Hyperparameter tuning

Tuning process

deeplearning.ai

Hyperparameters

$\rightarrow$ $\boxed{\alpha}$

$\boxed{\beta}$ ≈ 0.9

$\beta_1, \beta_2, \varepsilon$
$_{0.9}$ $_{0.999}$ $_{10^{-8}}$

$\boxed{\# \text{layers}}$

$\boxed{\# \text{hidden units}}$

$\boxed{\text{learning rate decay}}$

$\boxed{\text{mini-batch size}}$

Try random values: Don't use a grid

Hyperparameter 2

Hyperparameter 1

Hyperparameter 2

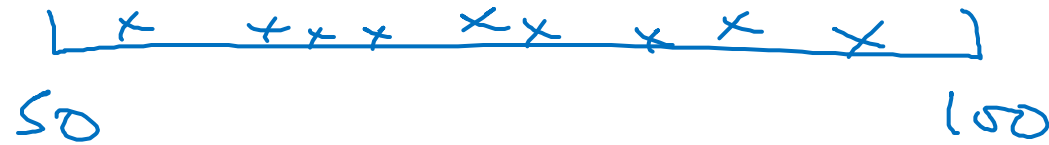Hyperparameter 1

Andrew Ng

Coarse to fine

Hyperparameter 2
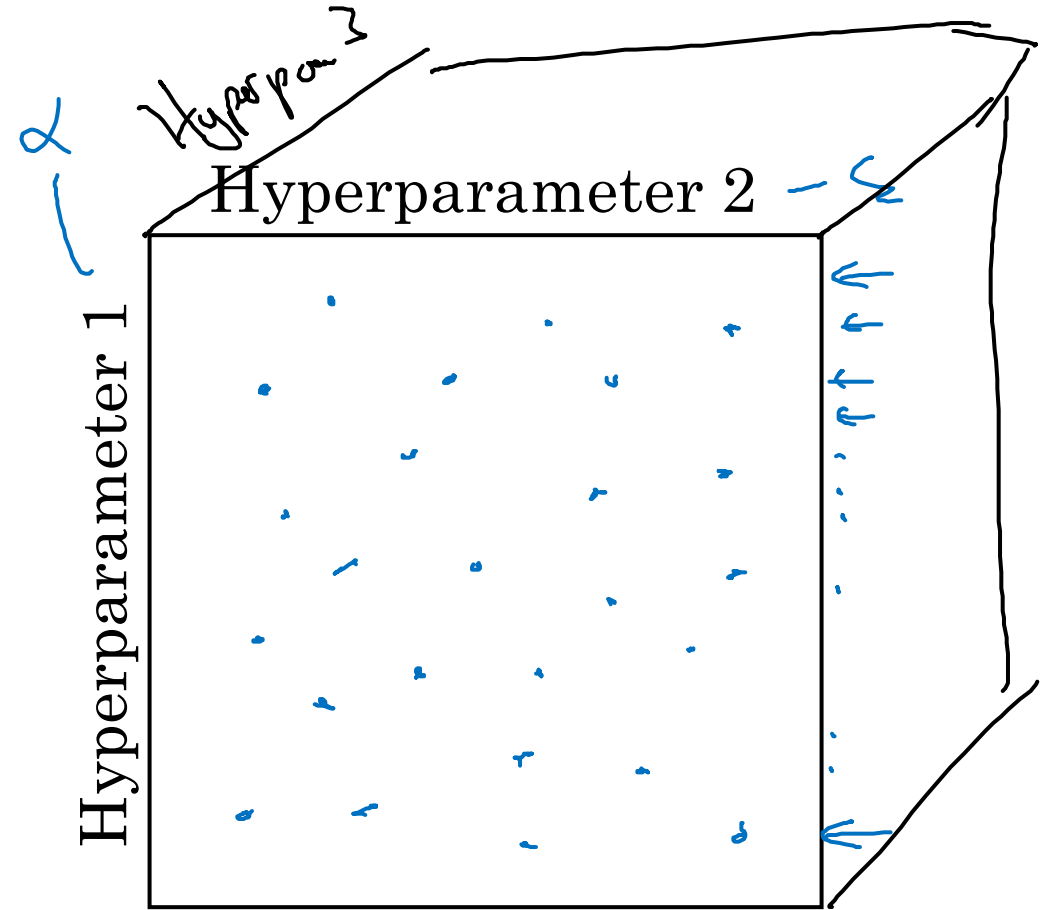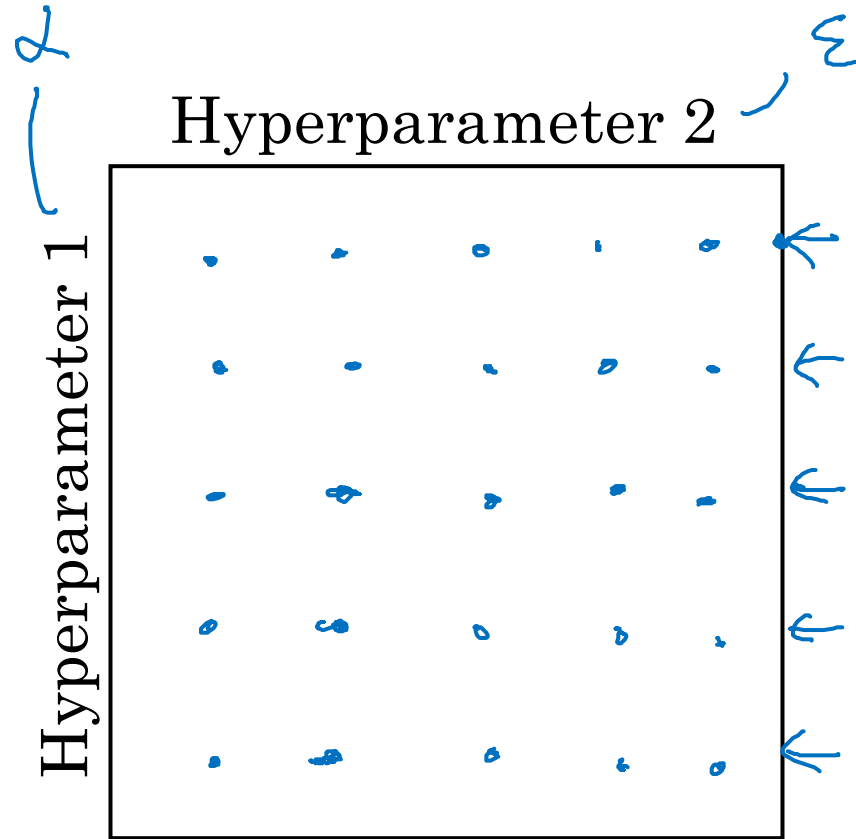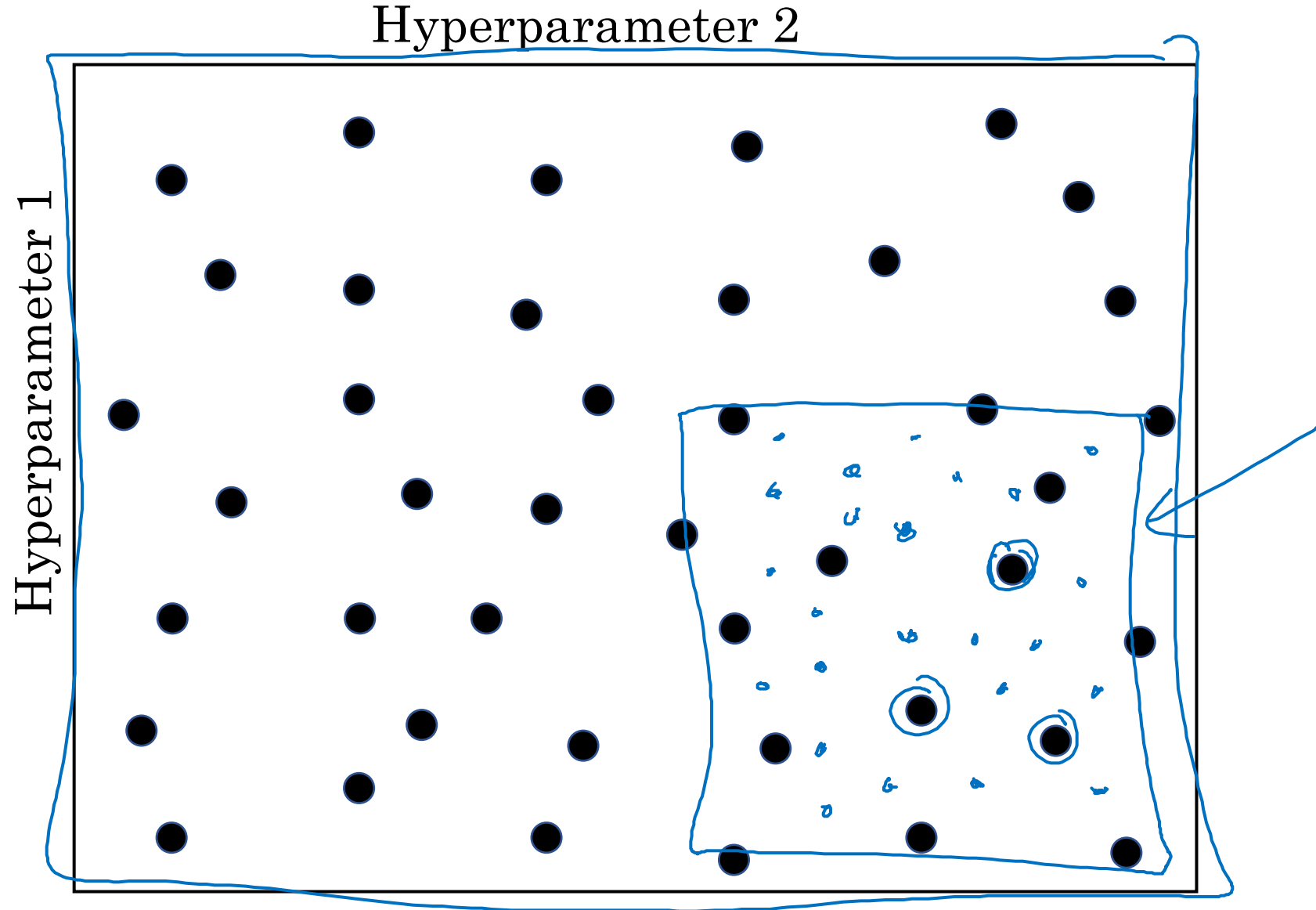
Hyperparameter 1

deeplearning.ai

Hyperparameter tuning

Using an appropriate scale to pick hyperparameters

Picking hyperparameters at random
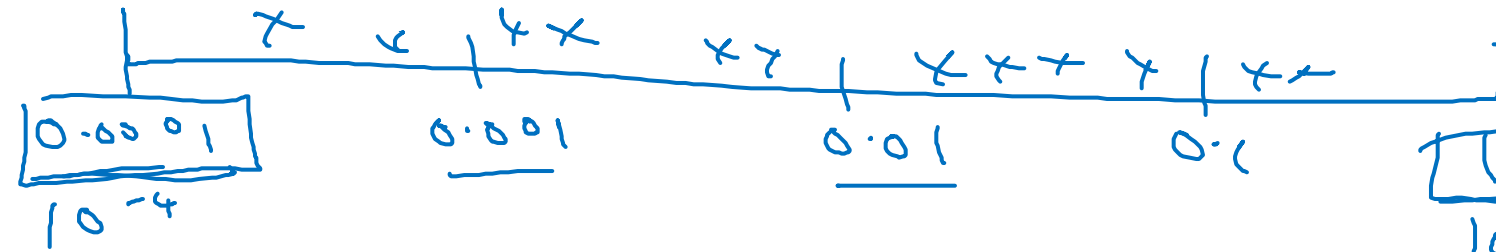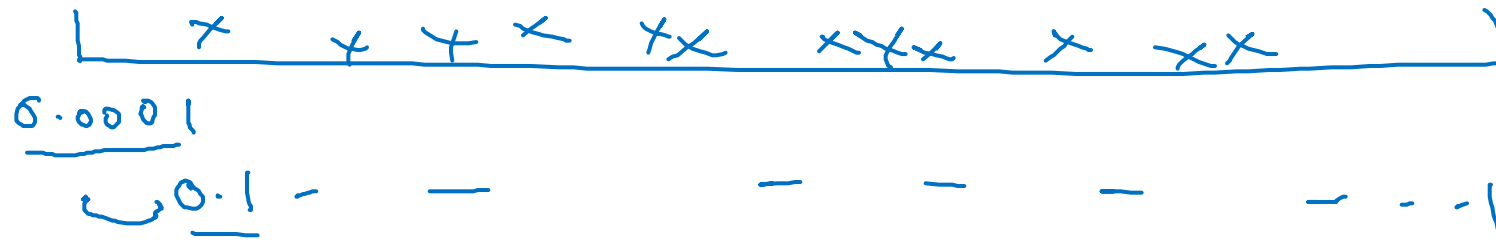
$\rightarrow n^{[\ell]} = 50, \ldots, 100$

$50 \qquad\qquad\qquad 100$

$\rightarrow$ #layers $L: \quad 2 - 4$

$2, 3, 4$

Appropriate scale for hyperparameters

$\alpha = 0.0001 \ , \ ..... \ , \ 1$

0.0001

$0.1 \ - \ \ - \ \ - \ \ - \ \ - \ \ - \ - \ -1$

0.0001 0.001 0.01 0.1

10^{-4} $10^0 \quad 10^b$

10^a

$a = \log_{10} 0.0001 \quad r = -4 * np.random.rand() \quad \leftarrow \quad r \in [-4, 0]$

$b = \log_{10} 1$

$= -4 \qquad \qquad \alpha = 10^r \qquad \qquad \leftarrow \quad 10^{-4} \ ... \ 10^0$

$= 0$

$10^a .... 10^b \qquad \dfrac{r \in [a, b]}{[-4, 0]} \qquad \alpha = 10^r$

Hyperparameters for exponentially weighted averages

$\beta = 0.9 \quad \cdots \quad 0.999$

$\downarrow \qquad\qquad \downarrow$

$10 \qquad\qquad 1000$

$1-\beta = 0.1 \quad \cdots \quad 0.001$

$\beta: 0.9000 \rightarrow 0.9005 \;\}\; \sim 10$

$\beta: 0.999 \rightarrow 0.9995$

$\sim 1000 \qquad\qquad \sim 2000$

$\dfrac{1}{1-\beta}$

$\underline{\; \times \;\times\; \times\times\; \times\;\times\; \times \;} \longleftarrow$

$0.9 \qquad\qquad\qquad 0.999$

$0.9 \qquad 0.99 \qquad 0.999$

$0.1 \qquad 0.01 \qquad 0.001$

$10^{-1} \qquad\qquad\qquad 10^{-3}$
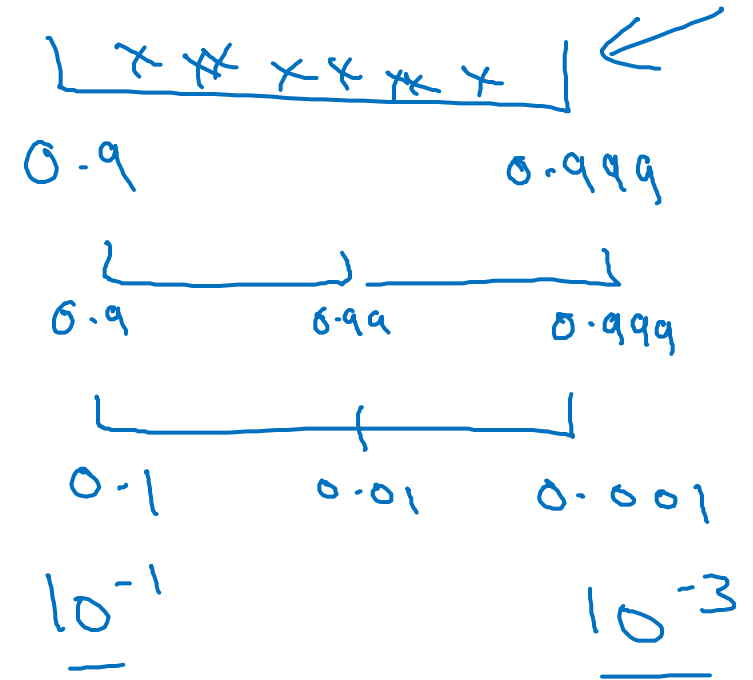
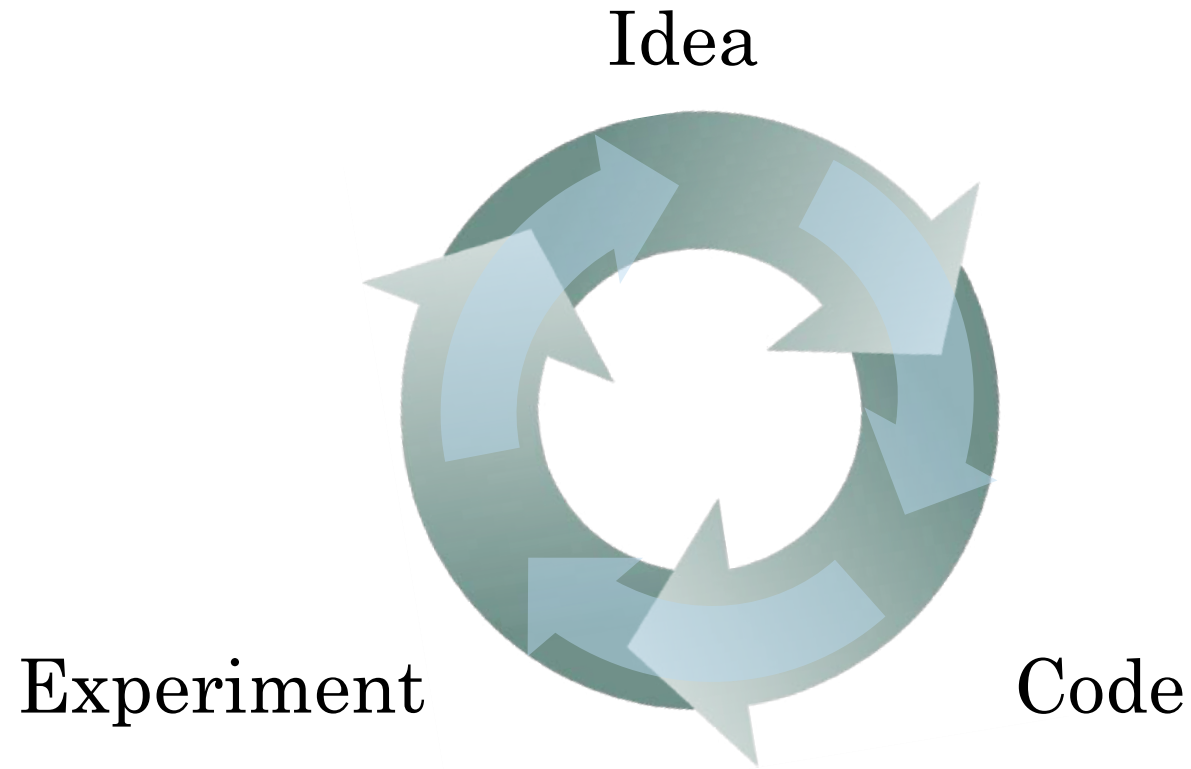$r \in [-3, -1]$

$1-\beta = 10^{r}$

$\beta = 1 - 10^{r}$

deeplearning.ai

Hyperparameters tuning

Hyperparameters tuning in practice: Pandas vs. Caviar
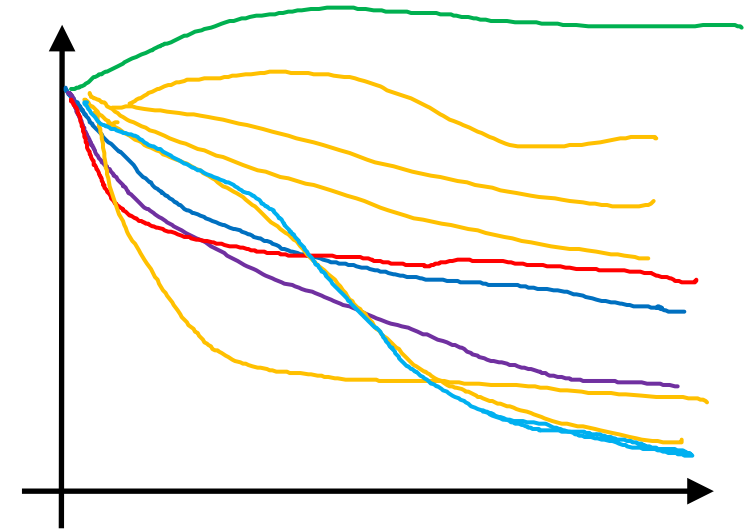
Re-test hyperparameters occasionally

Idea

Code

Experiment

- <u>NLP</u>, <u>Vision</u>, <u>Speech</u>, Ads, <u>logistics</u>, ...

- Intuitions do get stale. Re-evaluate occasionally.

Andrew Ng

Babysitting one model

day 0   day 1   day 2 3
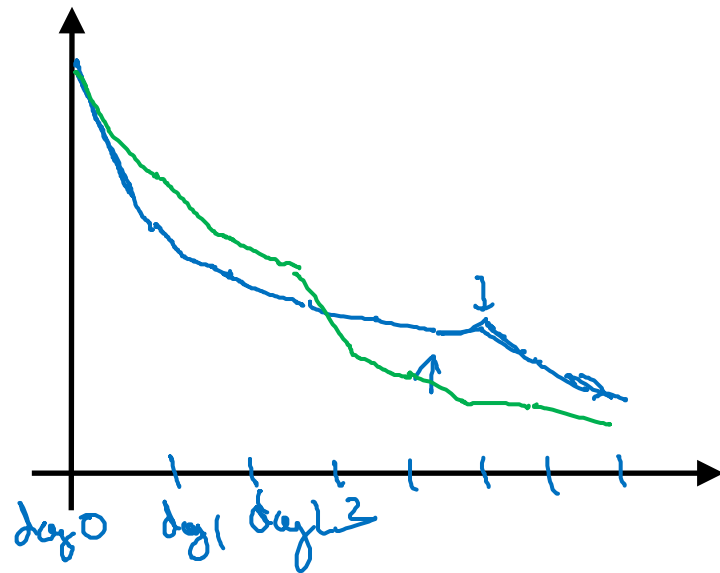
Panda ←

Training many models in parallel

Caviar ←

Andrew Ng

Batch Normalization

Normalizing activations in a network

Normalizing inputs to speed up learning

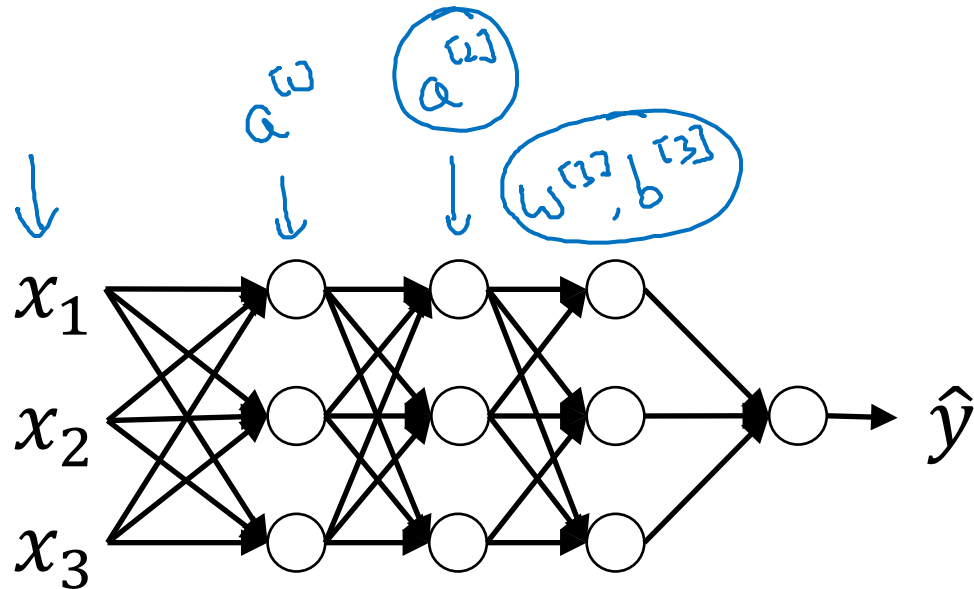x_1

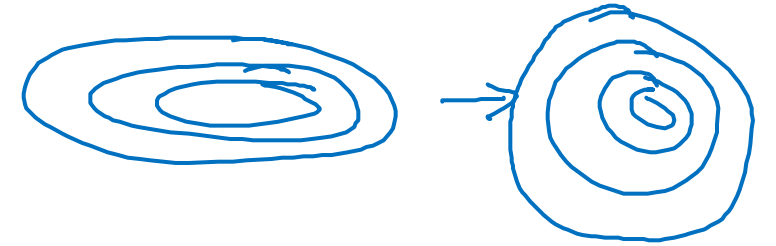w, b

x_2

x_3

$\hat{y}$

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

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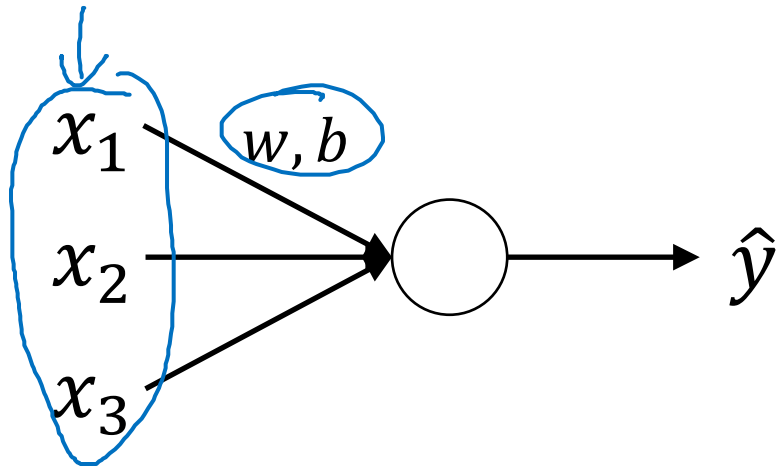
$$\sigma^2 = \frac{1}{m} \sum_i x^{(i)2} \quad \leftarrow \text{element-wise}$$

$$X = X / \sigma$$

$a^{[1]}$

$a^{[2]}$

$w^{[3]}, b^{[3]}$

x_1

x_2

x_3

$\hat{y}$

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

Normalize $z^{[2]}$

Implementing Batch Norm

Given some intermediate values in NN

$z^{(i)}, \ldots, z^{(m)}$

$z^{[l](i)}$

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

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

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

$$\tilde{z}^{(i)} = \gamma \, z_{norm}^{(i)} + \beta$$

learnable parameters of model.

If

$$\gamma = \sqrt{\sigma^2 + \varepsilon}$$

$$\beta = \mu$$

then $\tilde{z}^{(i)} = z^{(i)}$

X

$z^{(i)}$

Use $\tilde{z}^{[l](i)}$ instead of $z^{[l](i)}$.
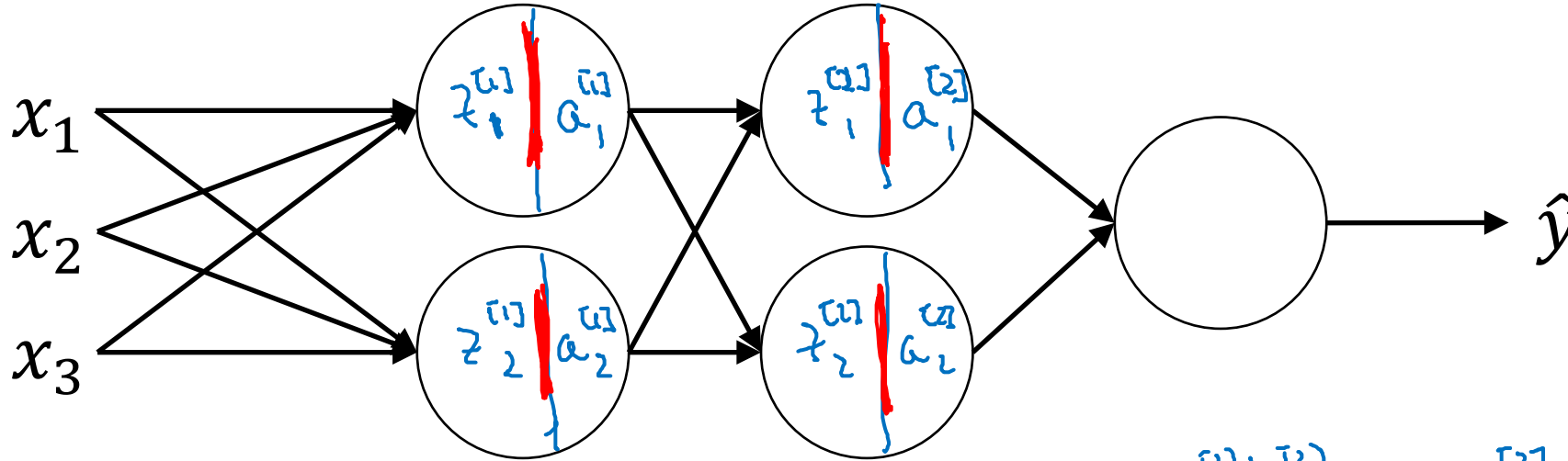
Andrew Ng
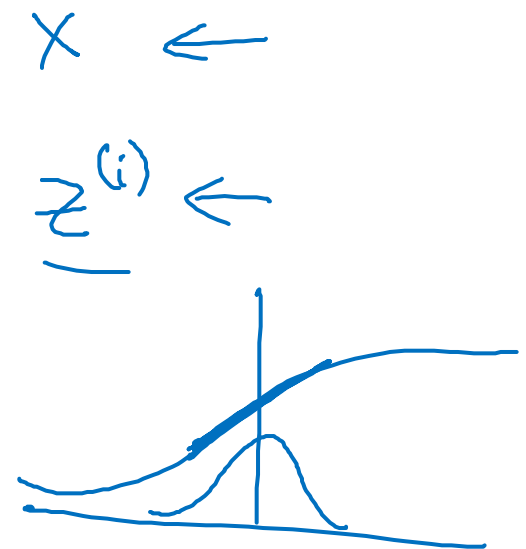
deeplearning.ai

Batch Normalization

Fitting Batch Norm into a neural network

Adding Batch Norm to a network

$$X \xrightarrow{\;W^{[1]}, b^{[1]}\;} Z^{[1]} \xrightarrow[\boxed{\text{Batch Norm (BN)}}]{\;\beta^{[1]}, \gamma^{[1]}\;} \tilde{Z}^{[1]} \rightarrow a^{[1]} = g^{[1]}(\tilde{Z}^{[1]}) \xrightarrow{\;W^{[2]}, b^{[2]}\;} Z^{[2]} \xrightarrow[\underline{BN}]{\;\beta^{[2]}, \gamma^{[2]}\;} \tilde{Z}^{[2]} \rightarrow a^{[2]} \rightarrow \cdots$$

Parameters: $W^{[1]}, b^{[1]}, W^{[2]}, b^{[2]}, \ldots, W^{[L]}, b^{[L]},$
$\rightarrow \beta^{[1]}, \gamma^{[1]}, \beta^{[2]}, \gamma^{[2]}, \ldots, \beta^{[L]}, \gamma^{[L]}$

$d\beta^{[l]}$

$\beta^{[l]} = \beta^{[l]} - \alpha \, d\beta^{[l]}$

tf.nn.batch-normalization

$\rightarrow \beta$

Andrew Ng

Working with mini-batches

$$X^{\{1\}} \xrightarrow{W^{[1]}, b^{[1]}} Z^{[1]} \xrightarrow[BN]{\beta^{[1]}, \gamma^{[1]}} \tilde{Z}^{[1]} \longrightarrow g^{[1]}(\tilde{Z}^{[1]}) = a^{[1]} \xrightarrow{W^{[2]}, b^{[2]}} Z^{[2]} \longrightarrow \dots$$

$$\boxed{X^{\{2\}}} \longrightarrow Z^{[1]} \xrightarrow[\underline{BN}]{\beta^{[1]}, \gamma^{[1]}} \tilde{Z}^{[1]} \longrightarrow \dots$$

$$X^{\{3\}} \longrightarrow \dots$$

Parameters: $W^{[\ell]}, \cancel{b^{[\ell]}}, \beta^{[\ell]}, \gamma^{[\ell]}.$

$(n^{[\ell]}, 1) \quad (n^{[\ell]}, 1) \quad (n^{[\ell]}, 1)$

$Z^{[\ell]}$

$(n^{[\ell]}, 1)$

$\longrightarrow Z^{[\ell]} = W^{[\ell]} a^{[\ell-1]} + \boxed{\cancel{b^{[\ell]}}}$

$Z^{[\ell]} = W^{[\ell]} a^{[\ell-1]}$

$Z^{[\ell]}_{norm}$

$\longrightarrow \tilde{Z}^{[\ell]} = \gamma^{[\ell]} Z^{[\ell]}_{norm} + \boxed{\beta^{[\ell]}}$

Andrew Ng

Implementing gradient descent

for $t = 1$ .... num Mini Batches

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

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

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

  Update paramters $\left.\begin{array}{l} W^{[l]} := W^{[l]} - \alpha\, dW^{[l]} \\ \beta^{[l]} := \beta^{[l]} - \alpha\, d\beta^{[l]} \\ \gamma^{[l]} := \cdots \end{array}\right\} \leftarrow$

Works w/ momentum, RMSprop, Adam.

Andrew Ng

Batch Normalization

Why does Batch Norm work?

deeplearning.ai

Learning on shifting input distribution

x_1
x_2
x_3
$\hat{y}$

Cat
$y = 1$
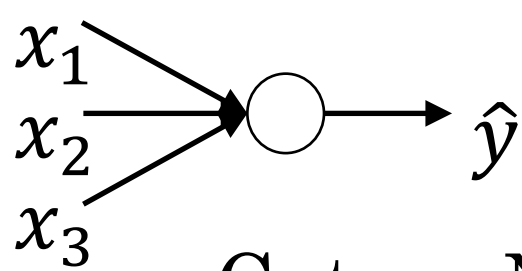
Non-Cat
$y = 0$
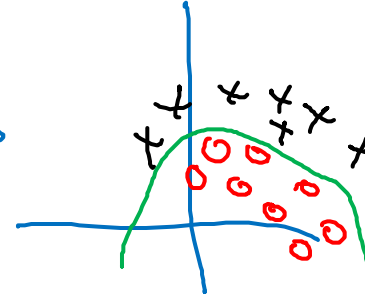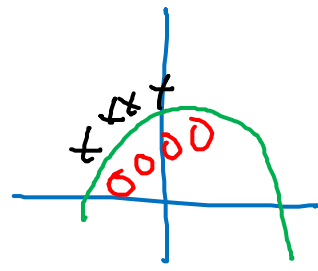
$y = 1$

$y = 0$

"Covariate shift"

$x \longrightarrow y$

Why this is a problem with neural networks?

$\hat{y}$

$W^{[3]}, b^{[3]}$ $W^{[4]}, b^{[4]}$

$a_1^{[2]}$
$a_2^{[2]}$
$a_3^{[2]}$
$a_4^{[2]}$

$z_2^{[2]}$

$z_1^{[2]}$

$z_2^{[2]}$

$z_1^{[2]}$

Mean 0
Variance 1

$\Rightarrow \beta^{[2]}, \gamma^{[2]}$

Andrew Ng
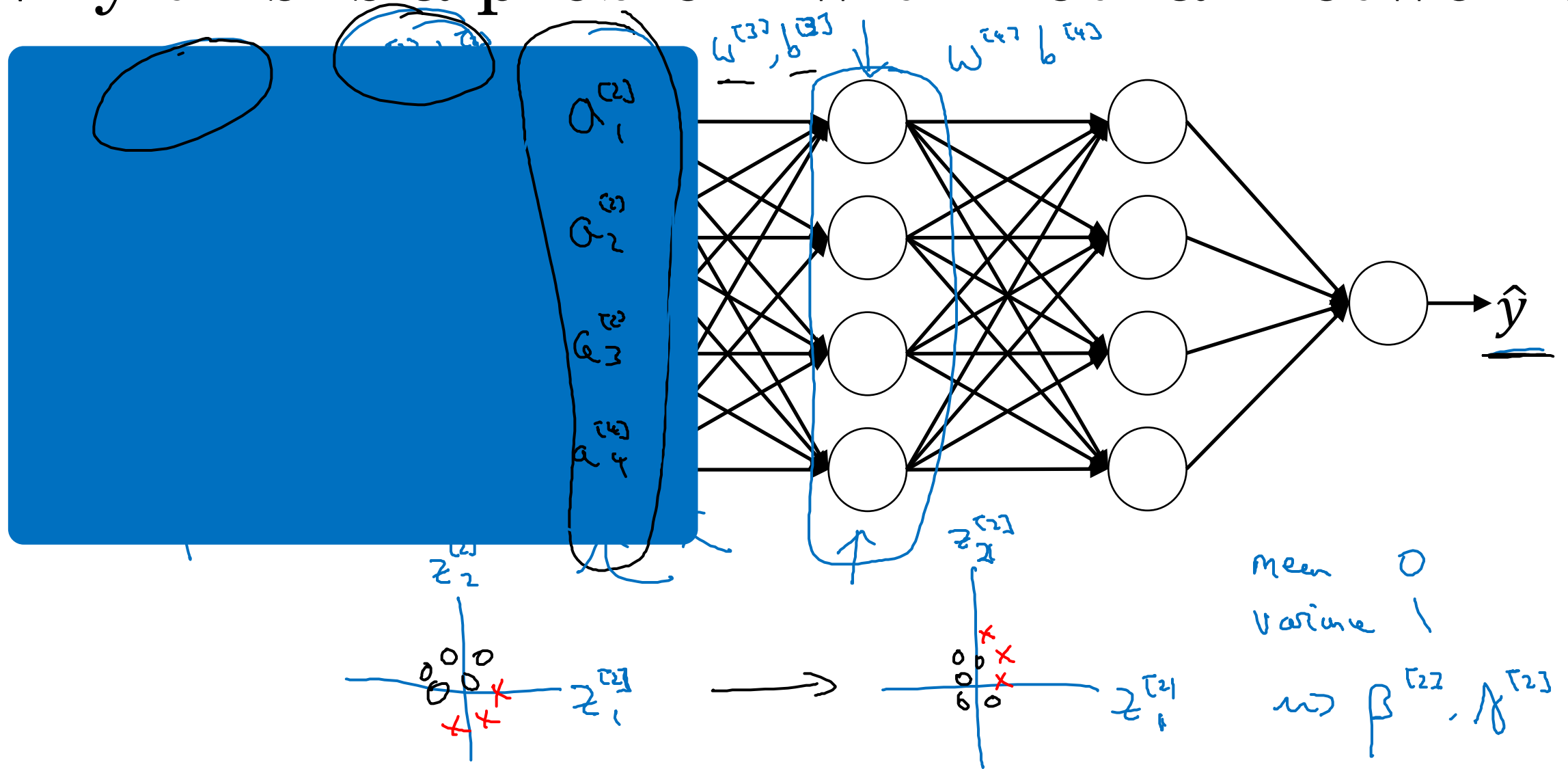
Batch Norm as regularization

- 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.

$\tilde{z}^{[l]}$

$64, 128$

$z^{[l]}$

$X^{\{t\}}$

μ, σ^2

$\text{Mini-batch} : 64 \longrightarrow 512$

Multi-class classification

Softmax regression

deeplearning.ai

Recognizing cats, dogs, and baby chicks
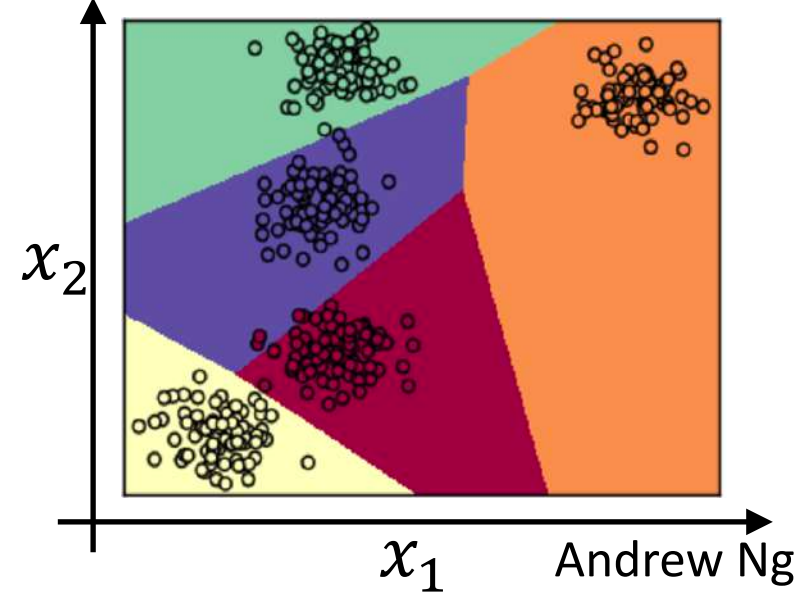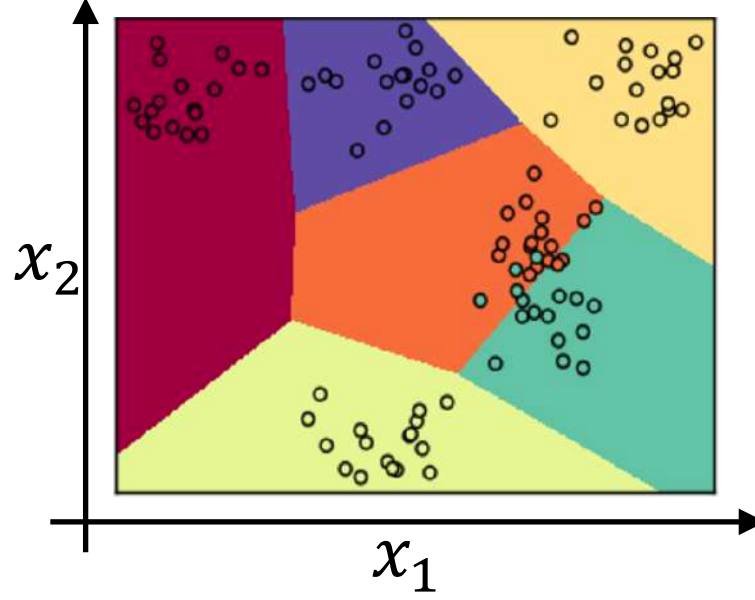

3   1   2   0   3   2   0   1
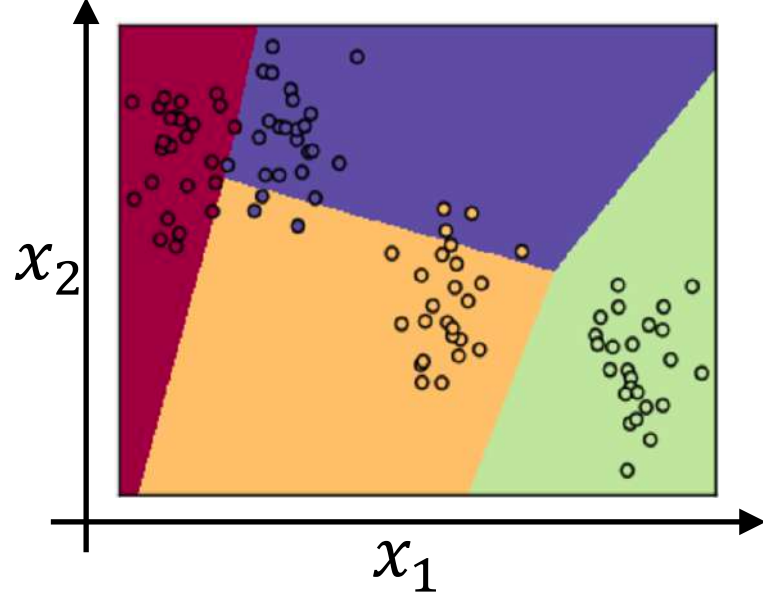
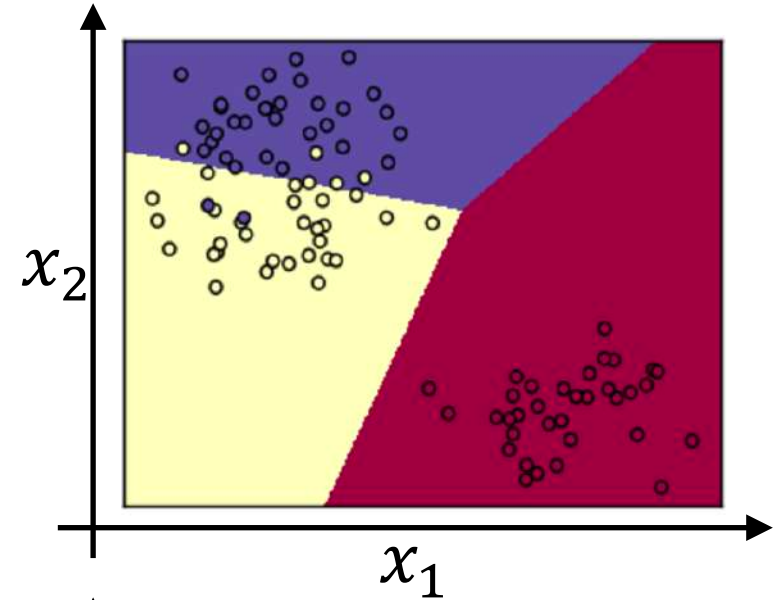$$X \rightarrow \hat{y}$$

Softmax layer

X $\longrightarrow$ $\hat{y}$
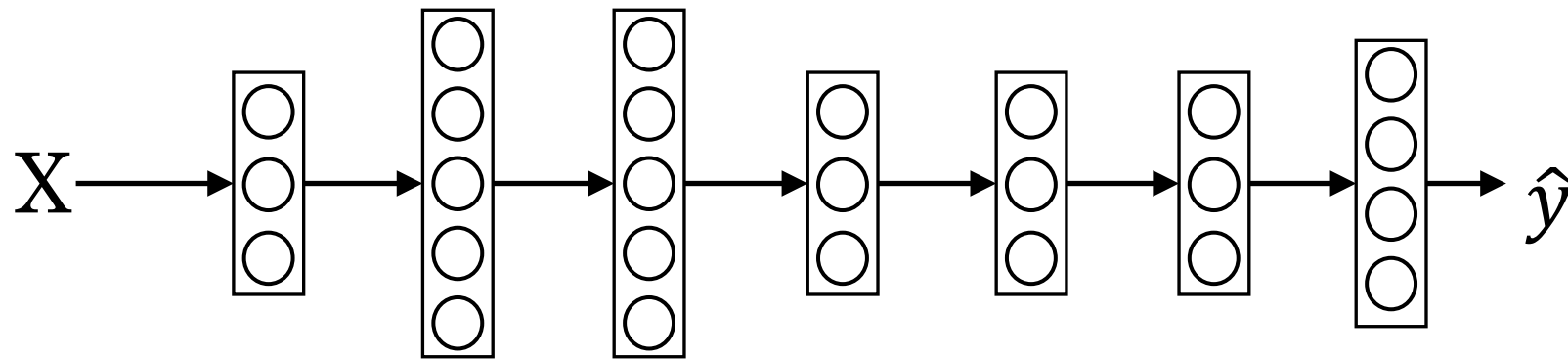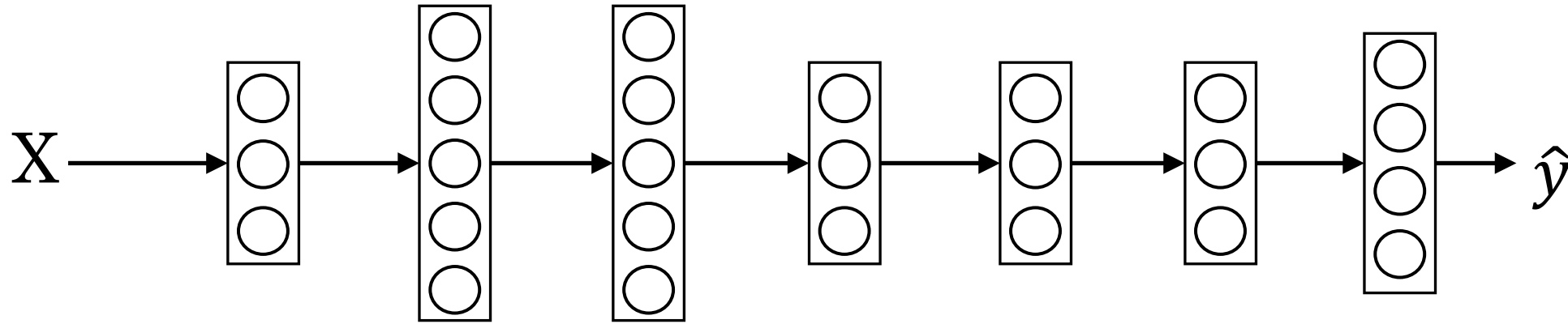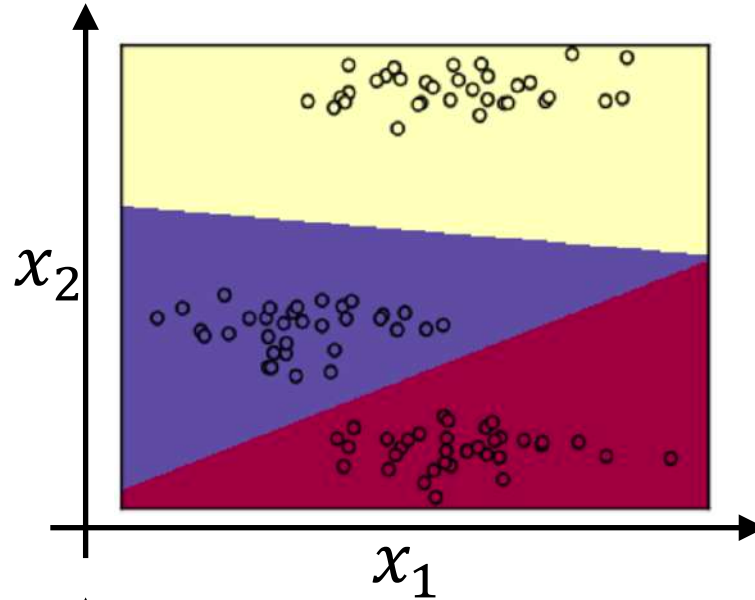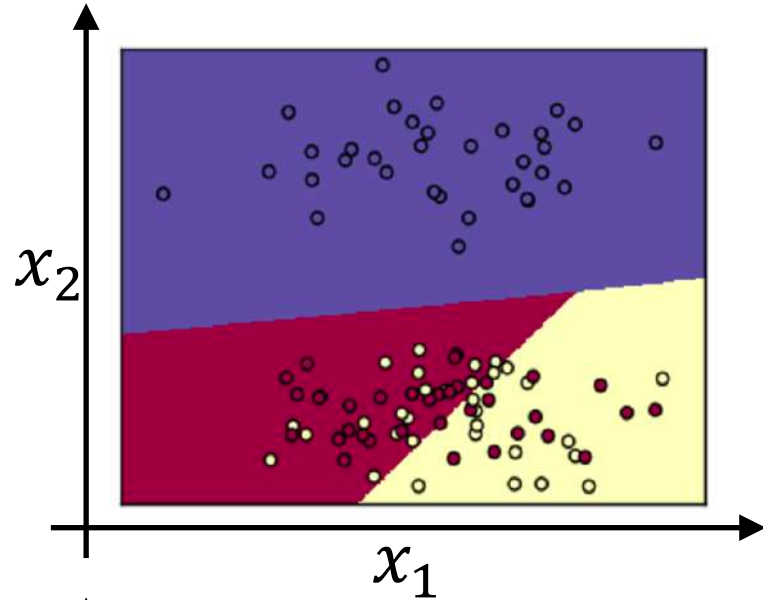
Softmax examples

deeplearning.ai

Programming
Frameworks

Deep Learning
frameworks

Deep learning frameworks

- Caffe/Caffe2
- CNTK
- DL4J
- Keras
- Lasagne
- mxnet
- PaddlePaddle
- TensorFlow
- Theano
- Torch

Choosing deep learning frameworks
- Ease of programming (development and deployment)
- Running speed
- Truly open (open source with good governance)

Andrew Ng

Programming
Frameworks
_____

TensorFlow

deeplearning.ai

Motivating problem

$$J(\omega) = \boxed{\omega^2 - 10\omega + 25}$$
(cost)

$(\omega - 5)^2$

$\omega = 5$

$J(W, b)$
↑ ↑