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

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

$\mathbb{R}^{n_x}$ $\mathbb{R}^{n_x}$

Non Vectorized:

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

Vectorized: (numpy)

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

Avoid for loops whenever possible

$u = AV$ $u = np.dot(A, V)$

$u_i = A_{ij} V_j$

u = np.zeros((n,1))

for i - - -

for j - . .
  u_i += $A_{ij} \cdot v_j$

Vectors & Matrix valued Function:

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

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

```
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:

$$dw = np.zeros((n_x, 1))$$

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

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

Vectorizing Logistic Regression:

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

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

$$Z = [z^{(1)} \; z^{(2)} \ldots z^{(m)}] = w^T X + \underbrace{[b \; b \; b \ldots b]}_{1 \times m}$$

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

Broadcasting → Real number 1×1

$$A = [a^{(1)} \; a^{(2)} \ldots a^{(m)}] = \sigma(Z)$$

Vectorising logistic Reg.'s Gradient Output:

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

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

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

$Y = [y^{(1)} \; y^{(2)} \; \dots \; y^{(m)}]$

$dZ = A - Y$

$dw = 0$

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

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

$\vdots$

$dw \mathrel{/}= m$

$db = 0$

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

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

$\vdots$

$db \mathrel{/}= m$

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

$= \frac{1}{m} \begin{bmatrix} \vdots & \vdots & \vdots \\ x_1 & x_2 & x_m \\ \vdots & \vdots & \vdots \end{bmatrix} \begin{bmatrix} dz^{(1)} \\ \vdots \\ dz^{(m)} \end{bmatrix}$

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

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

Summing all up:

for i in range (10,000):

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

$$A = \sigma(z)$$
$$dz = A - Y$$
$$dw = \frac{1}{m} X dz^T$$

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

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