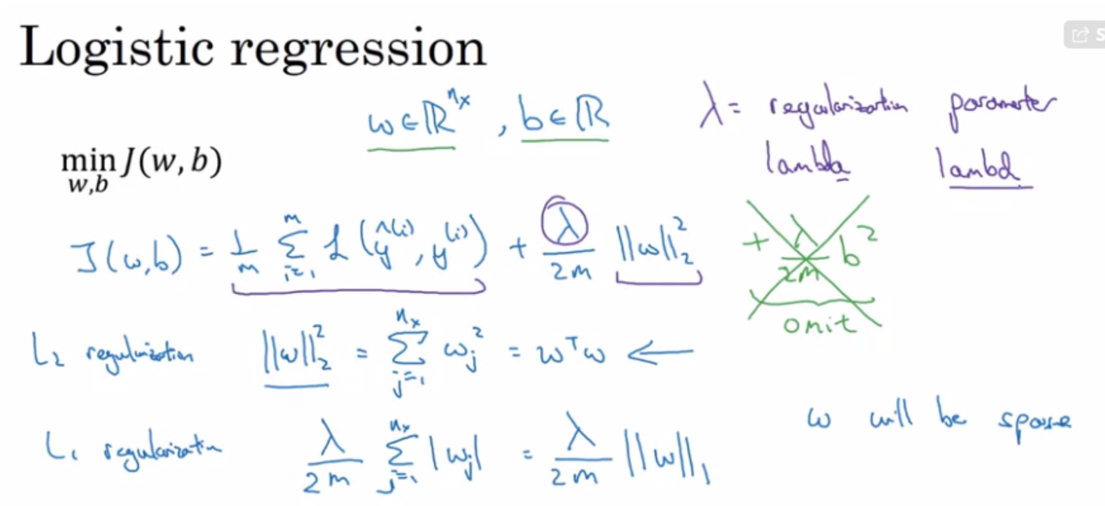
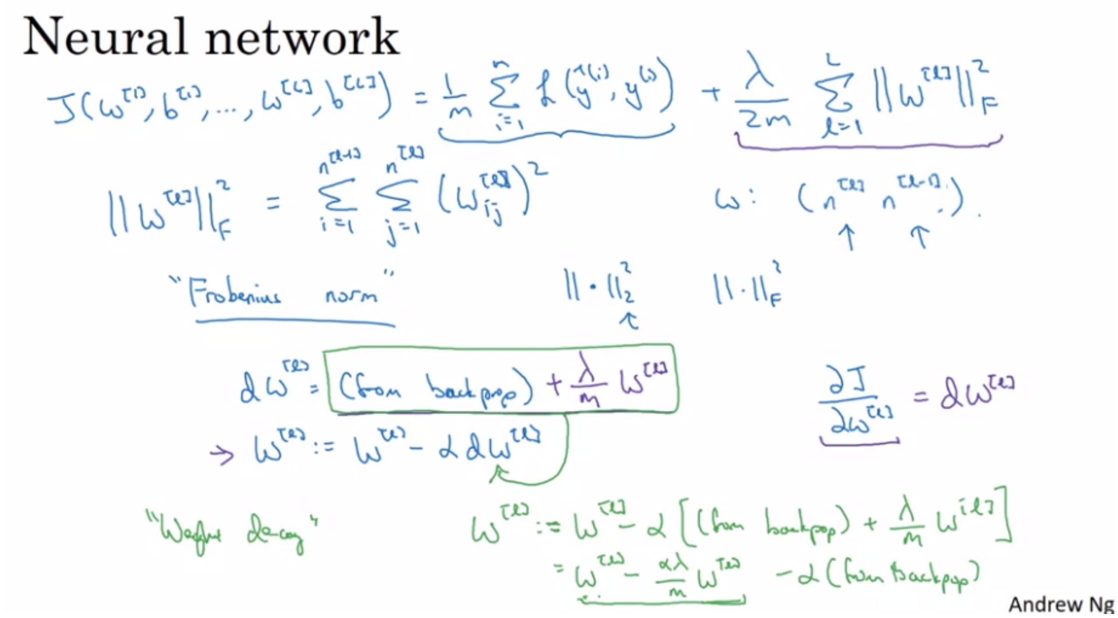
Week 1-2 Regularization

笔记本:	DL 2 - Deep NN Hyperparameter Tunning, Regularization & Optimization
创建时间:	2021/1/9 09:25
更新时间:	2021/1/9 09:42



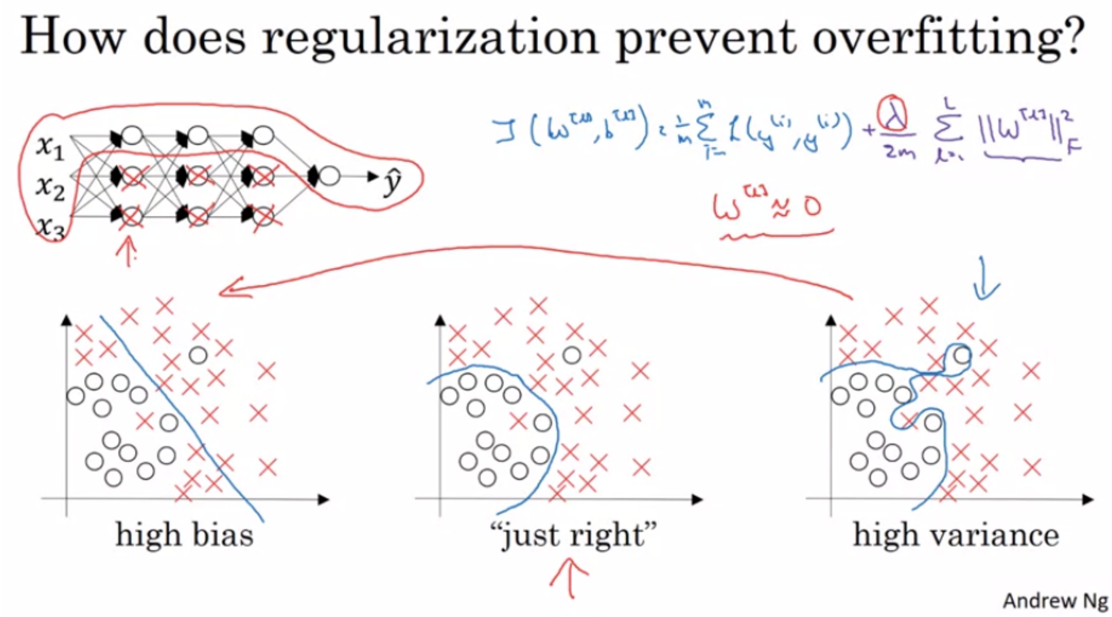
L2 regularization (L2 norm)
w will be sparse for L1 regularization

For NN: (Forbenius norm)

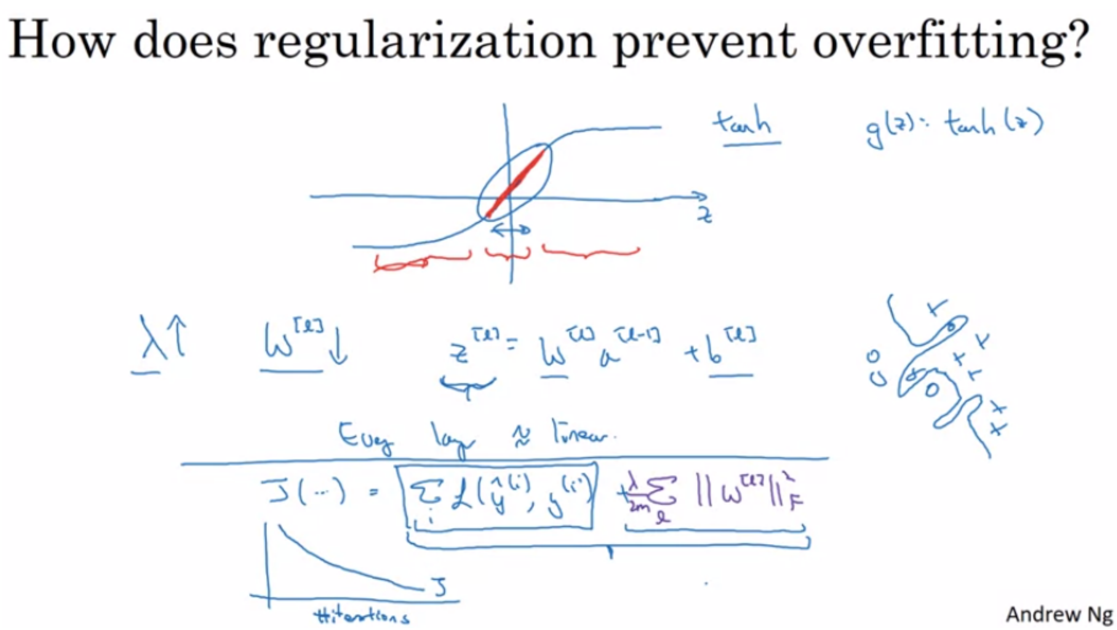


(also called weight decay)

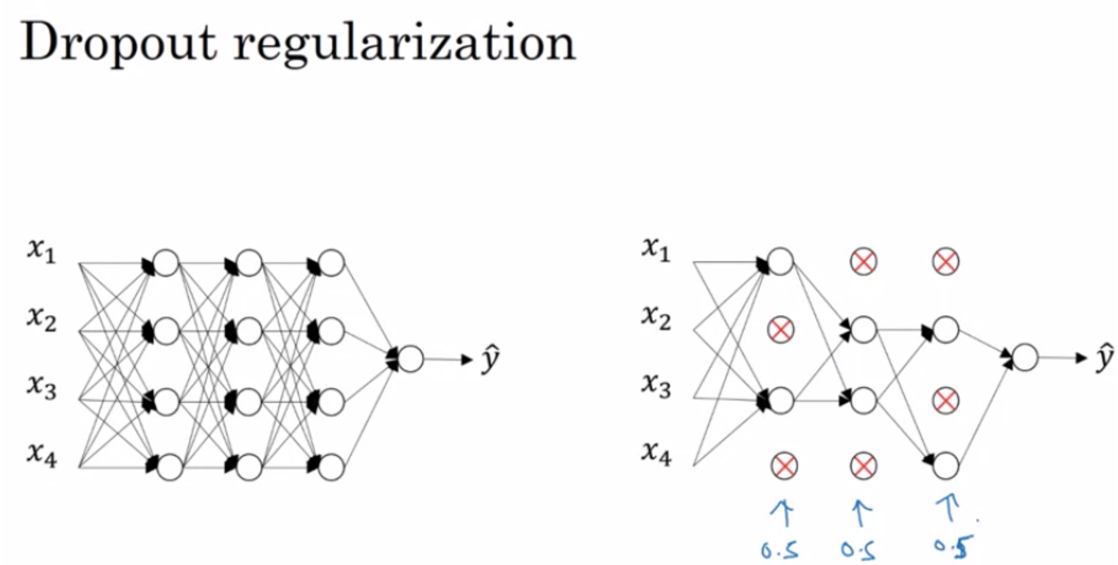
Why it works?



also, every layer approximately linear



Dropout Regularization



trained with randomly reduced NN

Implementing dropout ("Inverted dropout")

Illustrate with layer $l=3$.   keep-prob = 0.8   0.2

→ `d3 = np.random.rand(a3.shape[0], a3.shape[1]) < keep-prob`

`a3 = np.multiply(a3, d3)`   `# a3 *= d3.`

→ $\boxed{\texttt{a3 /= } \cancel{0.8} \texttt{ keep-prob}}$ ←

50 units. ⇝ 10 units shut off

$$z^{[4]} = W^{[4]} \cdot a^{[3]} + b^{[4]}$$

reduced by 20%.   Test

/= 0.8

(Inverted dropout makes sure that Exp(a) stays the same)

Making predictions at test time

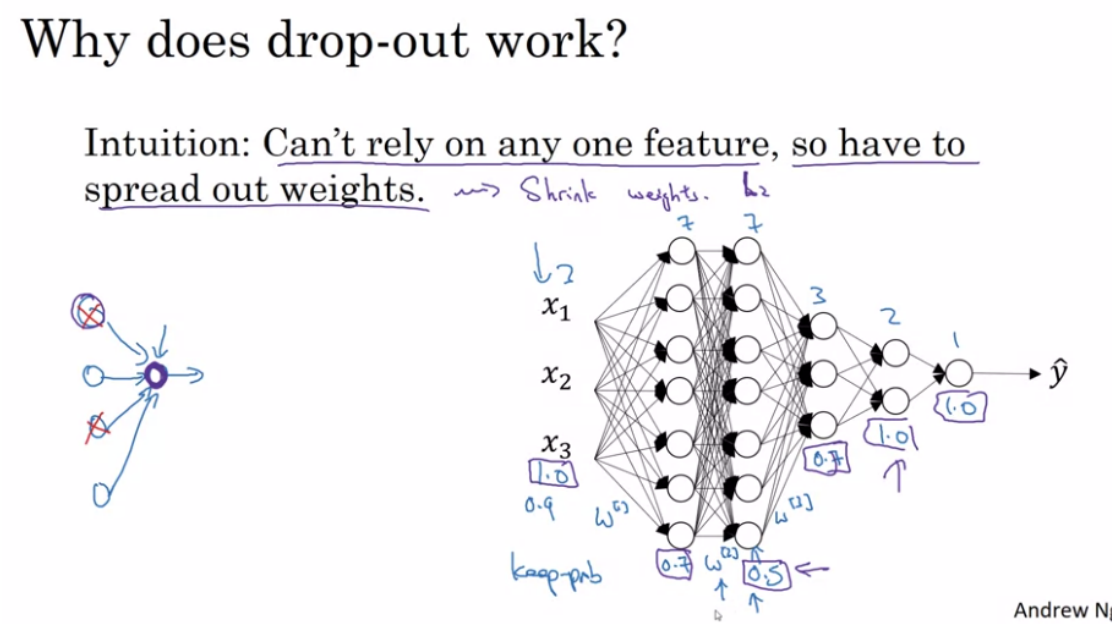
$$a^{[0]} = X$$

No drop out.

$$z^{[1]} = W^{[1]} a^{[0]} + b^{[1]}$$
$$a^{[1]} = g^{[1]}(z^{[1]})$$
$$z^{[2]} = W^{[2]} a^{[1]} + b^{[2]}$$
$$a^{[2]} = \ldots$$
$$\downarrow$$
$$\hat{y}$$

/= keep-prob

Why dropout works? (usually used in CV)

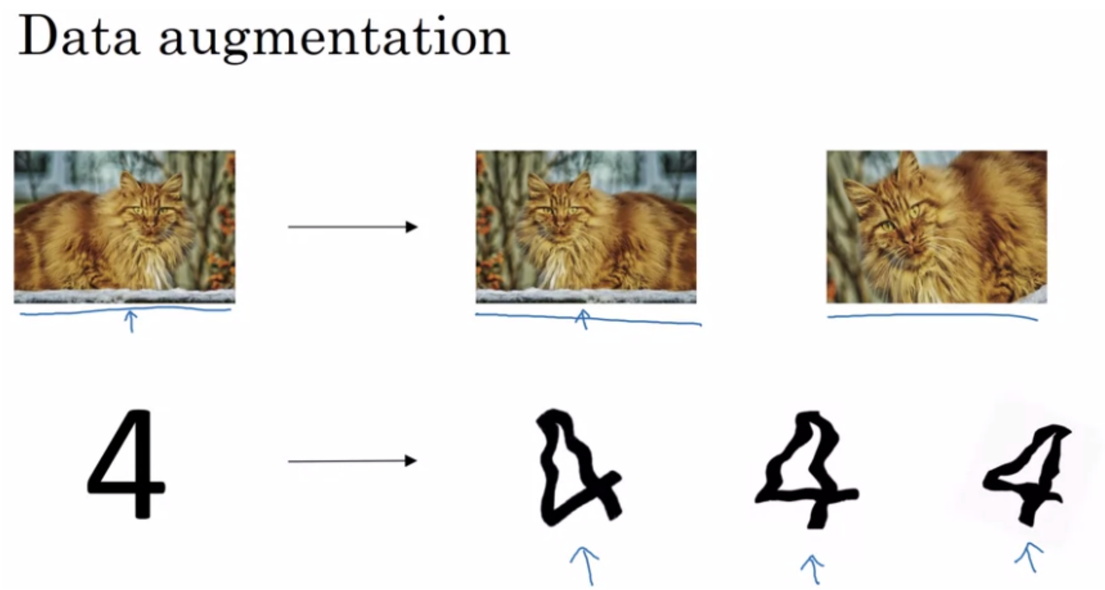


tune keep-prob for different layers

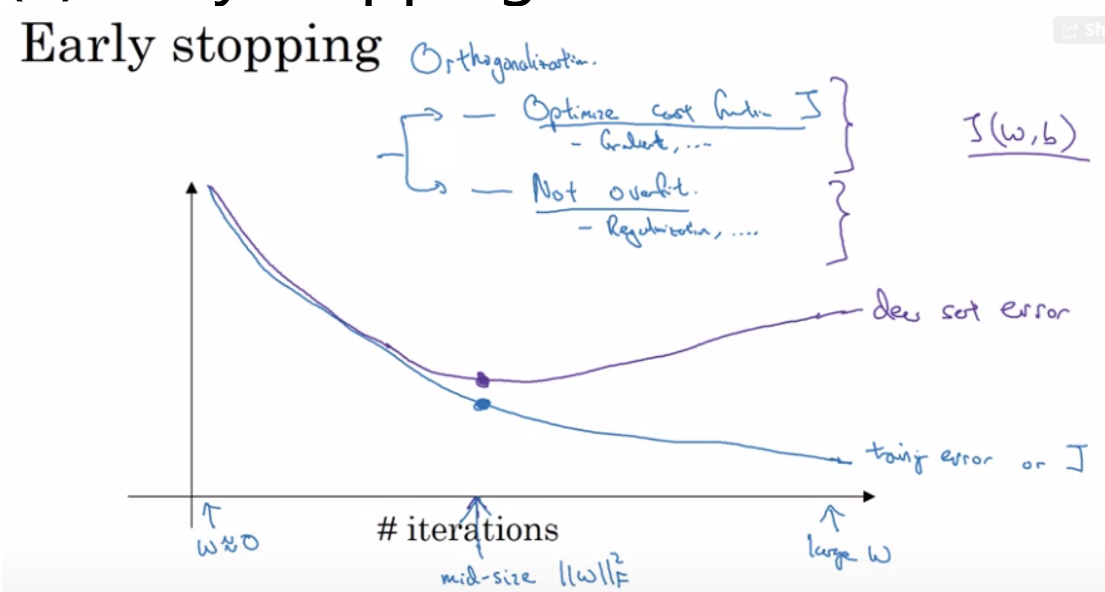
Downside: cost function is not well-defined

Other methods:

(1) Data augmentation



(2) Early stopping



Downside: (Orthogonalization, we do one task at a time, but early stopping enquires we do both tasks at the same time)