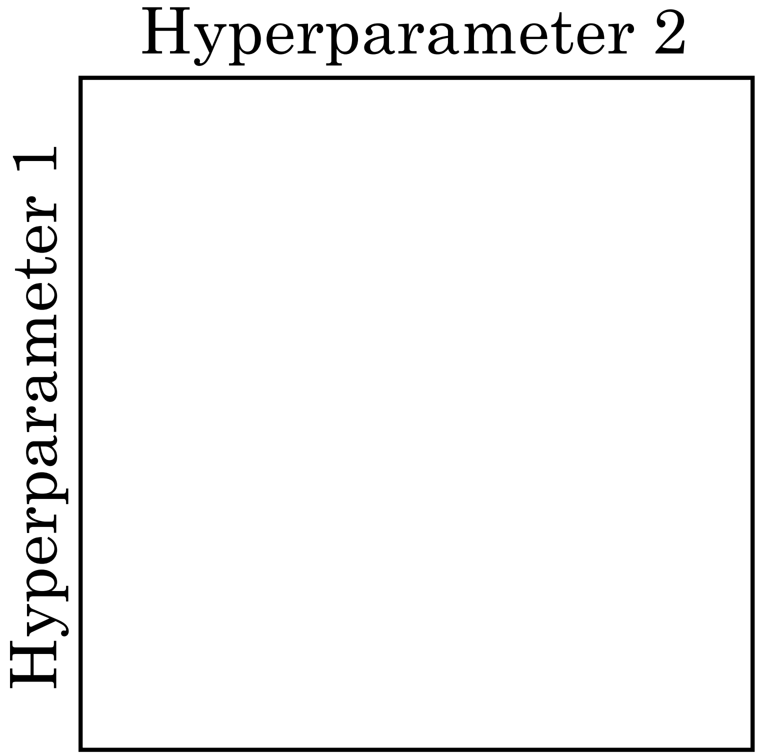
deeplearning.ai

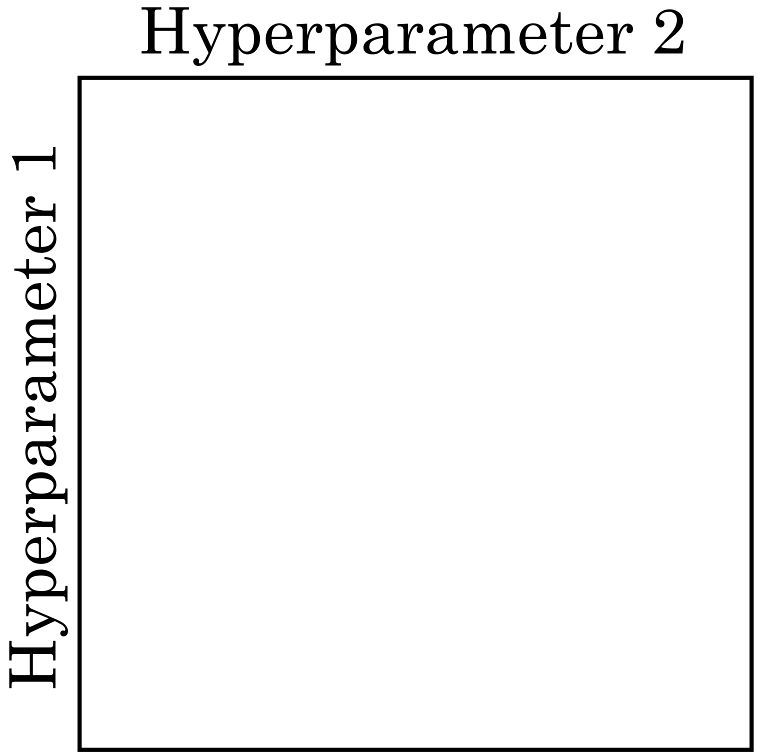
Hyperparameter tuning

Tuning process

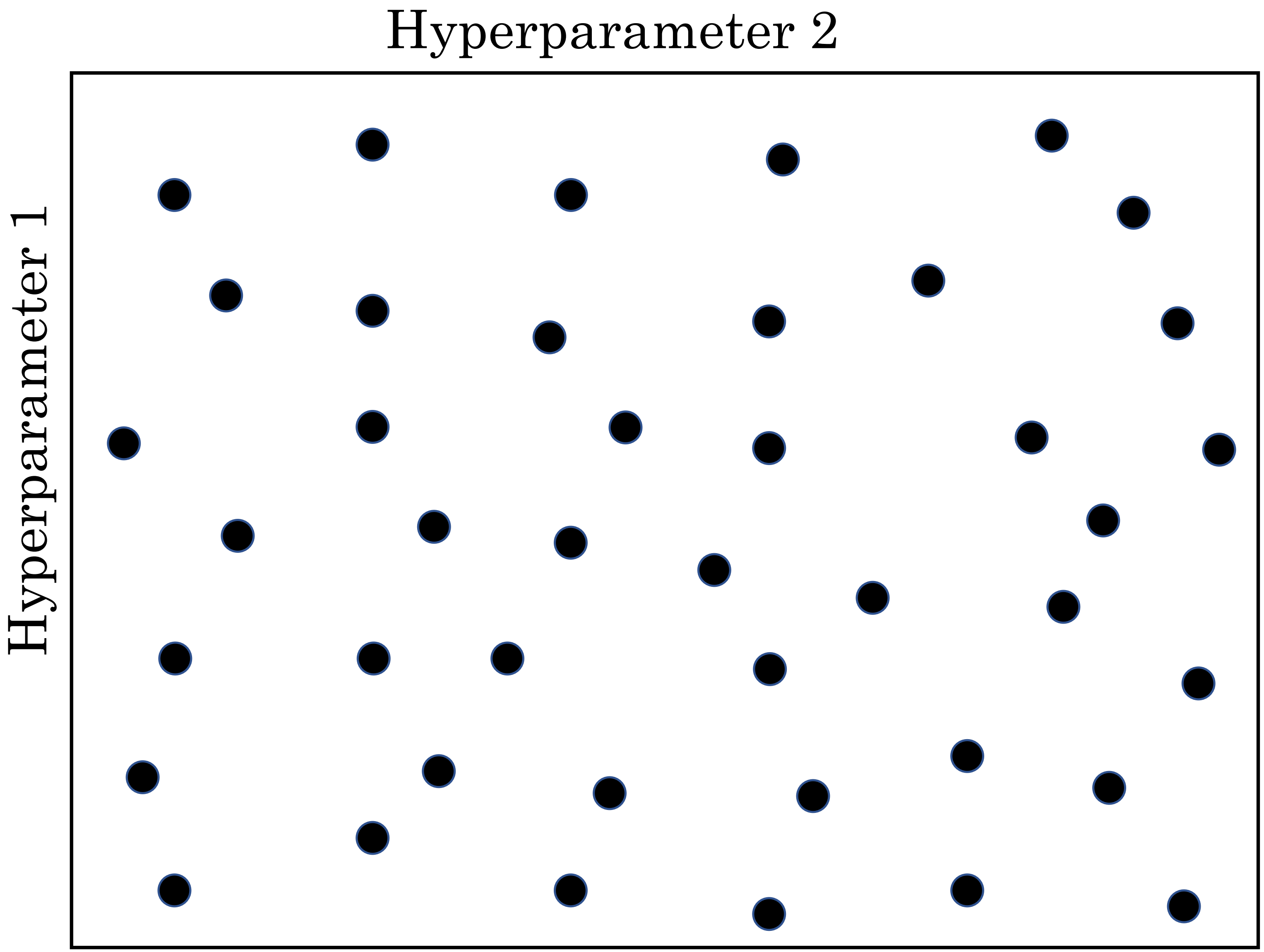
Hyperparameters

Try random values: Don’t use a grid





Coarse to fine



Andrew Ng

deeplearning.ai

Hyperparameter tuning

Using an appropriate scale to pick hyperparameters

Picking hyperparameters at random

Appropriate scale for hyperparameters

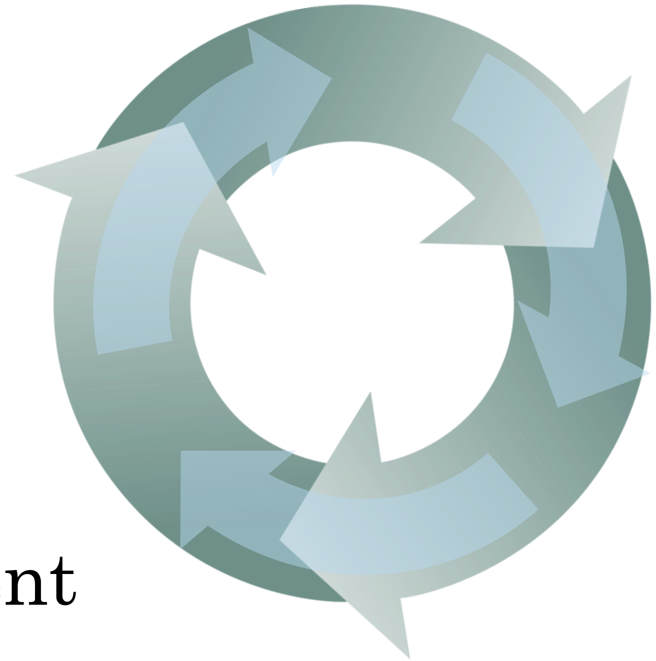
Hyperparameters for exponentially weighted averages

Hyperparameters tuning

deeplearning.ai

Hyperparameters tuning in practice: Pandas vs. Caviar

Re-test hyperparameters occasionally



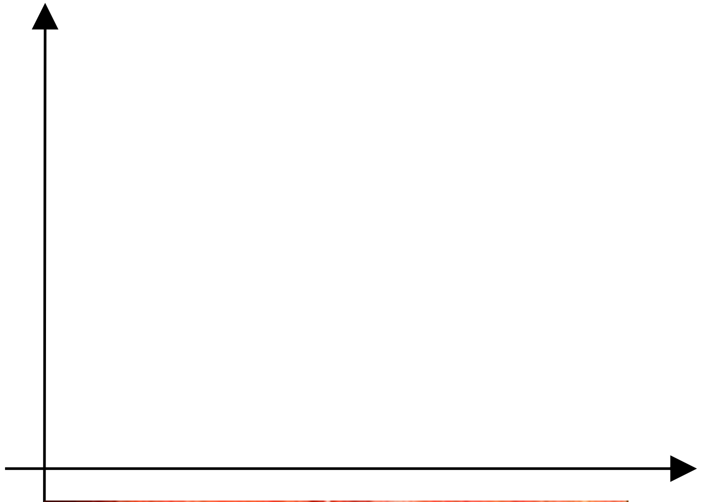
- NLP, Vision, Speech, Ads, logistics, ….

- Intuitions do get stale. Re-evaluate occasionally.

Babysitting one model

Panda

Training many models in parallel



Caviar

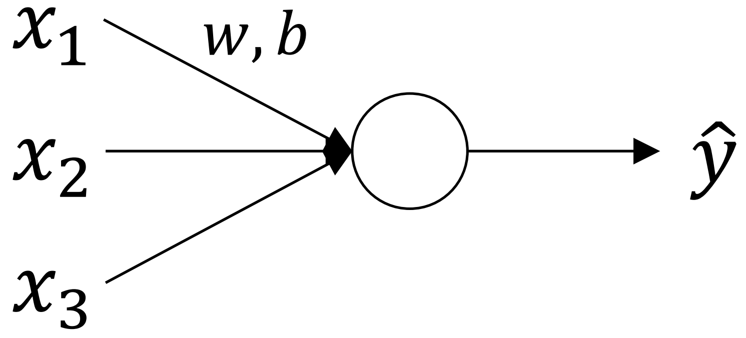
Andrew Ng

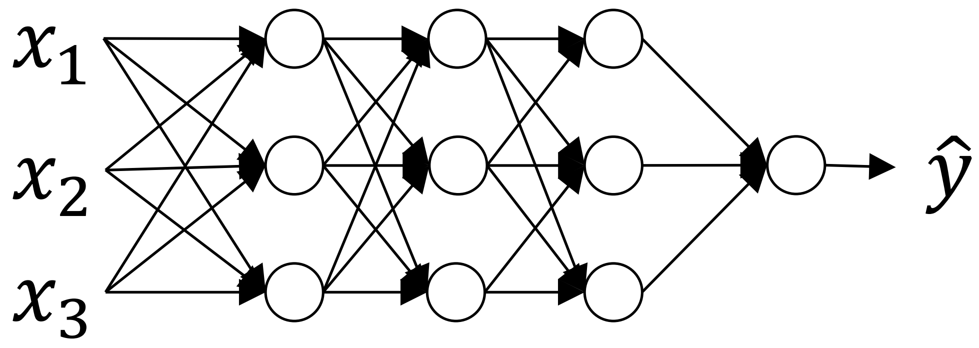
deeplearning.ai

Batch Normalization

Normalizing activations in a network

Normalizing inputs to speed up learning





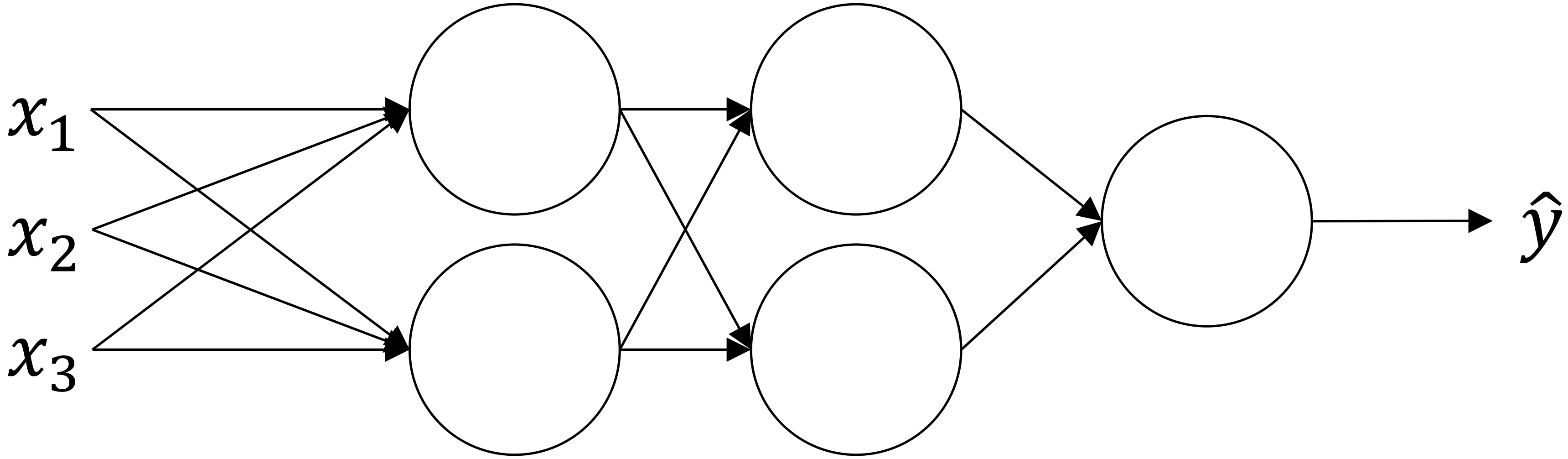
Implementing Batch Norm

Batch Normalization

deeplearning.ai

Fitting Batch Norm into a neural network

Adding Batch Norm to a network



Andrew Ng

Working with mini-batches

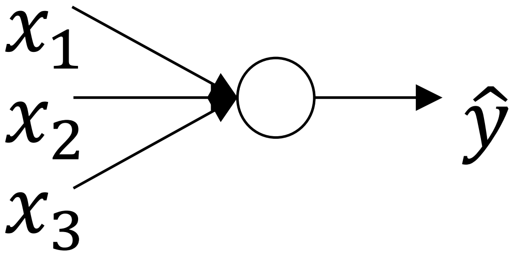
Implementing gradient descent

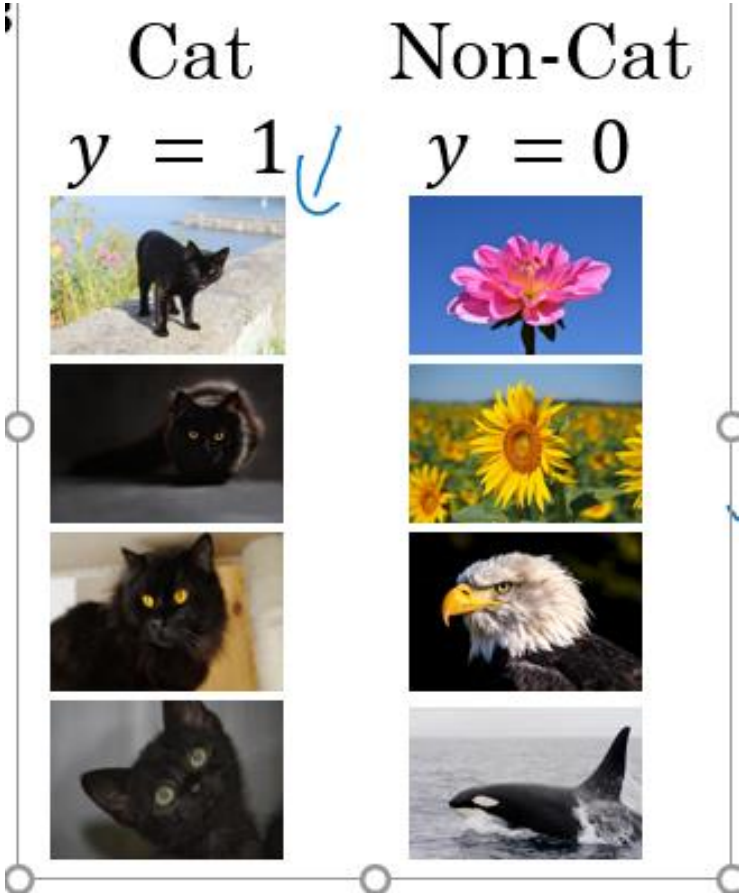
Batch Normalization

deeplearning.ai

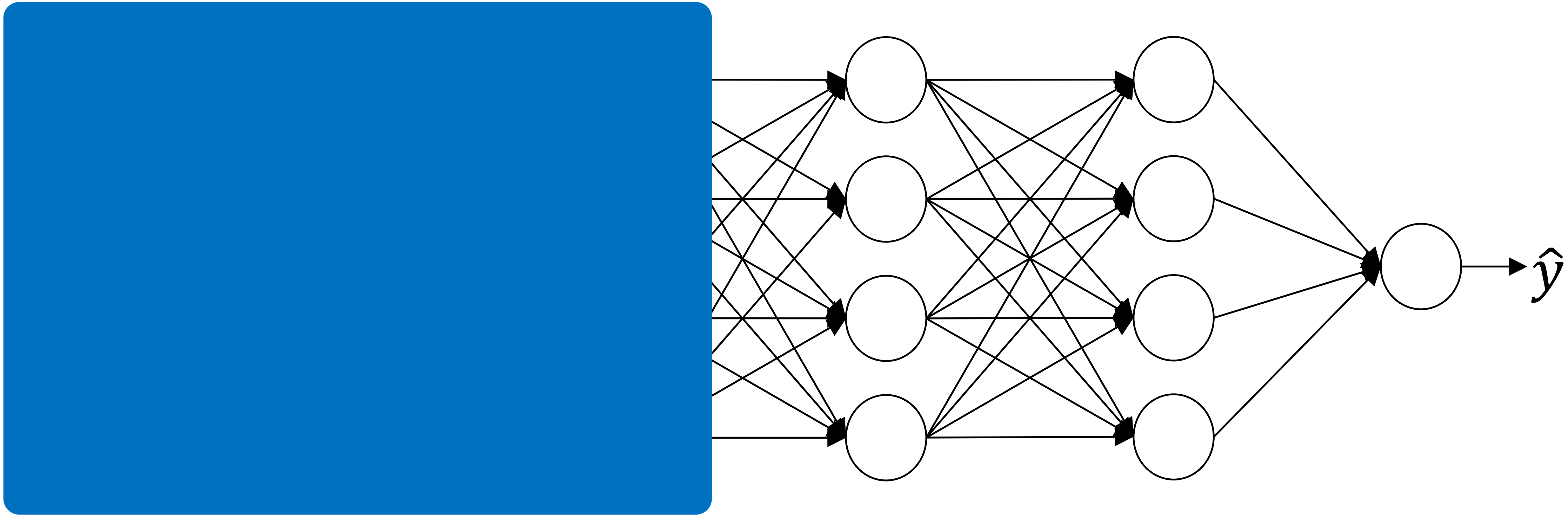
Why does Batch Norm work?

Learning on shifting input distribution





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.

deeplearning.ai

Batch Normalization

Batch Norm at test time

Batch Norm at test time

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

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

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

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

deeplearning.ai

Batch Normalization

Batch Norm at test time

Batch Norm at test time

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

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

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

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

Andrew Ng

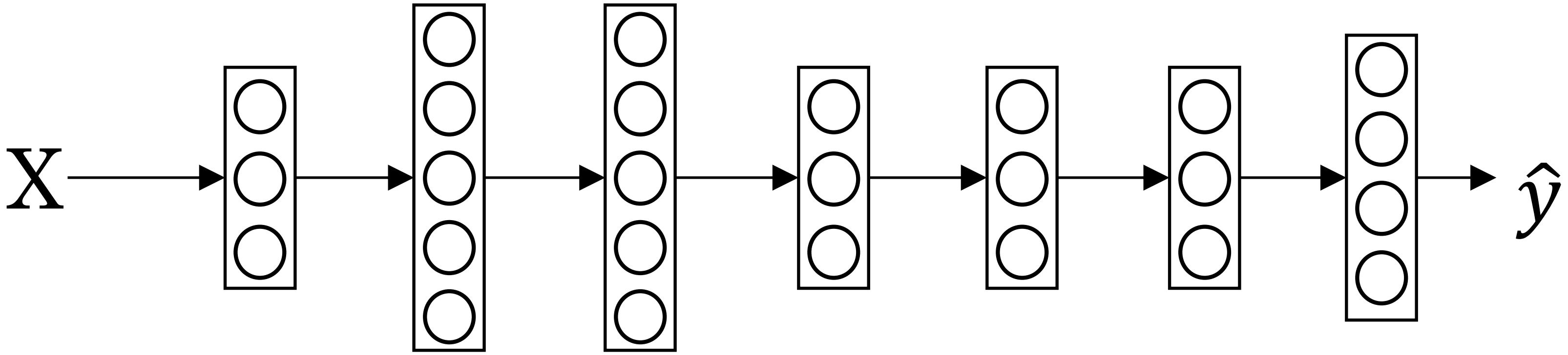
Multi-class classification

deeplearning.ai

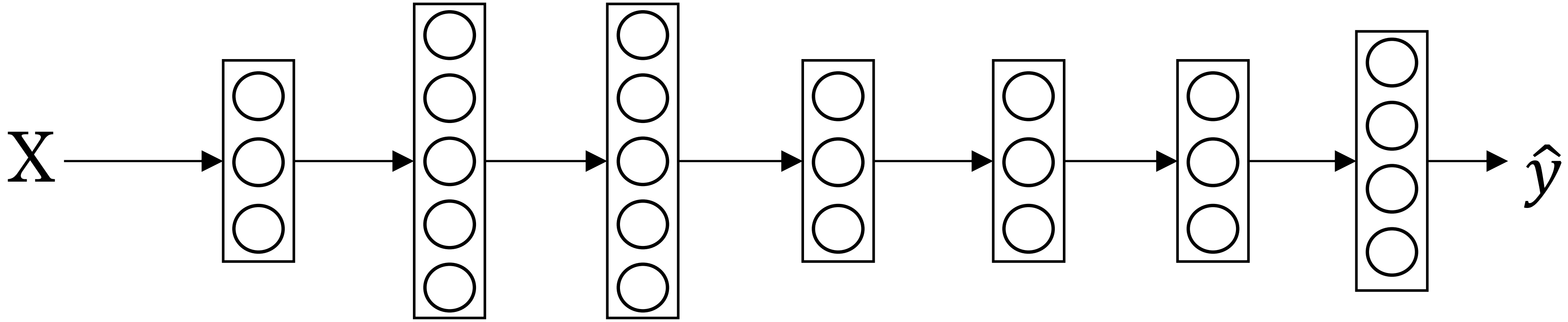
Softmax regression

Recognizing cats, dogs, and baby chicks

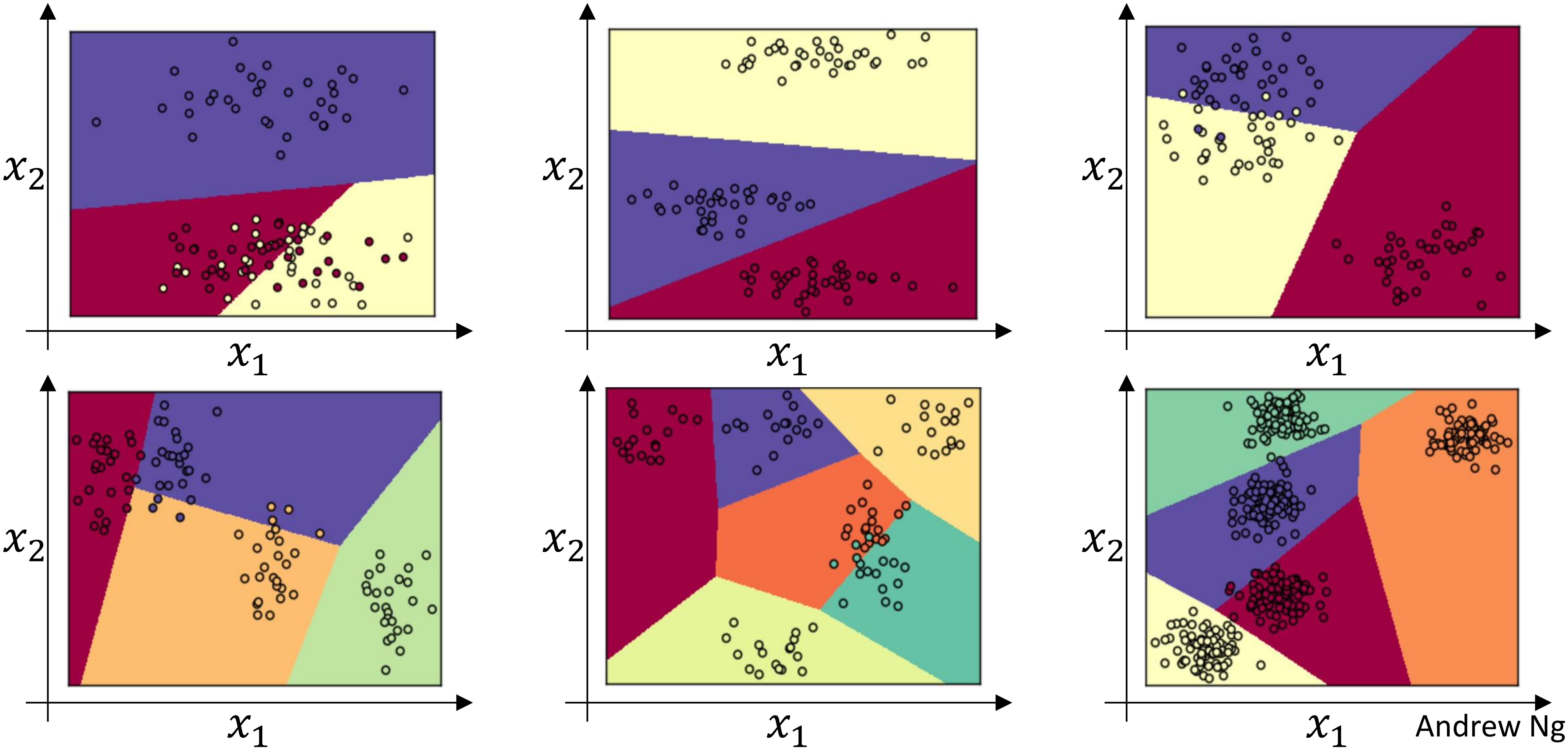
3 1 2 0 3 2 0 1



Softmax layer



Softmax examples



Andrew Ng

deeplearning.ai

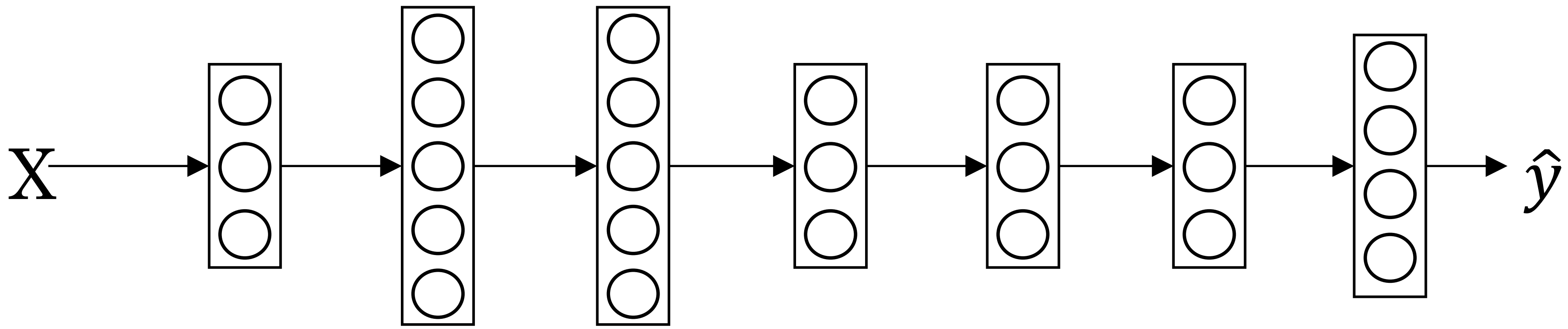
Multi-class classification

Trying a softmax classifier

Understanding softmax

Loss function

Summary of softmax classifier



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)

deeplearning.ai

Programming Frameworks

TensorFlow

Motivating problem

Code example

```
import numpy as np
import tensorflow as tf

coefficients = np.array([[1], [-20], [25]])

w = tf.Variable([0],dtype=tf.float32)
x = tf.placeholder(tf.float32, [3,1])
cost = x[0][0]*w**2 + x[1][0]*w + x[2][0]    # (w-5)**2
train = tf.train.GradientDescentOptimizer(0.01).minimize(cost)
init = tf.global_variables_initializer()
session = tf.Session()
session.run(init)
print(session.run(w))

for i in range(1000):
    session.run(train, feed_dict={x:coefficients})
print(session.run(w))
```

```
with tf.Session() as session:
    session.run(init)
    print(session.run(w))
```

Andrew Ng