$

$$dw + = x^{(i)} dz^{(i)}$$

$$dw \big| = m$$

Vectorizing Logistic Regression:

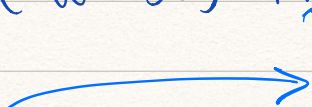
$$\begin{aligned} z^{(1)} &= w^T x^{(1)} + b & z^{(2)} &= w^T x^{(2)} + b & z^{(3)} &= w^T x^{(3)} + b \\ a^{(1)} &= \sigma(z^{(1)}) & a^{(2)} &= \sigma(z^{(2)}) & a^{(3)} &= \sigma(z^{(3)}) \end{aligned}$$

$$Z = [z^{(1)} \quad z^{(2)} \quad \dots \quad z^{(m)}] = W^T X + [b \quad b \quad \dots \quad b]$$

$1 \times m$

$$z = np.dot(w^T x) + b$$

Broadcasting

 Real number
 1×1

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

Vectorising logistic Reg's Gradient Output:

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

$$\begin{aligned} dz &= [dz^{(1)} \quad dz^{(2)} \quad \dots \quad dz^{(m)}] \\ A &= [a^{(1)} \quad a^{(2)} \quad \dots \quad a^{(m)}] \\ y &= [y^{(1)} \quad y^{(2)} \quad \dots \quad y^{(m)}] \end{aligned}$$

$$dz = A - y$$

$$\begin{aligned} dw &= 0 \\ dw + &= x^{(1)} dz^{(1)} \\ dw + &= x^{(2)} dz^{(2)} \end{aligned}$$

$$\vdots \\ dw / = m$$

$$\begin{aligned} db &= 0 \\ db + &= dz^{(1)} \\ db + &= dz^{(2)} \end{aligned}$$

$$\vdots \\ db / = m$$

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

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

$$= \frac{1}{m} \begin{bmatrix} x_1 & x_2 & \dots & x_m \\ \vdots & \vdots & \ddots & \vdots \end{bmatrix} \begin{bmatrix} dz^{(1)} \\ \vdots \\ dz^{(m)} \end{bmatrix} = \frac{1}{m} \text{np.sum}(dz)$$

Summing all up:

for i in range(10000):

$$\begin{aligned} z &= w^T x + b \\ &= \text{np.dot}(w^T x) + b \end{aligned}$$

$$A = \sigma(z)$$

$$dz = A - Y$$

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

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

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

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