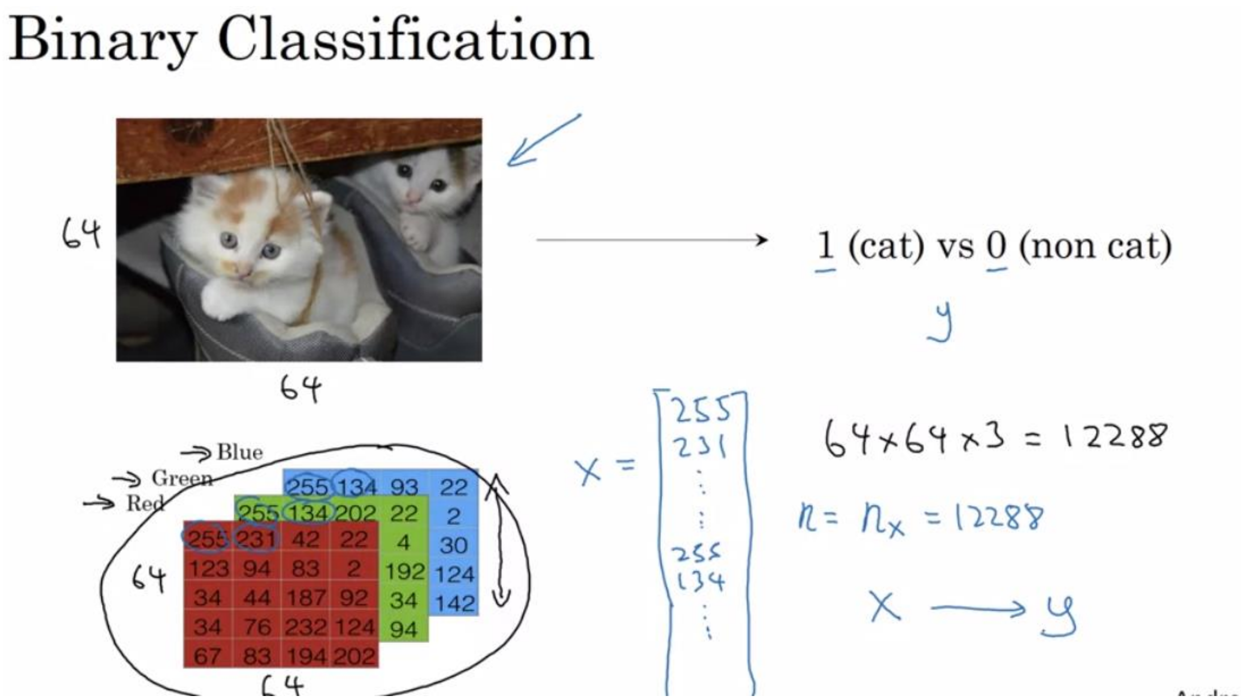
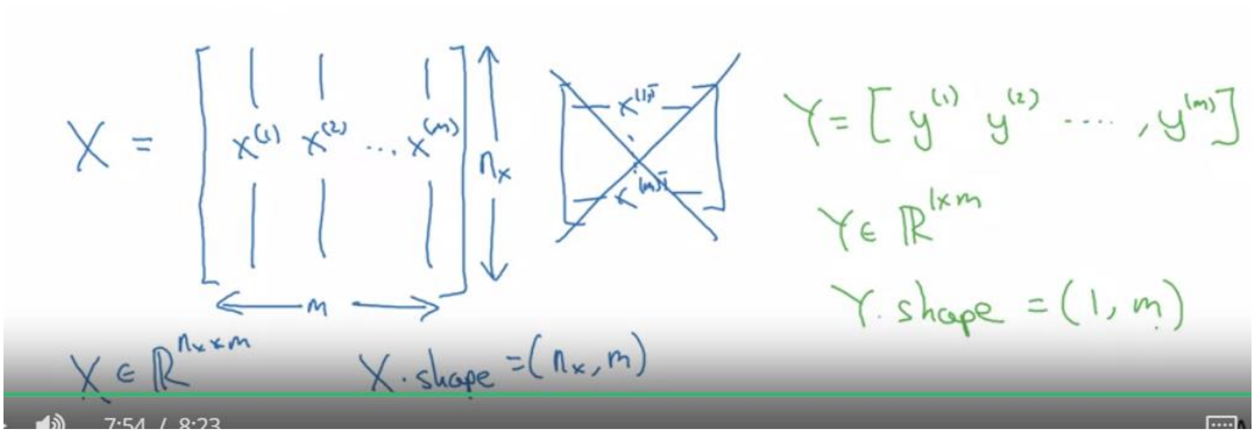
Neural Networks and Deep Learning





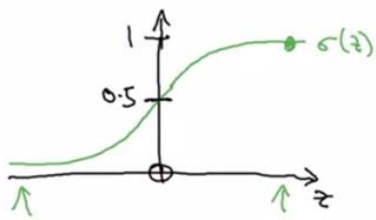
Logistic Regression

Given x, want $\hat{y} = P(y=1 \mid x)$

$x \in \mathbb{R}^{n_x}$ $\quad 0 \leq \hat{y} \leq 1$

Parameters: $w \in \mathbb{R}^{n_x}$, $b \in \mathbb{R}$.

Output $\hat{y} = \sigma(\underbrace{w^T x + b}_{z})$



$\sigma(z) = \frac{1}{1+e^{-z}}$

If z large $\sigma(z) \approx \frac{1}{1+0} = 1$

If z large negative number

$\sigma(z) = \frac{1}{1+e^{-z}} \approx \frac{1}{1+\text{Bignum}} \approx 0$

Andrew

Logistic Regression cost function

$\rightarrow \hat{y}^{(i)} = \sigma(w^T x^{(i)} + b)$, where $\sigma(z^{(i)}) = \frac{1}{1+e^{-z^{(i)}}}$ $\qquad z^{(i)} = w^T x^{(i)} + b$

Given $\{(x^{(1)}, y^{(1)}), \ldots, (x^{(m)}, y^{(m)})\}$, want $\hat{y}^{(i)} \approx y^{(i)}$. $\qquad x^{(i)}, y^{(i)}, z^{(i)}$ $\quad$ i-th example.

Loss (error) function: $\mathcal{L}(\hat{y}, y) = \frac{1}{2}(\hat{y} - y)^2$

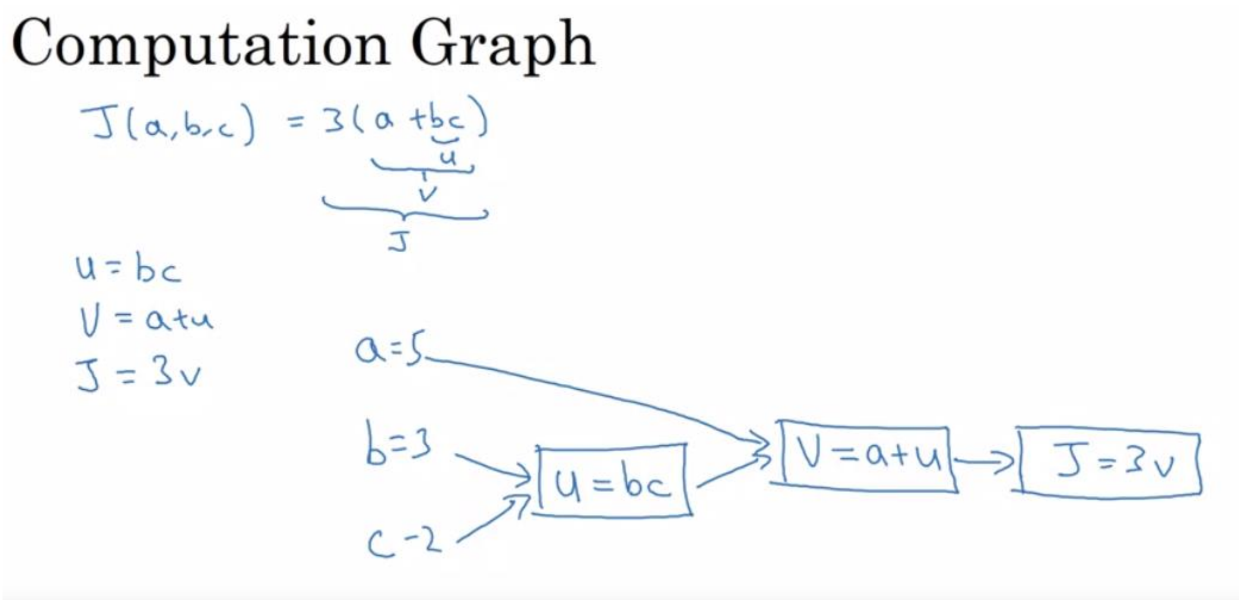
$\mathcal{L}(\hat{y}, y) = -(\boxed{y \log \hat{y}} + (1-y)\log(1-\hat{y})) \leftarrow$

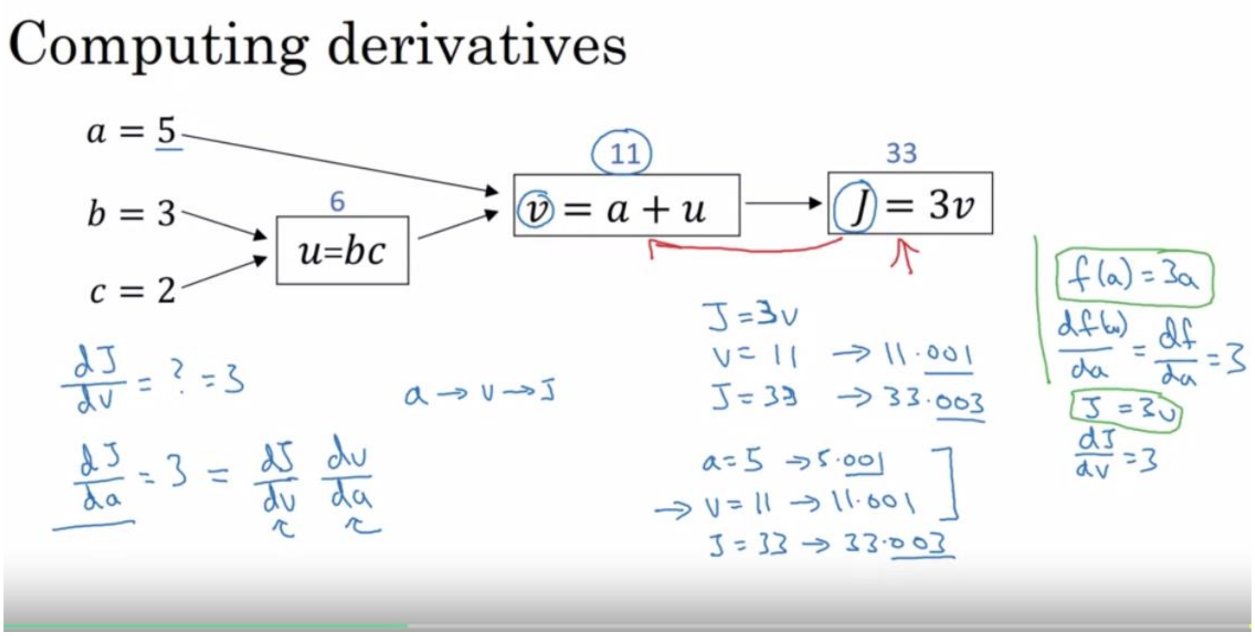
If $y=1$: $\mathcal{L}(\hat{y}, y) = -\log \hat{y} \leftarrow$ Want $\log \hat{y}$ large, want $\hat{y}$ large.

If $y=0$: $\mathcal{L}(\hat{y}, y) = -\log(1-\hat{y}) \leftarrow$ Want $\log 1-\hat{y}$ large .... want $\hat{y}$ small

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

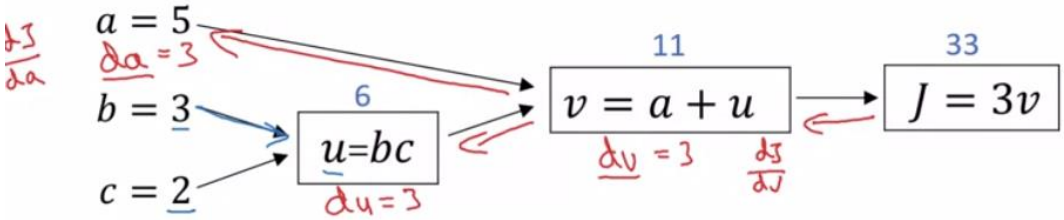
Derivative = Slope 😊





Chain Rule of Calculus!

Computing derivatives



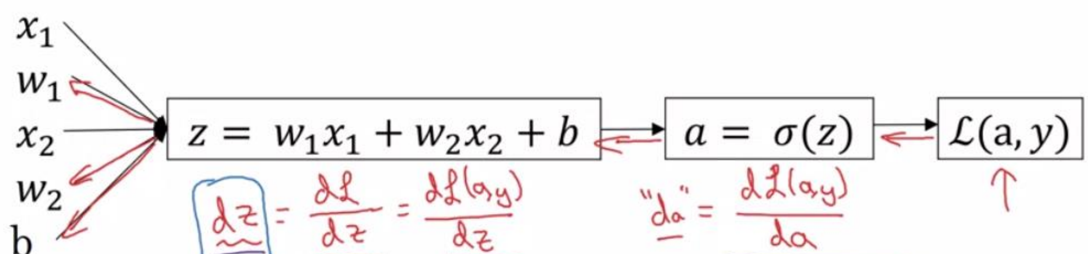
$$\frac{dJ}{du} = 3 = \underbrace{\frac{dJ}{dv}}_{3} \cdot \underbrace{\frac{dv}{du}}_{1}$$

$$\frac{dJ}{db} = \frac{dJ}{du} \cdot \underbrace{\frac{du}{db}}_{=2}$$

$u = 6 \rightarrow 6.001$
$v = 11 \rightarrow 11.001$
$J = 33 \rightarrow 33.003$

$b = 3 \rightarrow 3.001$
$u = b \cdot c = 6 \rightarrow 6.002 \qquad c = 2$

Logistic regression derivatives



$$dz = \frac{d\mathcal{L}}{dz} = \frac{d\mathcal{L}(a,y)}{dz}$$
$$= a - y$$
$$= \frac{d\mathcal{L}}{da} \cdot \frac{da}{dz}$$

$$a(1-a)$$

$$\text{"}da\text{"} = \frac{d\mathcal{L}(a,y)}{da} = \boxed{-\frac{y}{a} + \frac{1-y}{1-a}}$$

$$\frac{\partial \mathcal{L}}{\partial w_1} = \text{"}dw_1\text{"} = x_1 \cdot dz. \qquad dw_2 = x_2 \cdot dz. \qquad db = dz.$$

$w_1 := w_1 - \alpha\, dw_1$
$w_2 := w_2 - \alpha\, dw_2$
b .

Logistic regression on m examples

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

For $i=1$ to m

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

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

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

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

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

$dw_2 \mathrel{+}= x_2^{(i)}dz^{(i)}$ $\quad n=2$

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

$J/=m$

$dw_1/=m;\ dw_2/=m;\ db/=m.$

```
250286.989866
Vectorized version:1.5027523040771484ms
250286.989866
For loop:474.29513931274414ms
```

Logistic regression derivatives

J = 0, ~~dw1 = 0, dw2 = 0~~, db = 0 $\quad$ $dw = np.zeros((n\_x, 1))$

for i = 1 to m:

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

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

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

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

for $j = 1 \ldots n_x$

$\quad dw_j \mathrel{+}= \ldots$

$\quad \cancel{dw_1 \mathrel{+}= x_1^{(i)} dz^{(i)}}$ $\quad n_x = 2$ $\quad dw \mathrel{+}= x^{(i)} dz^{(i)}$

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

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

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

$dw /= m.$

Vectorizing Logistic Regression

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

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

$$X = \begin{bmatrix} | & | & & | \\ x^{(1)} & x^{(2)} & \cdots & x^{(m)} \\ | & | & & | \end{bmatrix} \qquad (n_x, m) \quad \mathbb{R}^{n_x \times m} \qquad [\;\text{———}\;] \; w^T \begin{bmatrix} | & | & & | \\ x^{(1)} & x^{(2)} & \cdots & x^{(m)} \\ | & | & & | \end{bmatrix}$$

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

$\rightarrow Z = np.dot(w.T, X) + b \quad (1,1) \quad \mathbb{R}$ $\quad$ "Broadcasting"

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

Andrew Ng

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

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

(m, n) matrix	$+$ $-$ $*$ $/$	$(1, n)$	$\rightsquigarrow (m, n)$
		$(m, 1)$	$\rightsquigarrow (m, n)$

$$(m, 1) \quad + \quad \mathbb{R}$$
$$\begin{bmatrix} 1 \\ 2 \\ 3 \end{bmatrix} + 100 = \begin{bmatrix} 101 \\ 102 \\ 103 \end{bmatrix}$$
$$\begin{bmatrix} 1 & 2 & 3 \end{bmatrix} + 100 = \begin{bmatrix} 101 & 102 & 103 \end{bmatrix}$$

Matlab/Octave: bsxfun

```
a = np.random.randn(5,1)
print(a)
```

```
[[-0.0967311 ]
 [-2.38617377]
 [-0.3243588 ]
 [-0.96216349]
 [ 0.54410384]]
```

```
print(a.T)
```

```
[[-0.0967311  -2.38617377 -0.3243588  -0.96216349  0.54410384]]
```

Python/numpy vectors

```
a = np.random.randn(5)
```

a.shape = (5,)
"rank 1 array"

Don't use

```
a = np.random.randn(5,1)
```

→ a.shape = (5,1) column vector ✓

```
a = np.random.randn(1,5)
```

→ a.shape = (1,5) row vector. ✓

```
assert(a.shape == (5,1))
```

Logistic regression cost function

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

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

$\Big\}\ p(y|x)$

$$\boxed{p(y|x) = \hat{y}^{y}\,(1-\hat{y})^{(1-y)}}$$

If $y=1$: $p(y|x) = \hat{y}\ \underbrace{(1-\hat{y})^{0}}_{=1}$

If $y=0$: $p(y|x) = \underbrace{\hat{y}^{0}}_{=1}\ (1-\hat{y})^{(1-y)} = 1 \times (1-\hat{y}) = 1-\hat{y}$

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

$$= -\mathcal{L}(\hat{y}, y)$$

4:23 / 7:14

Andrew Ng

Cost on m examples

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

$$\log p(\ldots) = \sum_{i=1}^{m} \underbrace{\log p(y^{(i)}|x^{(i)})}_{-\mathcal{L}(\hat{y}^{(i)},\, y^{(i)})}$$

$$= -\sum_{i=1}^{m} \mathcal{L}(\hat{y}^{(i)}, y^{(i)})$$

Maximum likelihood estimation

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

(minimize)