3. 파이썬과 벡터화

벡터화

What is vectorization?

- for문을 없애 딥러닝의 학습 시간 단축시킴

ex.

```
import numpy as np

a = np.array([1,2,3,4])

# Vectorized
import time
a = np.random.rand(1000000)
b = np.random.rand(1000000)

tic = time.time()
c = np.dot(a,b) # 행렬곱
toc = time.time()

print("Vectorized version:" + str(1000*(toc-tic))+"ms")
```

>> 실행시간: 1.5ms

```
# For loop

c = 0
tic = time.time()
for i in range(100000):
  c += a[i]*b[i]
toc = time.time()
```

```
print("For loop:"+str(1000*(toc-tic))+"ms")
```

>> 실행시간: 500ms

→ **가능하면 for문을 쓰지 않는다!**

More Vectorization Examples

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

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

- 유용한 numpy의 내장 함수들
    - np.exp()
    - np.log()
    - np.max()
    - np.dot() : 행렬곱

- Linear Regression의 벡터화

$J = 0$; $\cancel{dw_1 = 0}$; $\cancel{dw_2 = 0}$, $db = 0$
→ dw = np.zeros((n-x, 1))

For i=1 to m :

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

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

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

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

$\cancel{dw_1 \mathrel{+}= w_1^{(i)} * dz^{(i)}}$

$\cancel{dw_2 \mathrel{+}= w_2^{(i)} * dz^{(i)}}$ → $dw \mathrel{+}= x^{(i)} dz^{(i)}$

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

$J \mathrel{/}= m$

$\cancel{dw_1 \mathrel{/}= m}$; $\cancel{dw_2 \mathrel{/}= m}$; $db \mathrel{/}= m$
→ $dw \mathrel{/}= m$

로지스틱 회귀와 벡터화

활성값 a() 계산 벡터화

```
# Non-vectorized ver.
z[1] = w.T*x[1] + b
z[2] = w.T*x[2] + b
...
a[1] = sigmoid(z[1])
a[2] = sigmoid(z[2])
...
```

```
# vectorized version
Z = np.dot(w.T, x)+b
A = sigmoid(Z)
```

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

$$Z = [z^{(1)} \ z^{(2)} \ \cdots \ z^{(m)}] = w^T X + [b \ b \ \cdots \ b]$$

$$= [w^T x^{(1)} + b \quad w^T x^{(2)} + b \quad \cdots \quad w^T x^{(m)} + b]$$

$$\Rightarrow Z = \text{np.dot}(w.T, X) + \underbrace{b}_{\text{broadcasting}} \quad (1,1) \to (1,m)$$

$$A = \sigma(Z)$$

경사 계산 벡터화

- non-vetorized version

$J = 0,\ dw_1 = 0,\ dw_2 = 0,\ db = 0$

for i = 1 to m:

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

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

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

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

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

$\quad dw_2 \mathrel{+}= x_2^{(i)} dz^{(i)}$

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

$J = J/m,\ dw_1 = dw_1/m,\ dw_2 = dw_2/m$

$db = db/m$

- vectorized version

$$Z = w^T X + b$$
$$= \text{np.dot}(w.T, X) + b$$
$$A = \sigma(Z)$$
$$dZ = A - Y$$
$$dw = \frac{1}{m} X dZ^T$$
$$db = \frac{1}{m} \text{np.sum}(dZ)$$

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

브로드캐스팅

- Basic rules

#1. (m,n) $\begin{matrix}+\\-\\*\\/\end{matrix}$ $\begin{matrix}(1,n) \rightarrow (m,n)\\ (m,1) \rightarrow (m,n)\end{matrix}$

#2. $(m,1)$ $\begin{matrix}+\\-\\*\\/\end{matrix}$ $\mathbb{R} \rightarrow (m,1)$ 차원 추가.

참고)

Broadcasting — NumPy v1.25 Manual

Site Navigation

https://numpy.org/doc/stable/user/basics.broadcasting.html#

- examples

ex.

$$\underset{(4,1)}{\begin{bmatrix}1\\2\\3\\4\end{bmatrix}} + \underset{(1,\ )}{100}$$

$$\rightarrow \underset{(4,1)}{\begin{bmatrix}1\\2\\3\\4\end{bmatrix}} + \underset{(4,1)}{\begin{bmatrix}100\\100\\100\\100\end{bmatrix}} = \underset{(4,1)}{\begin{bmatrix}101\\102\\103\\104\end{bmatrix}}$$

ex2.

$$\underset{(2,3)}{\begin{bmatrix}1 & 2 & 3\\4 & 5 & 6\end{bmatrix}} + \underset{(1,3)}{\begin{bmatrix}100 & 200 & 300\end{bmatrix}}$$

$$\rightarrow \underset{(2,3)}{\begin{bmatrix}1 & 2 & 3\\4 & 5 & 6\end{bmatrix}} + \underset{(2,3)}{\begin{bmatrix}100 & 200 & 300\\100 & 200 & 300\end{bmatrix}} \rightarrow \underset{(2,3)}{\begin{bmatrix}101 & 202 & 303\\104 & 205 & 306\end{bmatrix}}$$

파이썬과 넘파이 벡터

- shape이 (n,)인 벡터 사용하지 않기!
    - (n,) → (n,1)
    - a = a.reshape(n,1)

로지스틱 회귀와 비용함수

<Loss function>

If $y=1$: $P(y|x) = \hat{y}$

$y=0$: $P(y|x) = 1-\hat{y}$

$(\hat{y} = \sigma(w^T x + b), \quad \hat{y} = P(y=1|x))$

$\Rightarrow P(y|x) = \hat{y}^{y} \cdot (1-\hat{y})^{(1-y)}$

pf. If $y=1$: $P(y|x) = \hat{y} \cdot (1-\hat{y})^0 = \hat{y}$

$y=0$: $P(y|x) = \hat{y}^0 (1-\hat{y})^1 = 1-\hat{y}$

$$\log P(y|x) = \log \hat{y}^{y} (1-\hat{y})^{(1-y)} = y \log \hat{y} + (1-y) \log(1-\hat{y})$$
$$= -L(\hat{y}, y)$$

<For entire training set>

$$\log P(\text{labels in training set}) = \log \prod_{i=1}^{m} P(y^{(i)} | x^{(i)})$$

$$\log P(\cdots) = \sum_{i=1}^{m} \log P(y^{(i)} | x^{(i)})$$
$$= -\sum_{i=1}^{m} L(\hat{y}^{(i)}, y^{(i)})$$

$$\text{Cost} : J(w,b) = \frac{1}{m} \sum_{i=1}^{m} L(\hat{y}^{(i)}, y^{(i)})$$