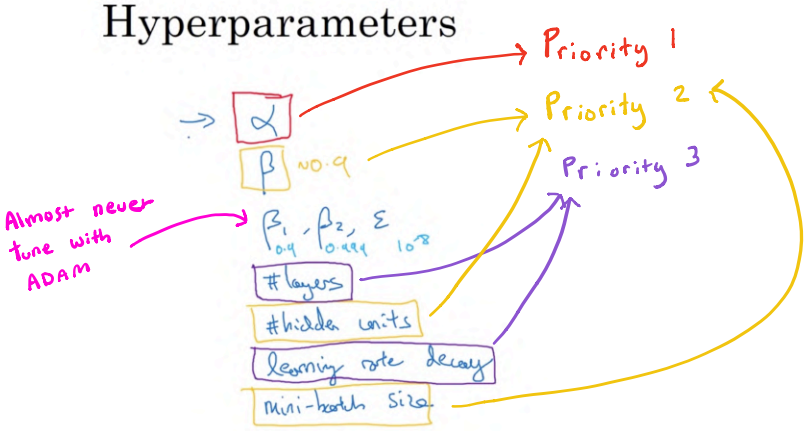
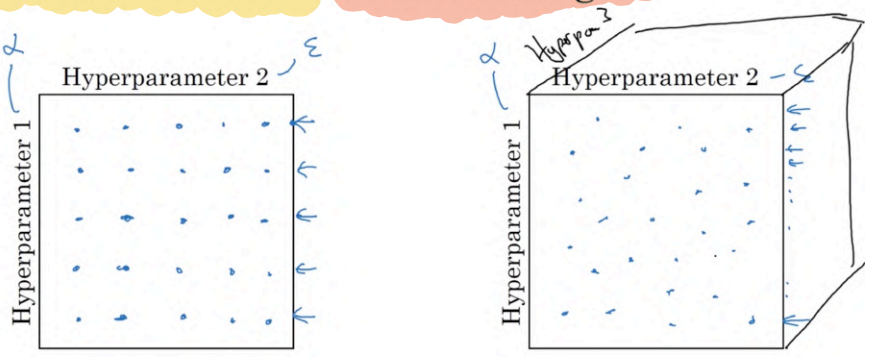
Hyperparameters



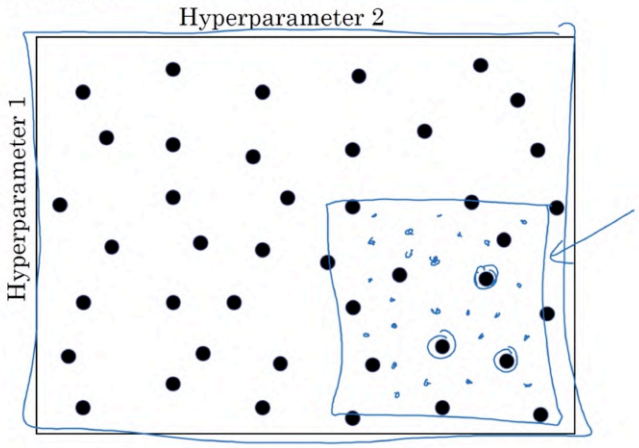
Andrew Ng

Try random values: Don't use a grid



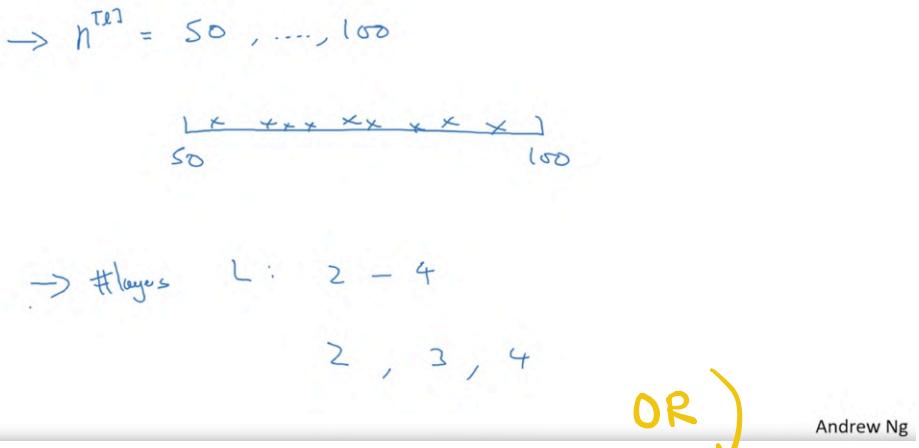
Andrew Ng

Coarse to fine



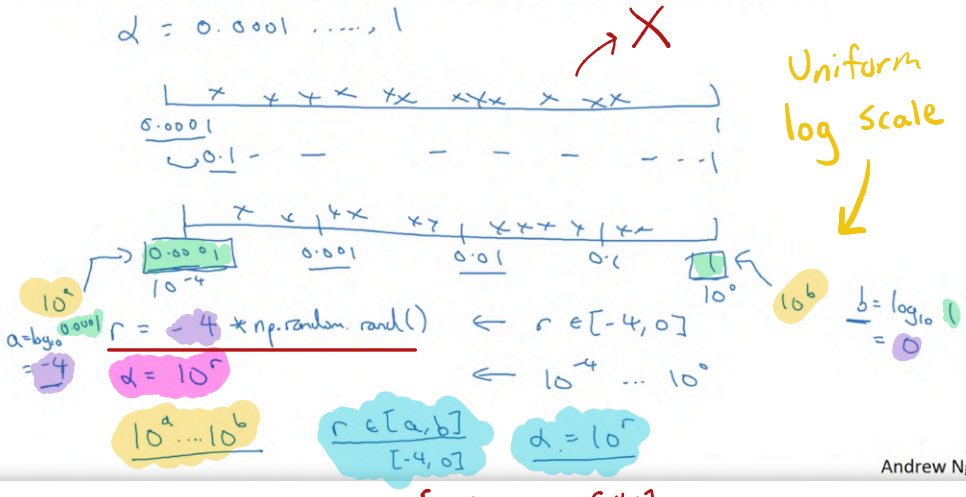
Andrew Ng

Picking hyperparameters at random



Andrew Ng

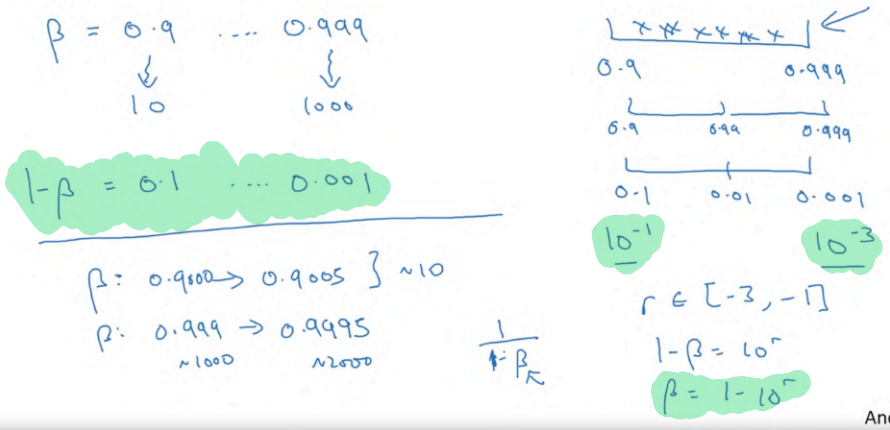
Appropriate scale for hyperparameters



Andrew Ng

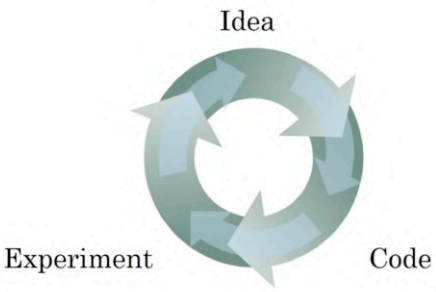
So $\alpha = 10^{r}$ where $r \in [-4, 0]$

Hyperparameters for exponentially weighted averages



Andrew Ng

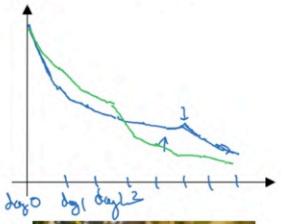
Re-test hyperparameters occasionally

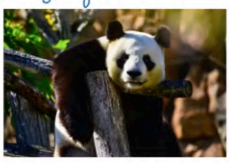


- NLP, Vision, Speech, Ads, logistics, ...
- Intuitions do get stale. Re-evaluate occasionally.

Andrew Ng

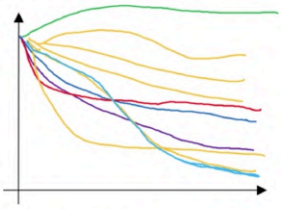
Babysitting one model





Panda

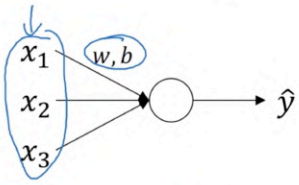
Training many models in parallel



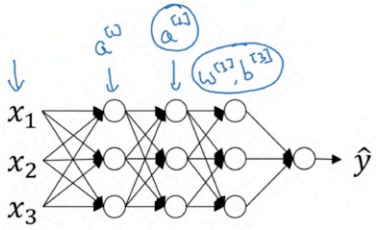
Caviar

Andrew Ng

Normalizing inputs to speed up learning



$$\mu = \frac{1}{m}\sum_i x^{(i)}$$
$$X = X - \mu \quad \text{(element-wise)}$$
$$\sigma^2 = \frac{1}{m}\sum_i x^{(i)2}$$
$$X = X / \sigma^2$$

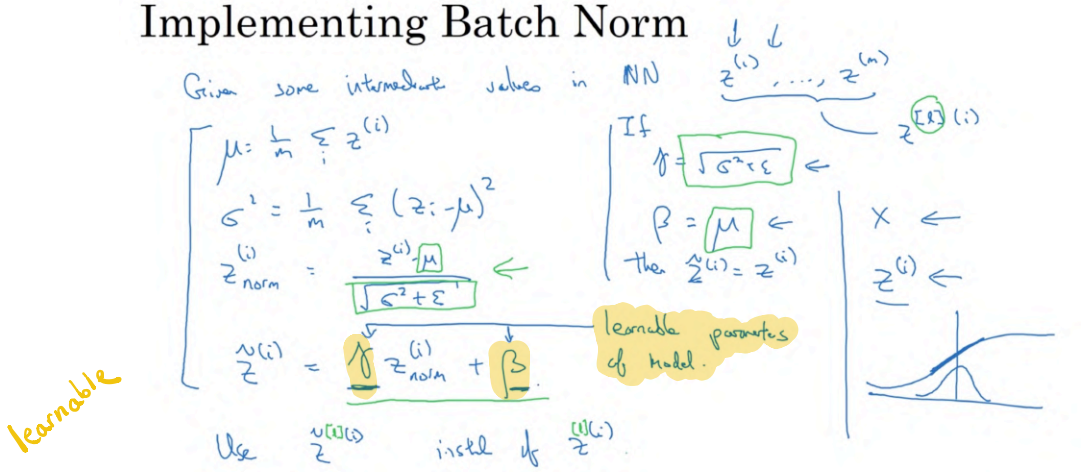


Can we normalize $a^{[2]}$ so as to train $w^{[3]}, b^{[3]}$ faster?

Normalize $z^{[2]}$

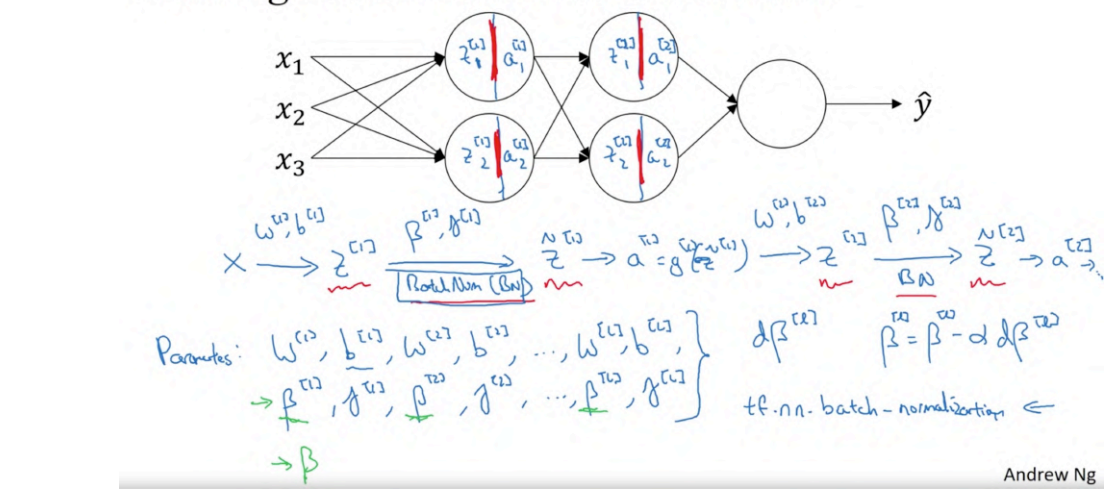
Andrew Ng

Implementing Batch Norm



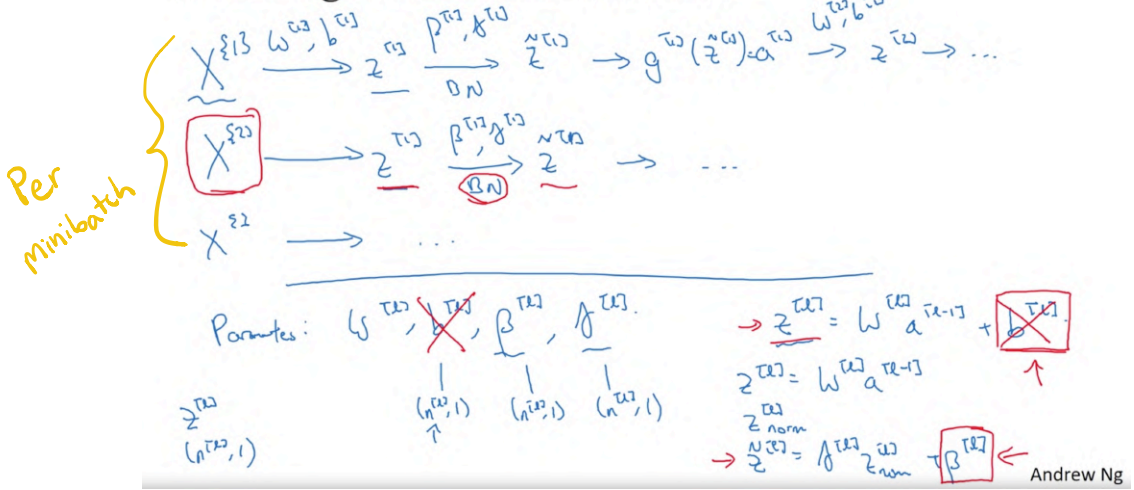
Andrew Ng

Adding Batch Norm to a network



Andrew Ng

Working with mini-batches



Andrew Ng

Implementing gradient descent

for $t = 1 \ldots$ numMiniBatches

Compute forward prop on $X^{\{t\}}$.

In each hidden layer, use BN to replace $z^{[l]}$ with $\tilde{z}^{[l]}$.

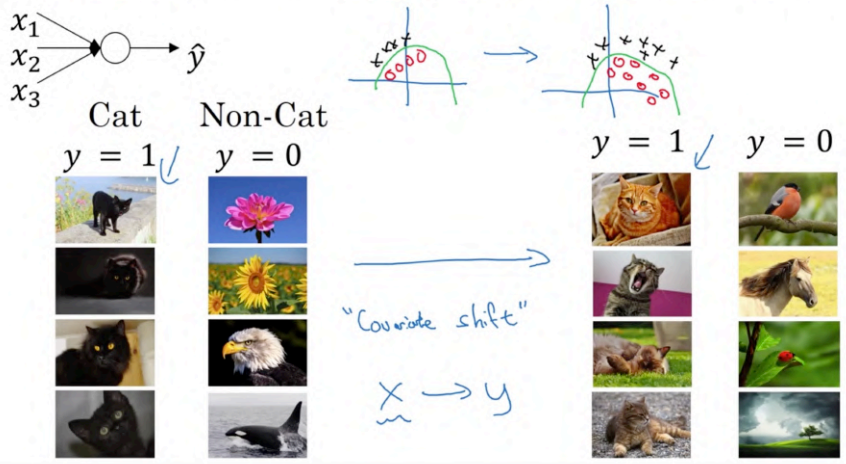
Use backprop to compute $dw^{[l]}$, ~~$db^{[l]}$~~, $d\beta^{[l]}$, $d\gamma^{[l]}$

Update parameters $w^{[l]} := w^{[l]} - \alpha\, dw^{[l]}$, $\beta^{[l]} := \beta^{[l]} - \alpha\, d\beta^{[l]}$, $\gamma^{[l]} := \ldots$

Works w/ momentum, RMSprop, Adam.

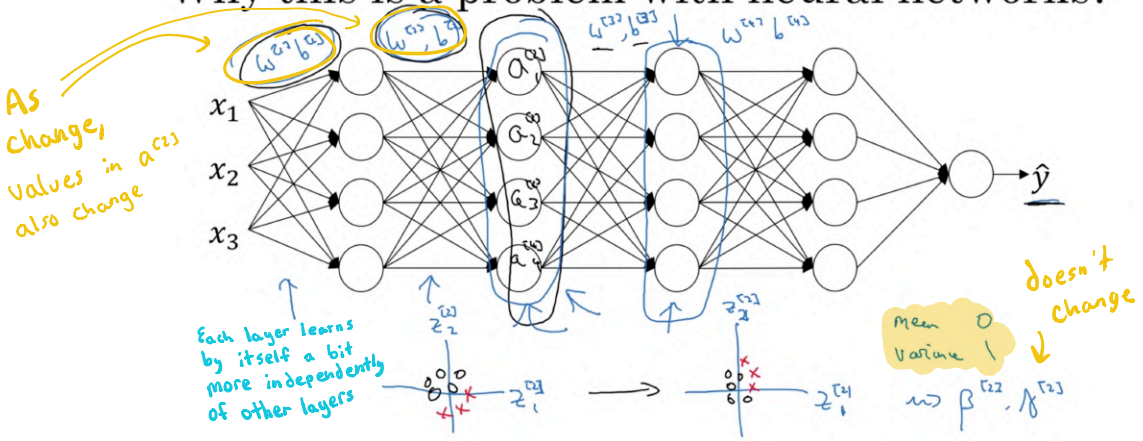
Andrew Ng

Learning on shifting input distribution



Andrew Ng

Why this is a problem with neural networks?



Andrew Ng

Batch Norm as regularization

- Each mini-batch is scaled by the mean/variance computed on just that mini-batch.
- This adds some noise to the values $z^{[l]}$ within that minibatch. So similar to dropout, it adds some noise to each hidden layer's activations.
- This has a slight regularization effect.

batch norm with dropout for greater regularization effect of batch norm

I wouldn't really use batch norm as a regularizer, that's really not the intent of batch norm, but sometimes it has this extra intended or unintended effect on your learning algorithm. But, really, don't turn to batch norm as a regularization. Use it as a way to normalize your hidden units activations and therefore speed up learning. And I think the regularization is an almost unintended side effect

Andrew Ng

Batch Norm at test time

$$\mu = \frac{1}{m}\sum_i z^{(i)}$$

$$\sigma^2 = \frac{1}{m}\sum_i (z^{(i)} - \mu)^2$$

$$z^{(i)}_{\text{norm}} = \frac{z^{(i)} - \mu}{\sqrt{\sigma^2 + \varepsilon}}$$

$$\tilde{z}^{(i)} = \gamma z^{(i)}_{\text{norm}} + \beta$$

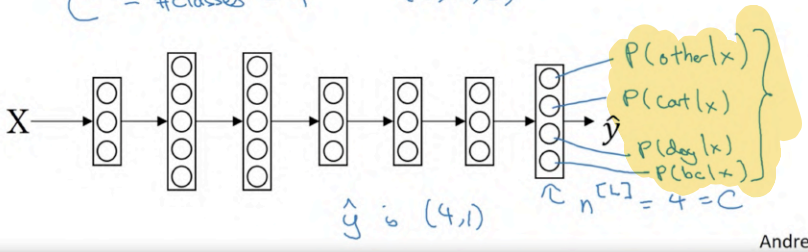
μ, σ^2: estimate using exponentially weighted average (across mini-batches).

Andrew Ng

Recognizing cats, dogs, and baby chicks, other

3 1 2 0 3 2 0 1

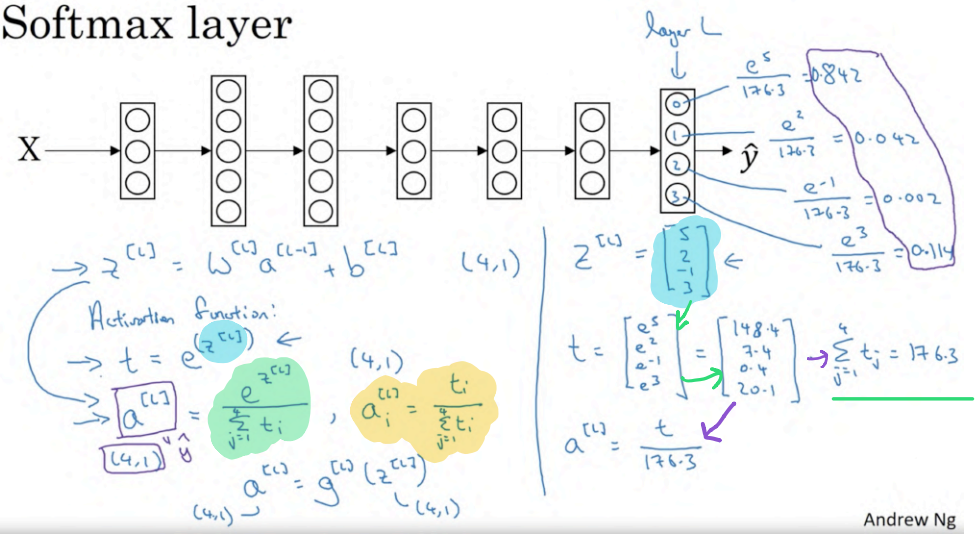
C = #classes = 4 $(0, \ldots, 3)$



$\hat{y}$ is (4,1)

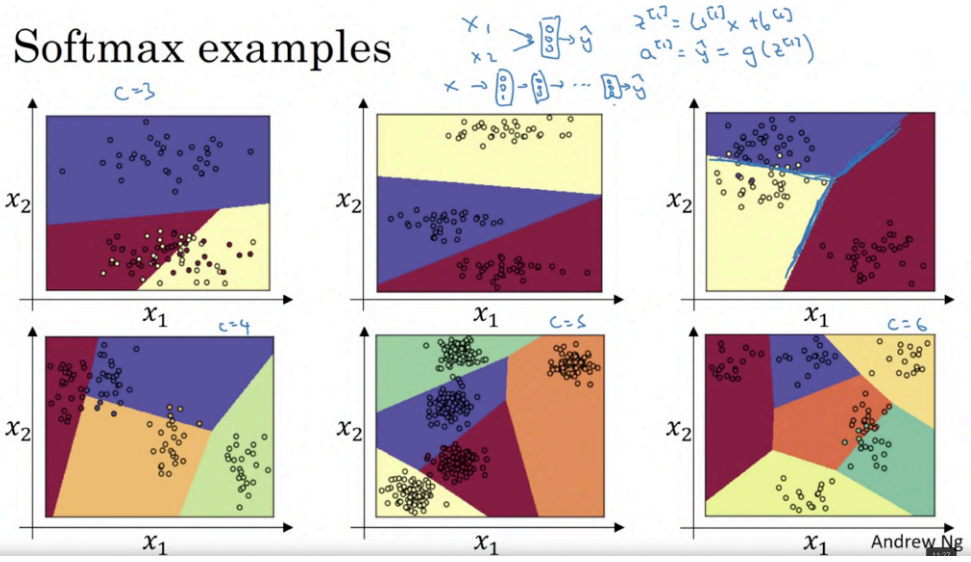
Andrew Ng

Softmax layer



Andrew Ng

Softmax examples



Andrew Ng

Understanding softmax

$$z^{[L]} = \begin{bmatrix} 5 \\ 2 \\ -1 \\ 3 \end{bmatrix} \quad t = \begin{bmatrix} e^5 \\ e^2 \\ e^{-1} \\ e^3 \end{bmatrix}$$

(4,1)

C = 4 $\quad g^{[L]}(\cdot)$

"soft max"

$$a^{[L]} = g^{[L]}(z^{[L]}) = \begin{bmatrix} e^5/(e^5+e^2+e^{-1}+e^3) \\ e^2/(e^5+e^2+e^{-1}+e^3) \\ e^{-1}/(e^5+e^2+e^{-1}+e^3) \\ e^3/(e^5+e^2+e^{-1}+e^3) \end{bmatrix} = \begin{bmatrix} 0.842 \\ 0.042 \\ 0.002 \\ 0.114 \end{bmatrix}$$

"hard max"

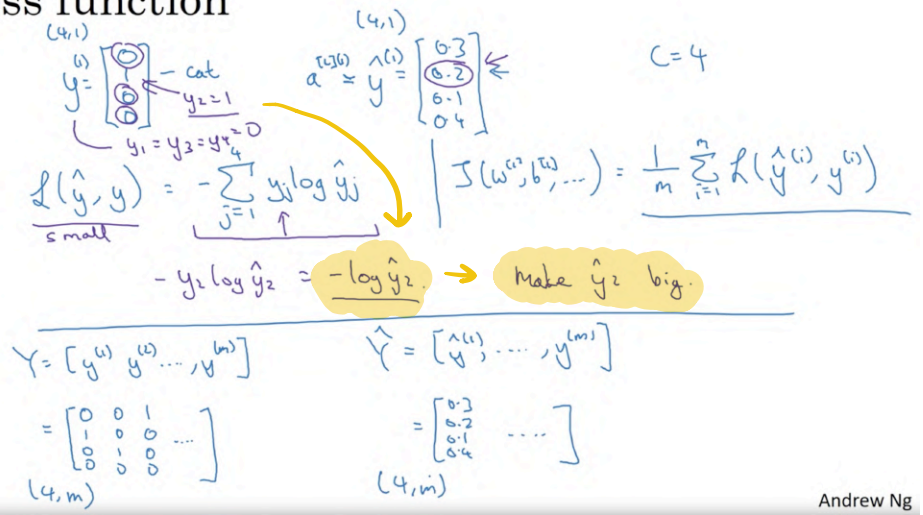
$$\begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

Softmax regression generalizes logistic regression to C classes.

If C=2, softmax reduces to logistic regression. $a^{[L]} = \begin{bmatrix} 0.842 \\ 0.158 \end{bmatrix}$

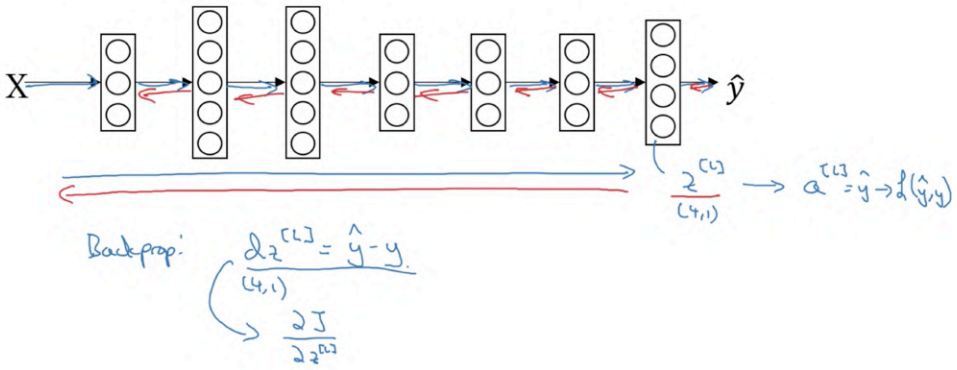
Andrew Ng

Loss function



Andrew Ng

Gradient descent with softmax



Andrew Ng