What is vectorization

$z = w^T x + b$

$w = \begin{bmatrix} w_1 \\ \vdots \\ w_n \end{bmatrix}$ $x = \begin{bmatrix} x_1 \\ \vdots \\ x_n \end{bmatrix}$ $w \in \mathbb{R}^n$, $x \in \mathbb{R}^n$

non-vectorized

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

Vectorized

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

In NN avoid explicit for-loops, use vectorization

$u = Av$

$u_i = \sum_j A_{ij} v_j$

```
u = np.zeros((n,1))
for i ...
    for j ...
        u[i] += A[i][j] * v[j]
```

$u = np.dot(A, v)$

$v = \begin{bmatrix} v1 \\ v2 \\ v3 \end{bmatrix}$ $U = \begin{bmatrix} e^{v_1} \\ e^{v_2} \\ \vdots \\ e^{v_n} \end{bmatrix}$

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

```
import numpy as np
u = np.exp(v)
np.log(v)
np.abs(v)
np.maximum(v,0)
v**2
1/v
```

J = 0, ~~dw1 = 0, dw2 = 0~~, db = 0 → $dw = np.zeros((n_x, 1))$

for i = 1 to m:

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

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

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

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

$\cancel{dw_1 \mathrel{+}= x_1^{(i)} dz^{(i)}}$

$\cancel{dw_2 \mathrel{+}= x_2^{(i)} dz^{(i)}}$ → $dw \mathrel{+}= x^{(i)} dz^{(i)}$

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

→ $dw /= m$

V17+18

Vectorizing Logistic Regression

z^(1) = w^T*x^(1)+b   z^(2) = w^T*x^(2)+b

a^(1) = sig(z^(1))   a^(2) = sig(z^(2))

... to m training examples

V19

$X = \begin{bmatrix} | & | & & | \\ x^{(1)} & x^{(2)} & \cdots & x^{(m)} \\ | & | & & | \end{bmatrix}$ (n_x, m) $\mathbb{R}^{n_x, m}$

$w^T \begin{bmatrix} | & | & & | \\ x^{(1)} & x^{(2)} & \cdots & x^{(m)} \\ | & | & & | \end{bmatrix}$

$[z^{(1)}, z^{(2)}, \ldots, z^{(m)}] = w^T X + \underbrace{[b, b, \ldots, b]}_{1 \times m} = [w^T x^{(1)} + b_1, \ldots, w^T x^{(m)} + b_2]$

$= Z$ → $Z = np.dot(w.T, x) + b$

Vectorizing Logistic Regression

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

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

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

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

$dw = 0$

$dw_1 \mathrel{+}= x^{(1)} dz$

$\vdots$

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

$\vdots$

$dw /= m$

$db = 0$

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

$\vdots$

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

$db /= m$

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

$\boxed{dw = 1/m * X dz^T}$

$= 1/m \begin{bmatrix} | & & | \\ x^{(1)} & \cdots & x^{(m)} \\ | & & | \end{bmatrix} \begin{bmatrix} dz^{(1)} \\ \vdots \\ dz^{(m)} \end{bmatrix}$

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

$Z = w^T x + b$

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

$A = \sigma(Z)$

$dZ = A - Y$

$dw = 1/m\, X dZ^T$

$db = 1/m\, np.sum(dZ)$

$w := w - \alpha dw$

$b := b - \alpha d$

for number of iterations of gradient descent