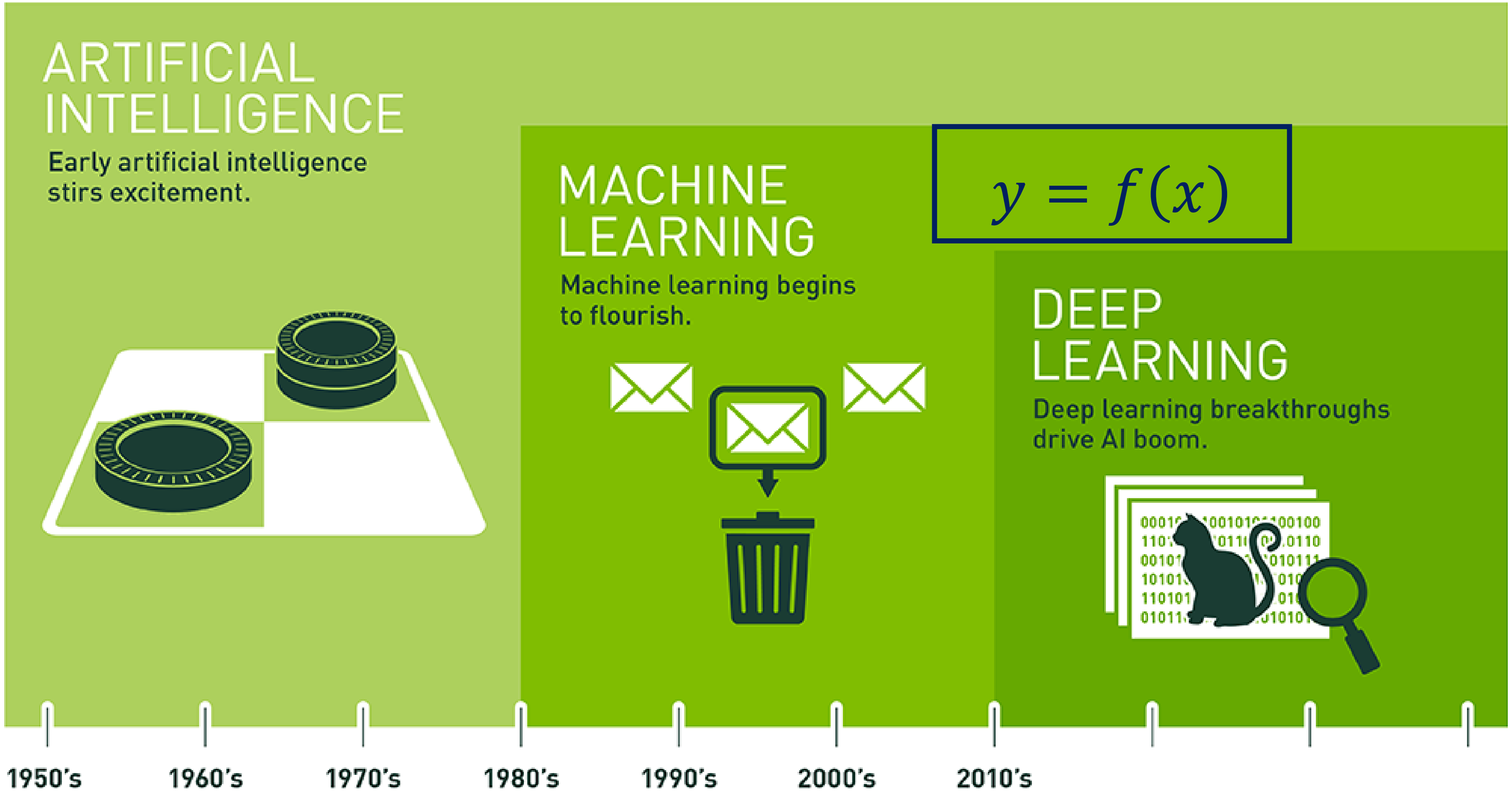
Regression vs Classification

y learned is continuous → Regression
y learned is categorical → Classification



https://blogs.nvidia.com/blog/2016/07/29/whats-difference-artificial-intelligence-machine-learning-deep-learning-ai/

Notations

x_i

No	age	t1	t2	t3	t4	t5	t6	time	Step frequency	n1	n2	n3	n4	n5	n6	px	py	pz	Steps	Gender	TUG	y1	BBS	y2
1	70	1.76	2.64	6.24	7.02	10	12.8	11	2.285	80	120	282	317	453	575	11.67	1.809	-1.99	13	F	11	0	26	0
2	86	1.64	2.6	5.82	7.27	10.4	12.6	11	1.934	75	118	263	328	470	570	11.14	2.302	-4.651	12	F	11	0	24	0
3	76	1.76	2.93	6.27	7.04	10.3	12.8	11	2.109	80	133	283	318	465	575	11.53	2.169	-3.253	14	F	11	0	22	1
4	70	2.38	3.29	5.58	6.47	9.02	10.4	8	2.461	108	149	252	292	407	468	11.6	1.838	-3.138	12	F	8	0	24	0
5	66	3.09	4.07	6.6	7.4	10.2	12.1	9	2.461	140	184	298	334	462	545	11.55	2.531	-2.742	12	F	9	0	26	0
6	79	1.76	2.91	5.87	6.6	10.2	12.8	11	2.109	80	132	265	298	462	575	[illegible]	1.788	-1.349	13	F	11	0	26	0
7	85	1.2	2.33	5.42	8.31	12.1	17.2	16	2.988	55	106	245	375	545	775	[illegible]	2.203	-4.89	17	M	16	1	18	1
8	81	1.64	2.93	5.98	7.47	10.9	13.6	12	1.758	75	133	270	337	493	615	11.1	2.667	-4.594	10	F	12	0	24	0
9	82	0.64	1.47	4.76	5.76	9.36	11.6	11	2.109	30	67	215	260	422	525	11.26	4.1	-2.693	14	M	11	0	24	0
10	69	1.64	2.49	5.02	5.98	9.82	12.6	11	2.637	75	113	227	270	443	570	11.27	3.292	-3.522	13	F	11	0	20	1
11	84	0.64	1.4	5.67	7.29	11.5	14.6	14	1.934	30	64	256	329	520	660	11.53	2.335	-2.999	15	M	14	1	26	0
12	69	1.09	1.98	5	5.62	8.38	10.1	9	2.109	50	90	226	254	378	455	11.15	1.919	-4.608	11	M	9	0	26	0
13	73	1.09	2.13	6.78	8.38	12.4	17.1	16	3.691	50	97	306	378	558	770	11.46	2.264	-3.333	16	F	16	1	14	1
14	81	0.64	1.87	9.24	11.2	19	22.6	22	1.934	30	85	417	507	857	1020	11.58	2.511	-2.157	27	M	22	1	24	0
15	80	0.76	1.71	3.98	5	7.58	9.76	9	2.109	35	78	180	226	342	440	11.33	2.821	-3.595	10	M	9	0	26	0
16	88	0.98	2.13	6.31	7.44	11.5	14	13	1.934	45	97	285	336	518	630	11.38	2.498	-3.702	16	M	14	1	26	0
17	81	1.09	2.09	4.18	5.16	7.76	10.1	9	2.285	50	95	189	233	350	455	11.21	2.241	-4.337	10	M	9	0	28	0
18	76	1.76	2.64	5.87	6.98	9.98	12.8	11	1.406	80	120	265	315	450	575	11.33	2.679	-3.736	10	M	11	0	26	0
19	69	0.36	3.76	13.3	16.7	24.2	29.4	29	3.691	17	170	598	753	1090	1322	11.31	1.361	-4.171	28	F	29	1	10	1
20	75	1.98	2.93	5.98	7.91	12.2	15	13	1.934	90	133	270	357	551	675	11.5	2.202	-1.495	14	M	13	0	28	0
21	87	1.53	3.2	10.9	13.8	21.3	26.5	25	2.9	70	145	492	624	960	1195	11.6	2.199	-2.54	19	F	25	1	16	1
22	72	0.2	1.02	3.36	4.11	7.42	10.2	10	1.758	10	47	152	186	335	460	11.52	2.658	-2.081	9	M	10	0	28	0
23	109	0.64	1.93	5.04	5.71	9.13	10.6	10	2.285	30	88	228	258	412	480	11.51	2.056	-3.158	15	F	10	0	28	0

x_i^n

x^n

$\hat{y}^n$

Mechanism for computer to learn from data

- Define a function to be learned: $y^n = f(x^n)$
- Define a loss function $\mathcal{L}(f)$ to describe the error between y^n and $\hat{y}^n$
- Find the optimal parameters that minimize $\mathcal{L}(f)$

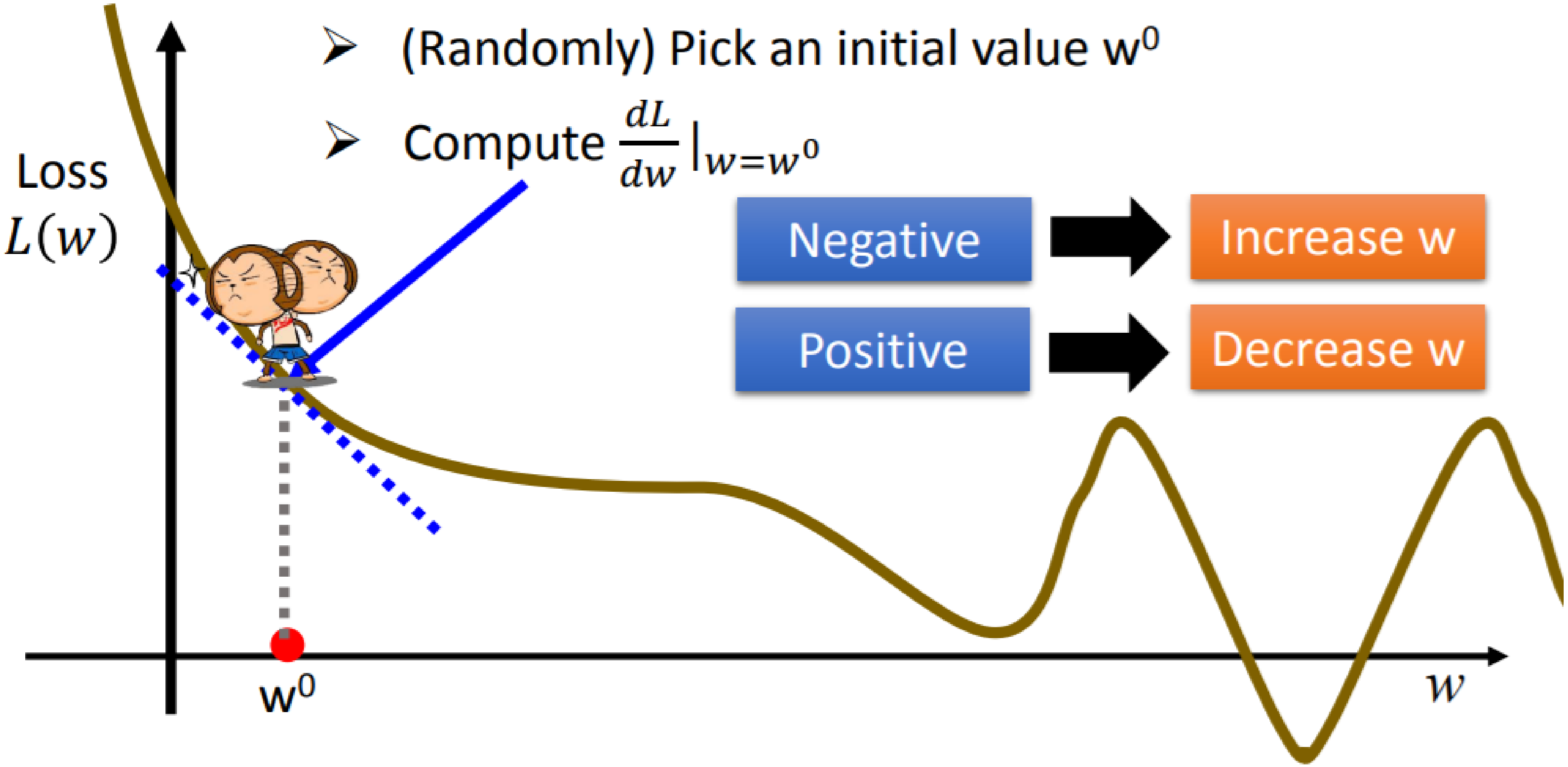
ML model (1) – Linear regression

- Linear model: $y^n = \sum_i (w_i x_i^n) + b$
- Loss function: $L(w,b) = \sum_{n=1}^{N} (\hat{y}^n - y^n)^2 \; = \; \sum_{n=1}^{N} (\hat{y}^n - (\sum_i (w_i x_i^n) + b\,))^2$
- Find the optimal parameters that minimize loss: $\mathbf{arg}\ \underset{w,\,b}{min}\ L(w,b)$

Use gradient decent to find optimal parameters

$$w^* = arg \min_{w} L(w)$$

- Consider loss function $L(w)$ with one parameter w:
  - ➢ (Randomly) Pick an initial value w^0
  - ➢ Compute $\frac{dL}{dw}|_{w=w^0}$



Reference: 李弘毅 ML Lecture 1 https://youtu.be/CXgbekl66jc

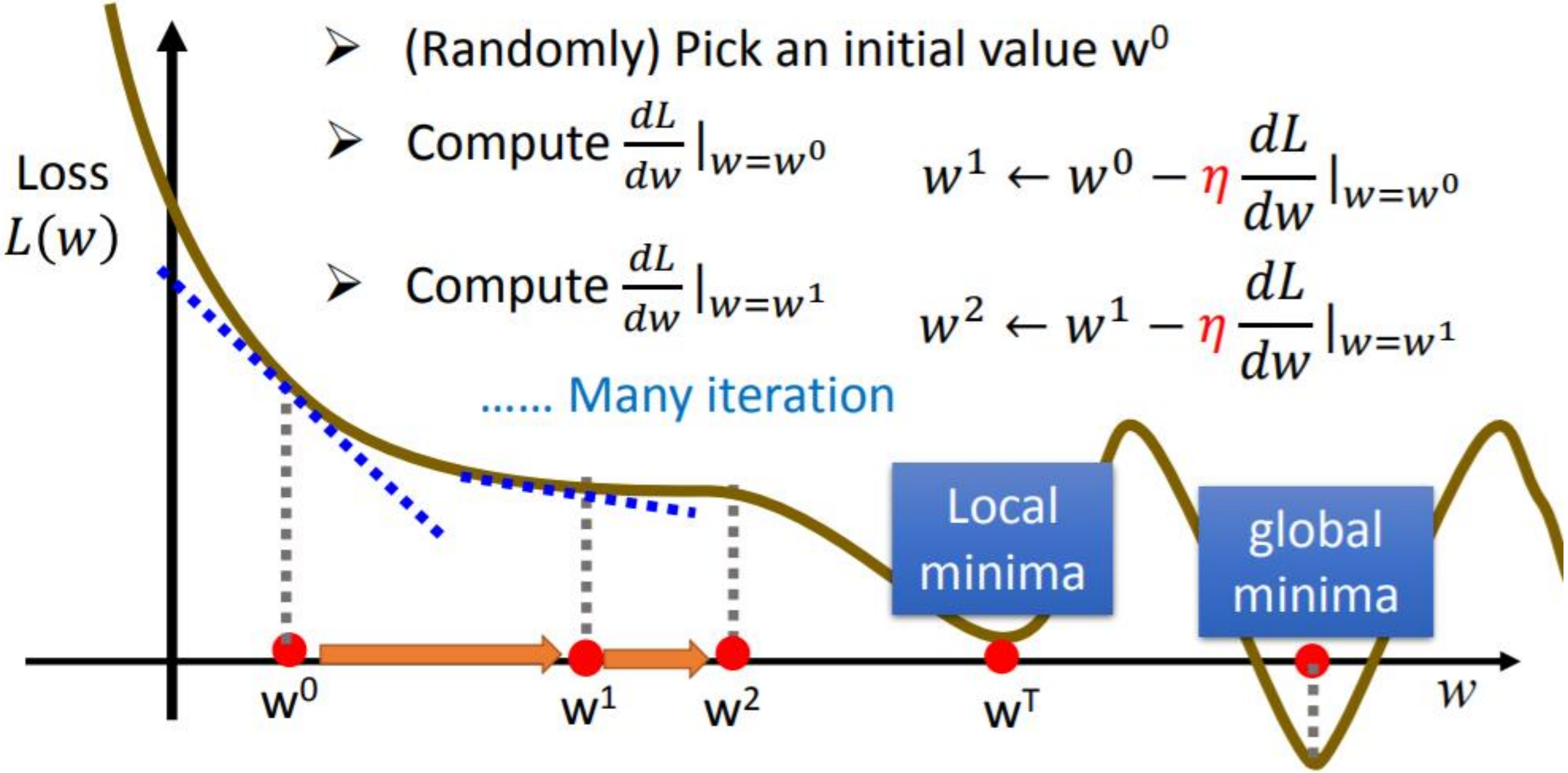
Gradient decent

$$w^* = arg \min_{w} L(w)$$

- Consider loss function $L(w)$ with one parameter w:

  - (Randomly) Pick an initial value w^0
  - Compute $\frac{dL}{dw}|_{w=w^0}$ $\quad w^1 \leftarrow w^0 - \eta \frac{dL}{dw}|_{w=w^0}$
  - Compute $\frac{dL}{dw}|_{w=w^1}$ $\quad w^2 \leftarrow w^1 - \eta \frac{dL}{dw}|_{w=w^1}$

…… Many iteration



Reference: 李弘毅 ML Lecture 1 https://youtu.be/CXgbekl66jc

Gradient decent to find two parameters w^* and b^*

- How about two parameters? $w^*, b^* = arg \min_{w,b} L(w,b)$

$$\begin{bmatrix} \frac{\partial L}{\partial w} \\ \frac{\partial L}{\partial b} \end{bmatrix} \text{gradient}$$

  - (Randomly) Pick an initial value w^0, b^0
  - Compute $\frac{\partial L}{\partial w}|_{w=w^0,b=b^0}, \frac{\partial L}{\partial b}|_{w=w^0,b=b^0}$

$$w^1 \leftarrow w^0 - \eta \frac{\partial L}{\partial w}|_{w=w^0,b=b^0} \qquad b^1 \leftarrow b^0 - \eta \frac{\partial L}{\partial b}|_{w=w^0,b=b^0}$$

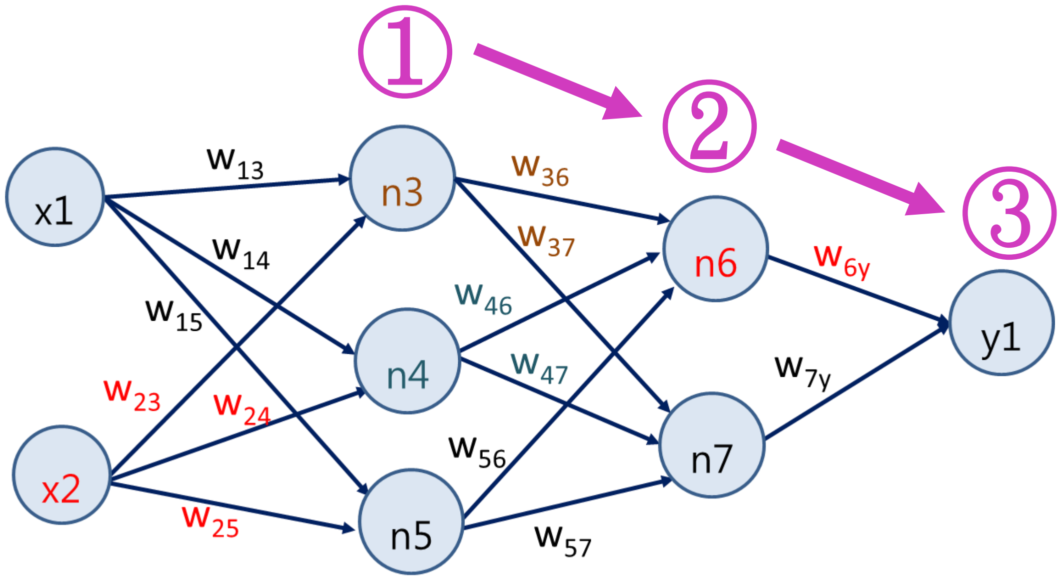
  - Compute $\frac{\partial L}{\partial w}|_{w=w^1,b=b^1}, \frac{\partial L}{\partial b}|_{w=w^1,b=b^1}$

$$w^2 \leftarrow w^1 - \eta \frac{\partial L}{\partial w}|_{w=w^1,b=b^1} \qquad b^2 \leftarrow b^1 - \eta \frac{\partial L}{\partial b}|_{w=w^1,b=b^1}$$

Reference: 李弘毅 ML Lecture 1 https://youtu.be/CXgbekl66jc

Machine learning model (2) – Deep learning

Define a function to be learned: $y^n = f(x^n)$



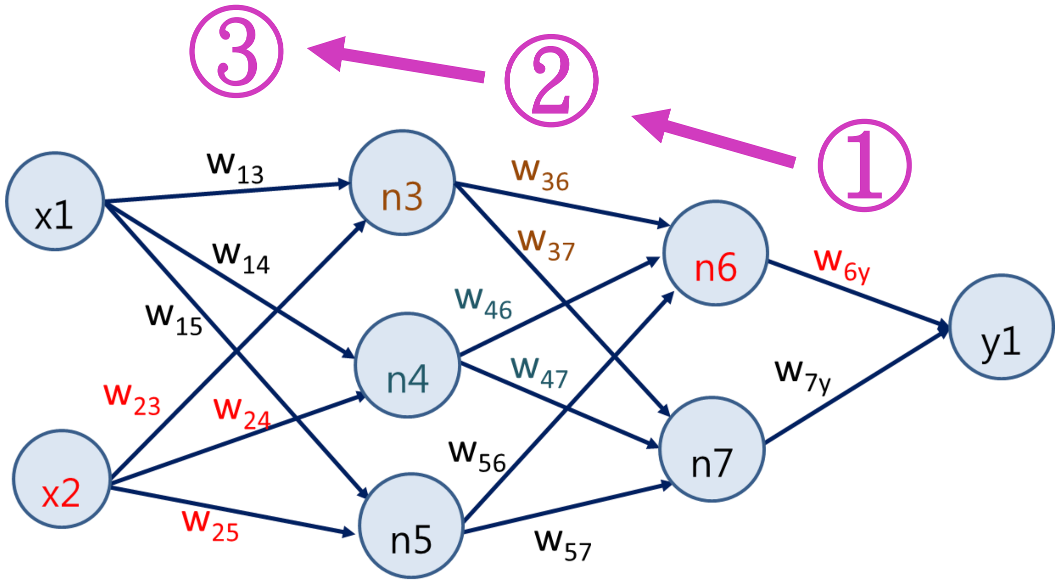
①
$$n_3 = \sigma(x_1 * w_{13} + x_2 * w_{23} + b_3)$$
$$n_4 = \sigma(x_1 * w_{14} + x_2 * w_{24} + b_4)$$
$$n_5 = \sigma(x_1 * w_{15} + x_2 * w_{25} + b_5)$$

②
$$n_6 = \sigma(n_3 * w_{36} + n_4 * w_{46} + n_5 * w_{56} + b_6)$$
$$n_7 = \sigma(n_3 * w_{37} + n_4 * w_{47} + n_5 * w_{57} + b_7)$$

③
$$y_1 = \sigma(n_6 * w_{6y} + n_7 * w_{7y} + b_y)$$

Machine learning model (2) – Deep learning

Use gradient decent to find optimal parameters



$$L = g(y - \hat{y}) \qquad y = \sigma(n_6 * w_{6y} + n_7 * w_{7y} + b_y)$$

①

$$w_{6y} \leftarrow w_{6y} - \eta \frac{\partial L}{\partial w_{6y}} \qquad \frac{\partial L}{\partial w_{6y}} = \frac{\partial L}{\partial y}\frac{\partial y}{\partial w_{6y}}$$

$$w_{7y} \leftarrow w_{7y} - \eta \frac{\partial L}{\partial w_{7y}} \qquad \frac{\partial L}{\partial w_{7y}} = \frac{\partial L}{\partial y}\frac{\partial y}{\partial w_{7y}}$$

$$w_i \leftarrow w_i - \eta \frac{\partial e}{\partial w_i}$$

②

$$w_{57} \leftarrow w_{57} - \eta \frac{\partial L}{\partial w_{57}} \qquad \frac{\partial L}{\partial w_{57}} = \frac{\partial L}{\partial y}\frac{\partial y}{\partial n_7}\frac{\partial n_7}{\partial w_{57}}$$

$$n_7 = f(n_3 * w_{37} + n_4 * w_{47} + n_5 * w_{57} + b_7)$$

Jeffery Hinton

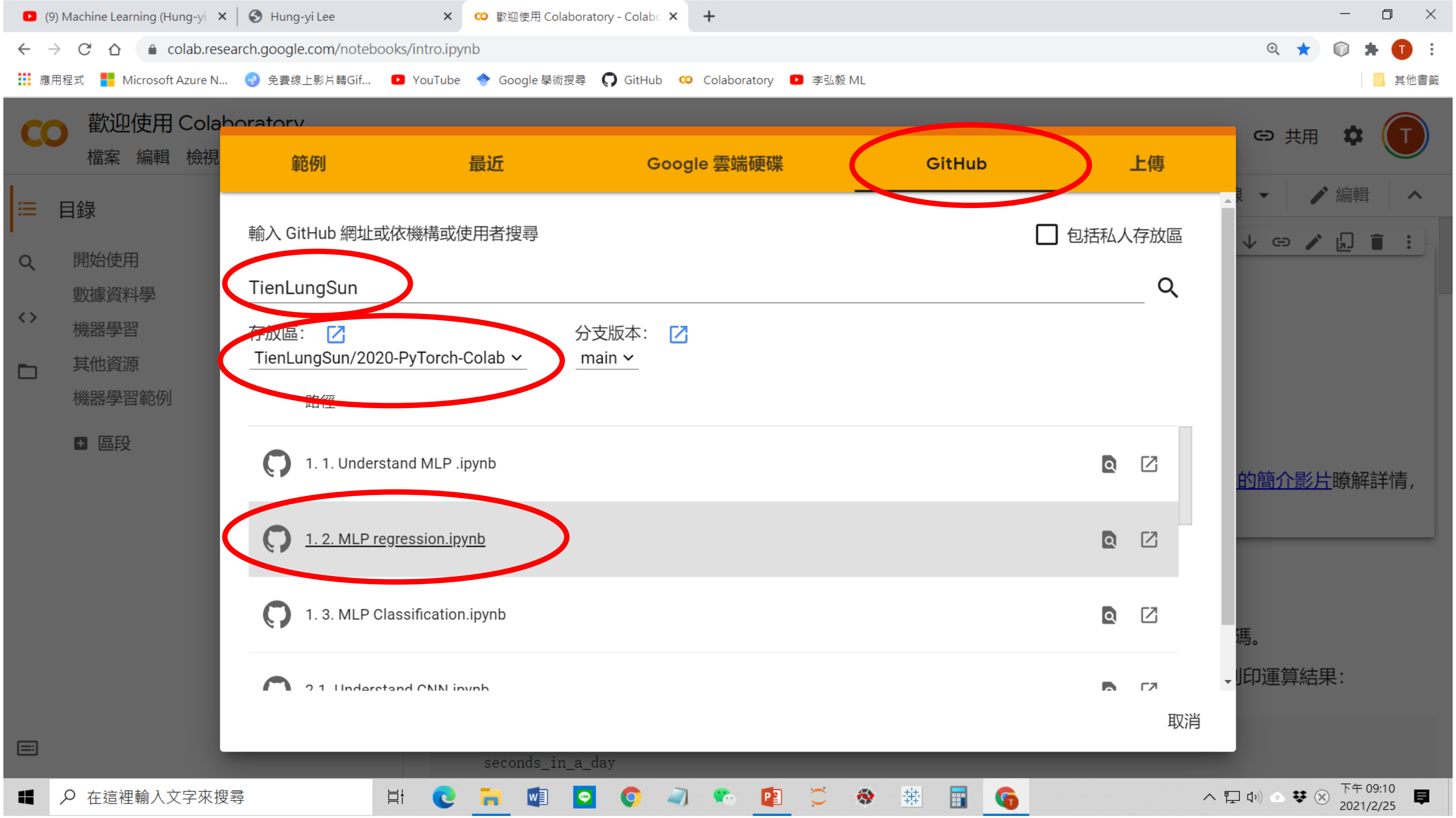
Geoffrey Hinton spent 30 years hammering away at an idea most other scientists dismissed as nonsense. Then, one day in 2012, he was proven right. Canada's most influential thinker in the field of artificial intelligence is far too classy to say I told you so

https://torontolife.com/tech/ai-superstars-google-facebook-apple-studied-guy/

For more than 30 years, Geoffrey Hinton hovered at the edges of artificial intelligence research, an outsider clinging to a simple proposition: that computers could think like humans do—using intuition rather than rules.

Geoffrey Hinton 多年來堅持着一個簡單的觀點：電腦可以像人類一樣思考-用直覺而不是規則。Hinton 一直好奇的是，電腦能不能像人類大腦一樣的工作：信息通過一個巨大的，由神經元圖譜連接起來的細胞網絡傳播，在多達十億條的路徑上發射、連接和傳輸。

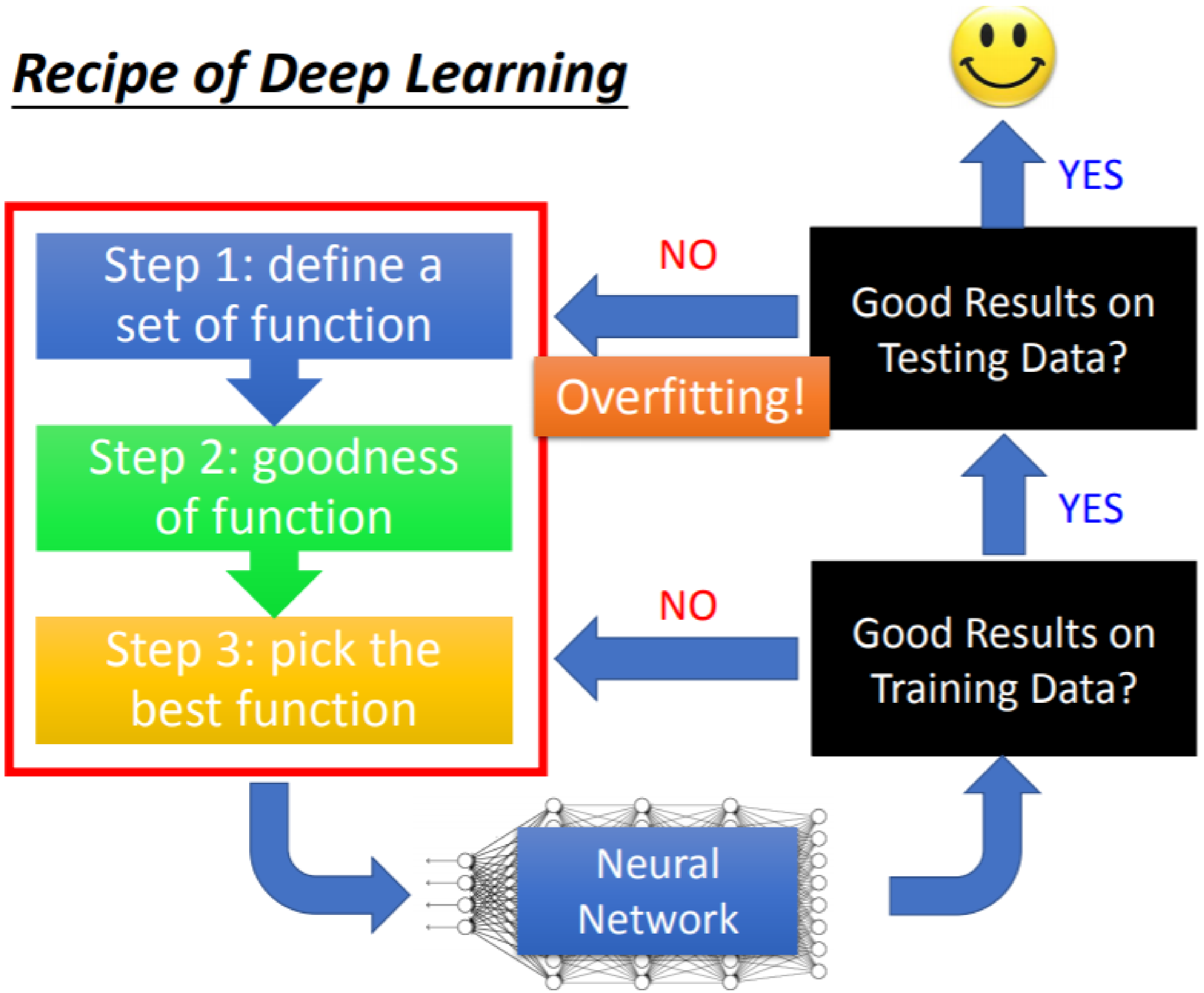
Practice – MLP regression



Practice – Read data from file

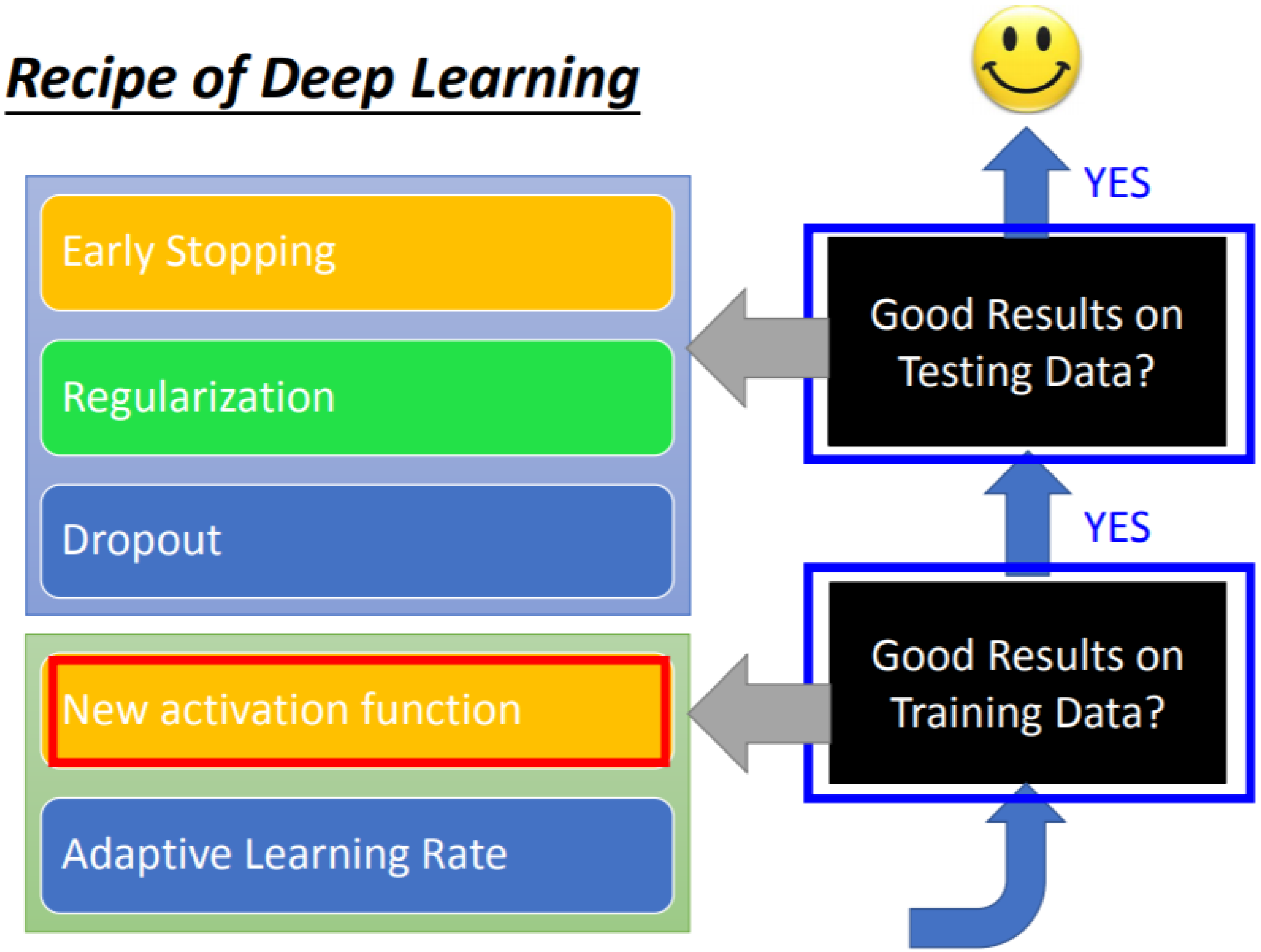
- Run "1. 2. MLP regression (Read Excel data file)"

Recipe of deep learning



Reference: 李弘毅 ML Lecture 9-1 https://youtu.be/xki61j7z-30

When training result is not good

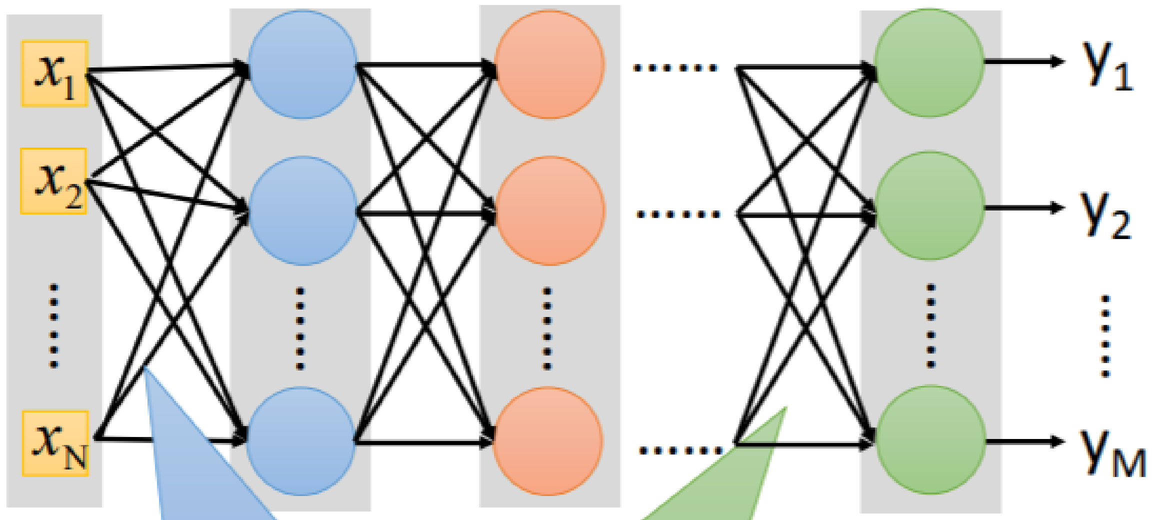


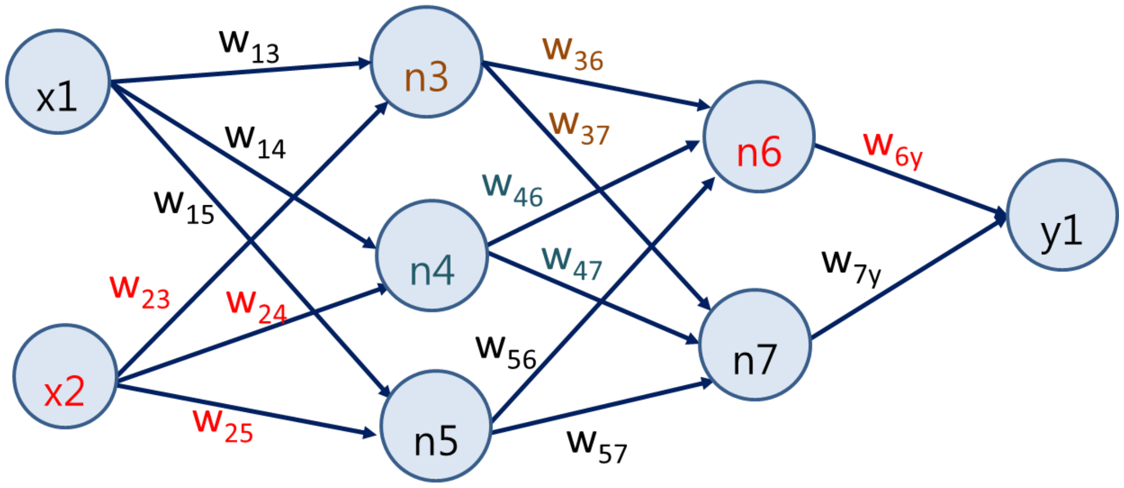
Reference: 李弘毅 ML Lecture 9-1 https://youtu.be/xki61j7z-30

Practice – Activation function

- Run "1.2 MLP regression.ipynb", change the activation function from ReLU to Sigmoid and explain why the results become worse?

Vanishing gradient problem





Smaller gradients

Learn very slow

Almost random

Larger gradients

Learn very fast

Already converge

based on random!?

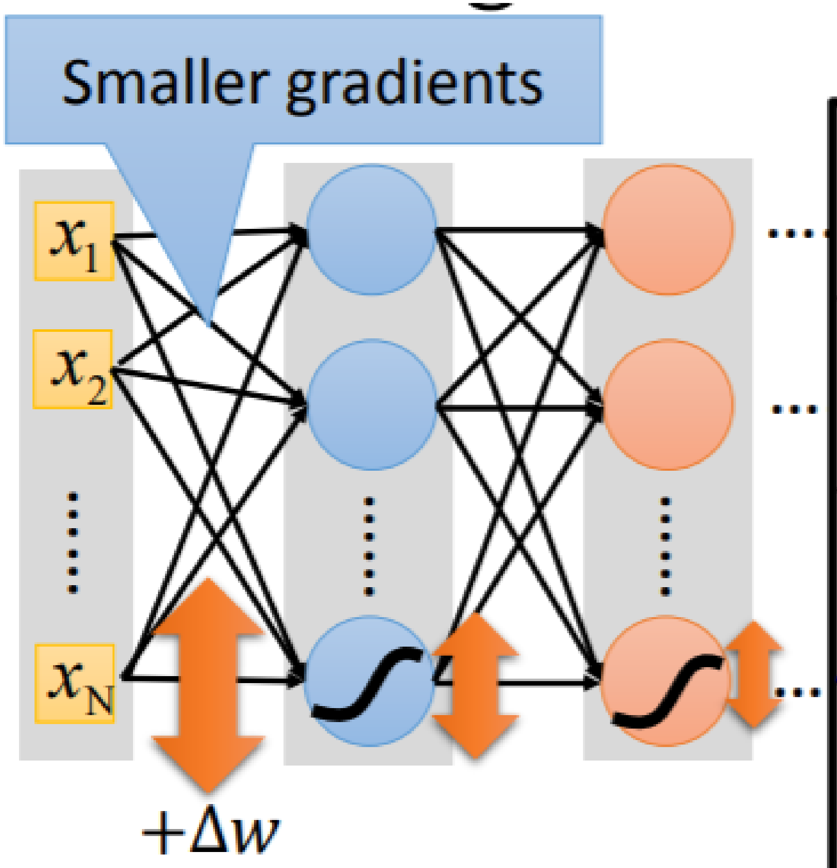
$$\frac{\partial L}{\partial w_{6y}} = \frac{\partial L}{\partial y_1}\frac{\partial y_1}{\partial w_{6y}}$$

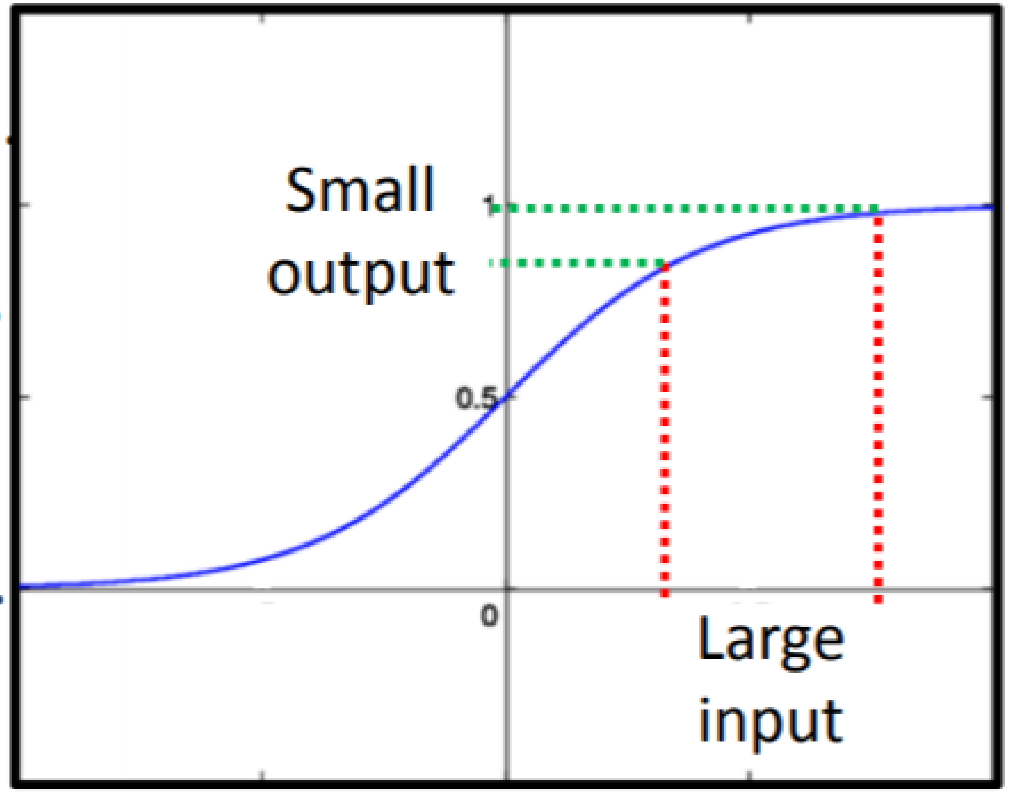
$$\frac{\partial L}{\partial w_{57}} = \frac{\partial L}{\partial y_1}\frac{\partial y_1}{\partial n_7}\frac{\partial n_7}{\partial w_{57}}$$

Reference: 李弘毅 ML Lecture 9-1 https://youtu.be/xki61j7z-30

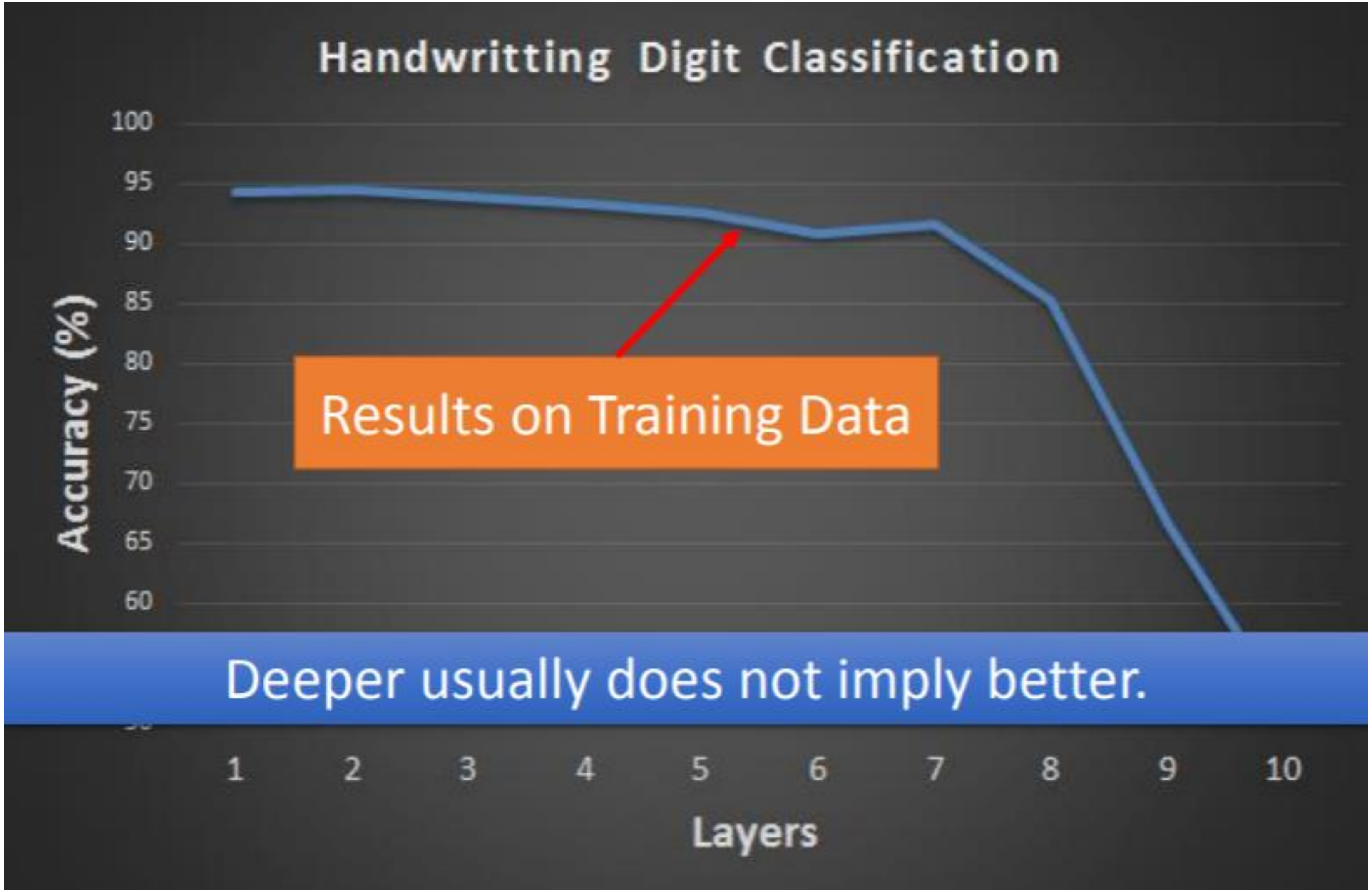
Vanishing gradient problem

$n_7 = \sigma(n_3 * w_{37} + n_4 * w_{47} + n_5 * w_{57} + b_7)$





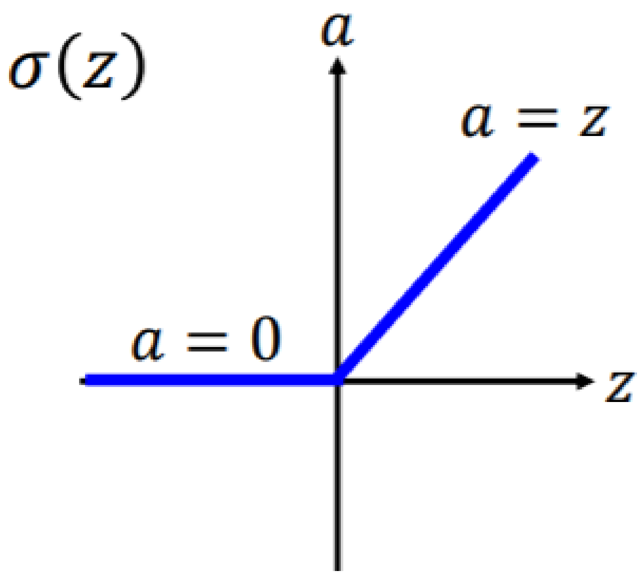
Sigmoid is hard to get the power of deep



Reference: 李弘毅 ML Lecture 9-1 https://youtu.be/xki61j7z-30

ReLU

- Rectified Linear Unit (ReLU)

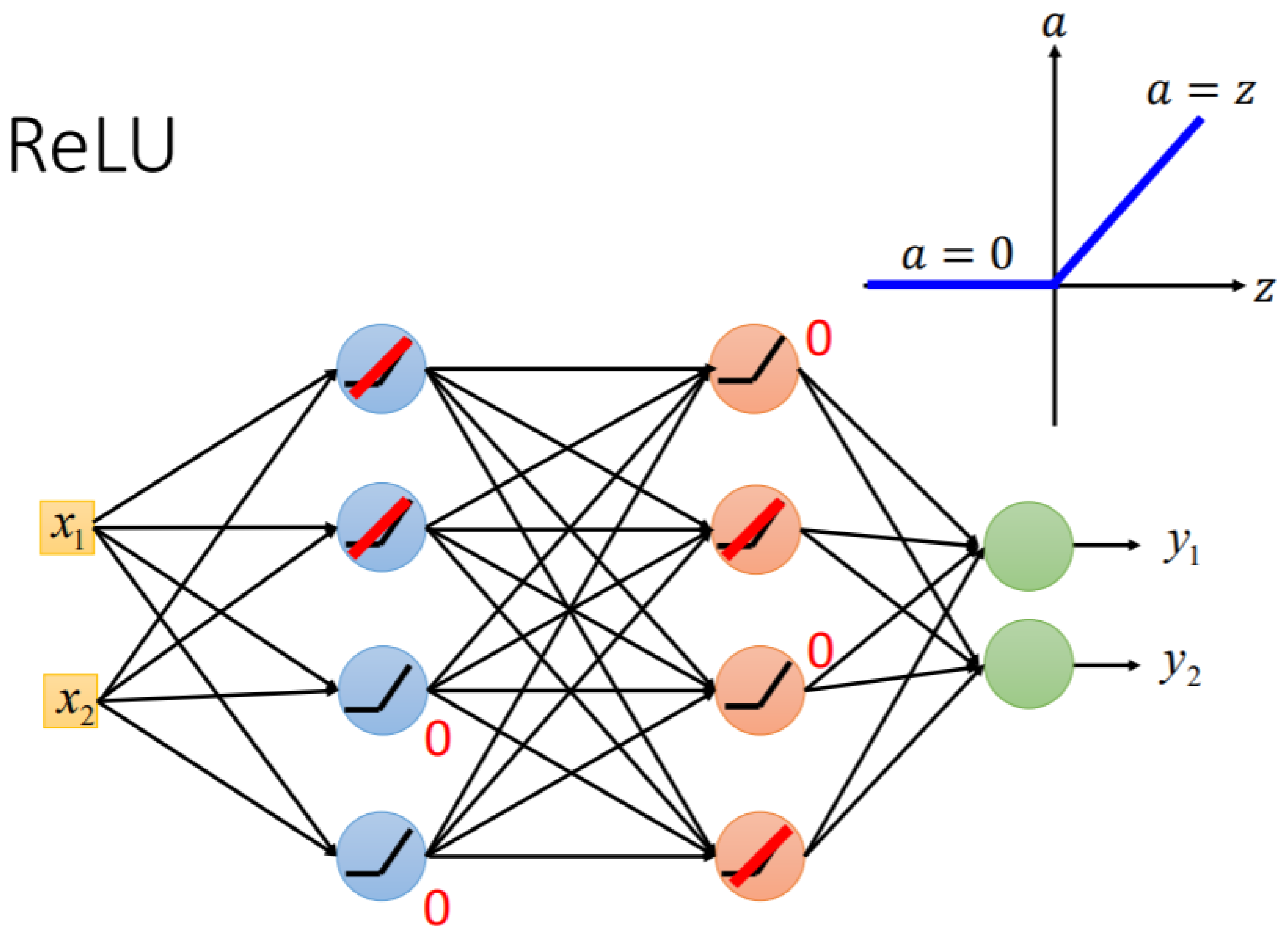


[Xavier Glorot, AISTATS'11]
[Andrew L. Maas, ICML'13]
[Kaiming He, arXiv'15]

Reason:

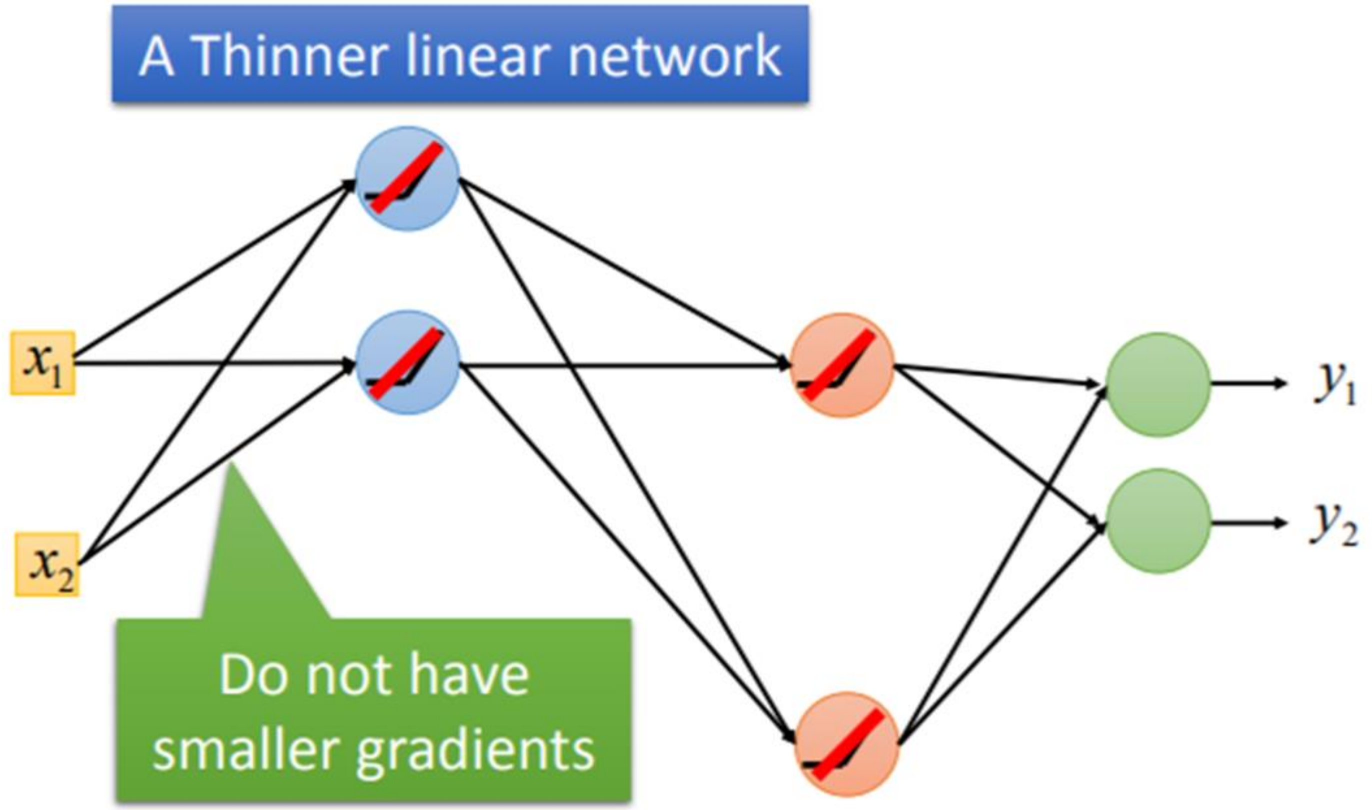
1. Fast to compute
2. Biological reason
3. Infinite sigmoid with different biases
4. Vanishing gradient problem

ReLU



Reference: 李弘毅 ML Lecture 9-1 https://youtu.be/xki61j7z-30

ReLU



Reference: 李弘毅 ML Lecture 9-1 https://youtu.be/xki61j7z-30

Practice – When training result is not good

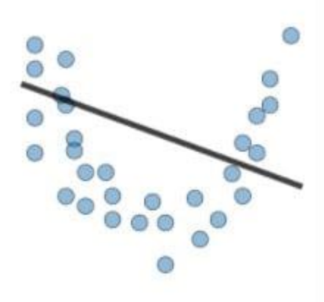
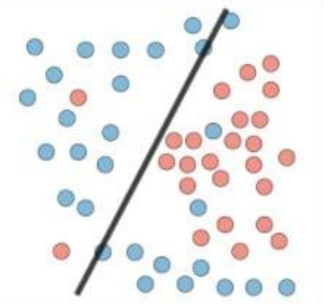
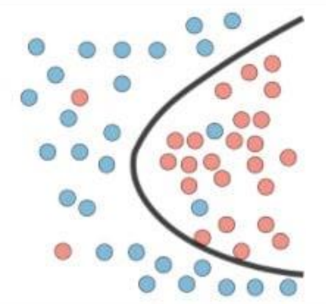
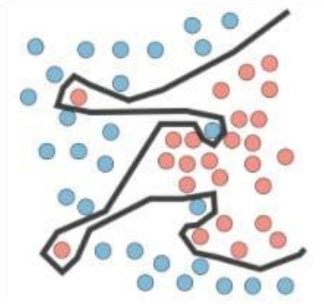
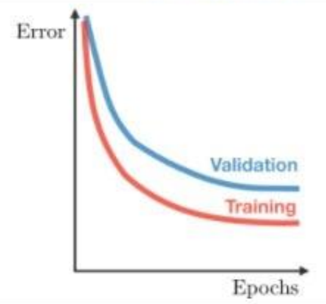
- Run "1. 2.1 MLP regression practice (1).ipynb".
- Run "1. 2.1 MLP regression practice (2).ipynb".

HW3 (1)

- Design your training data and $y = f(x)$. Experiment with different NN designs.

		Training
No of parameters =	Deep and thing	
	Shallow and fat	
	In-between version	
No of parameters =	Deep and thing	
	Shallow and fat	
	In-between version	

Overfitting problem – NN performs well on training data but not good on test data

	Underfitting	Just right	Overfitting
Symptoms	• High training error • Training error close to test error • High bias	• Training error slightly lower than test error	• Very low training error • Training error much lower than test error • High variance
Regression illustration			
Classification illustration			
Deep learning illustration			
Possible remedies	• Complexify model • Add more features • Train longer		• Perform regularization • Get more data

https://www.facebook.com/buggyprogrammers/photos/a.113230983855158/213140590530863/

Practice – Overfitting

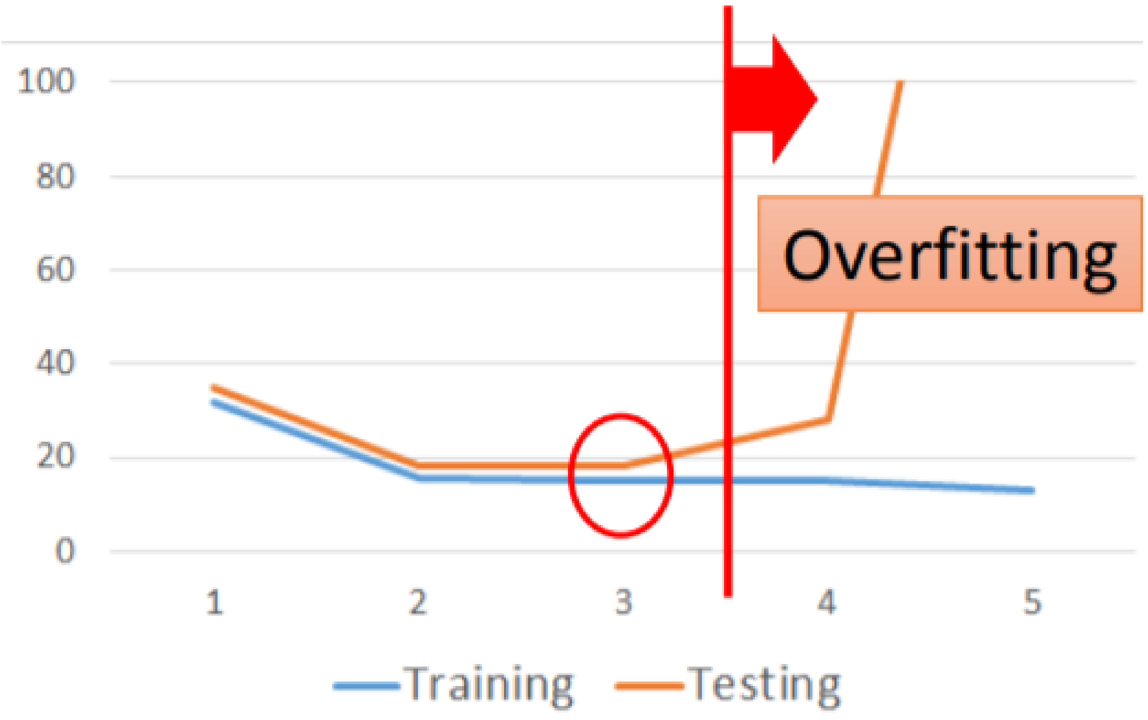
- Run "1. 2.2. Overfitting.ipynb", observe the overfitting problem.

HW3 (2)

- Extend HW3(1) to report overfitting problem.

		Training	Testing
No of parameters =	Deep and thing		
	Shallow and fat		
	In-between version		
No of parameters =	Deep and thing		
	Shallow and fat		
	In-between version		

Overfitting problem



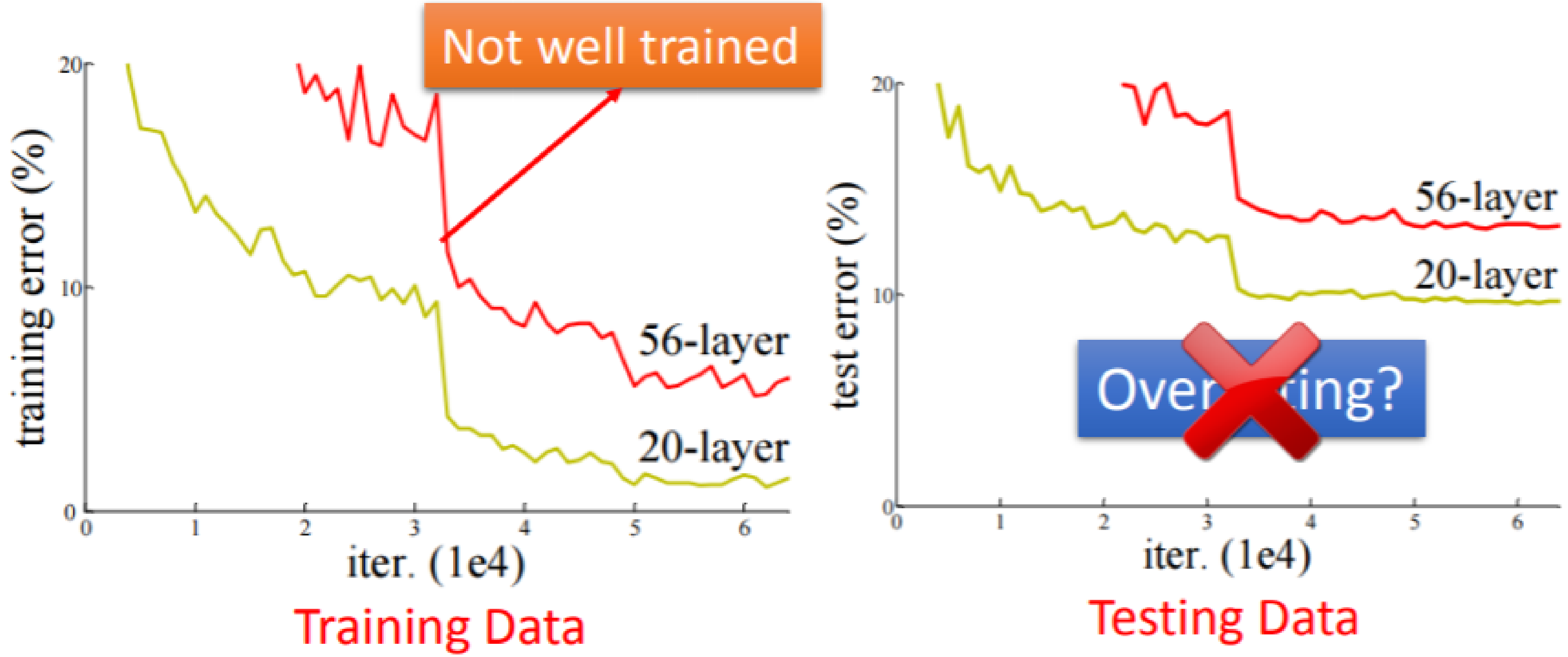
	Training	Testing
1	31.9	35.0
2	15.4	18.4
3	15.3	18.1
4	14.9	28.2
5	12.8	232.1

A more complex model does not always lead to better performance on ***testing data***.

This is ***Overfitting***. → Select suitable model

Reference: 李弘毅 ML Lecture 1 https://youtu.be/CXgbekl66jc

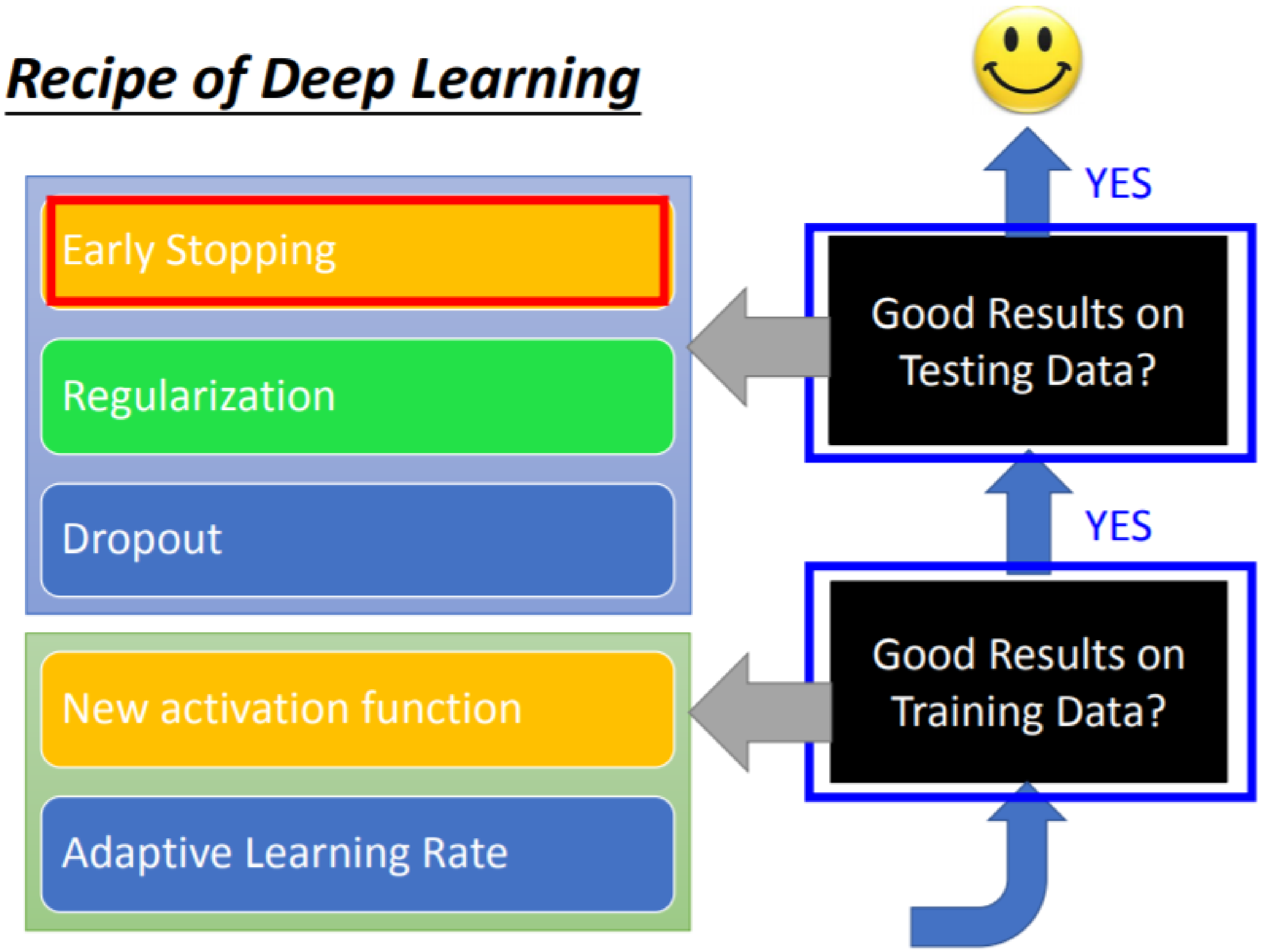
Do not always blame overfitting



Deep Residual Learning for Image Recognition
http://arxiv.org/abs/1512.03385

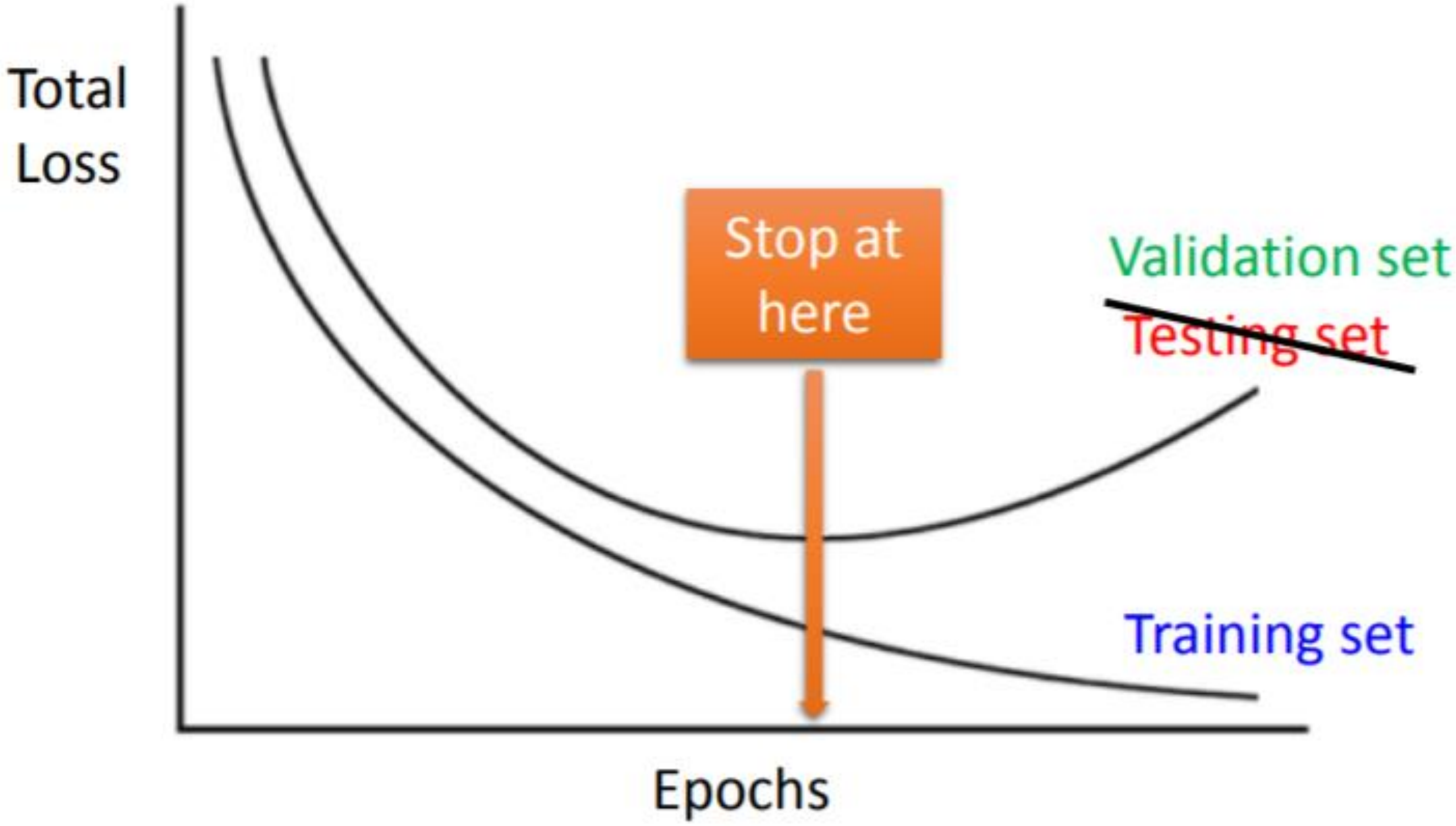
Reference: 李弘毅 ML Lecture 9-1 https://youtu.be/xki61j7z-30

What to do if overfitting?



Reference: 李弘毅 ML Lecture 9-1 https://youtu.be/xki61j7z-30

Early stopping



Keras: http://keras.io/getting-started/faq/#how-can-i-interrupt-training-when-the-validation-loss-isnt-decreasing-anymore

Reference: 李弘毅 ML Lecture 9-1 https://youtu.be/xki61j7z-30

Practice – Early stop

- Run "1. 2.3. Early stop.ipynb", observe how the overfitting problem is reduced.

HW3 (3)

- Based on HW3(2), experiment the effect of early stop on resolving overfitting.

		Training with early stop	Testing
No of parameters =	Deep and thing		
	Shallow and fat		
	In-between version		
No of parameters =	Deep and thing		
	Shallow and fat		
	In-between version		

Regularization – L2

- Find a set of weight not only minimizing original cost but also close to zero

$$L'(\theta) = \underline{L(\theta)} + \lambda \frac{1}{2} \underline{\|\theta\|_2}$$ → Regularization term

$$\theta = \{w_1, w_2, \ldots\}$$

Original loss

$$L(\theta) = \sum_{n=1}^{N} (\hat{y}^n - y^n)^2$$

L2 regularization:

$$\|\theta\|_2 = (w_1)^2 + (w_2)^2 + \ldots$$

(usually not consider biases)

Reference: 李弘毅 ML Lecture 9-1 https://youtu.be/xki61j7z-30

L2 regularization

$$L'(\theta) = L(\theta) + \lambda \frac{1}{2} \|\theta\|_2 \quad \text{Gradient:} \quad \frac{\partial L'}{\partial w} = \frac{\partial L}{\partial w} + \lambda w$$

$$\text{Update:} \quad w^{t+1} \rightarrow w^t - \eta \frac{\partial L'}{\partial w} = w^t - \eta \left(\frac{\partial L}{\partial w} + \lambda w^t \right)$$

$$= \underbrace{(1 - \eta\lambda) w^t}_{\text{Closer to zero}} - \eta \underline{\frac{\partial L}{\partial w}}$$

Weight Decay

Vanilla gradient decent update $w^{t+1} \rightarrow w^t - \eta \frac{\partial L}{\partial w}$

Reference: 李弘毅 ML Lecture 9-1 https://youtu.be/xki61j7z-30

Practice – L2 regularization

- Run "1. 2.3. L2_Regularization.ipynb", observe the overfitting problem.

HW3 (4)

- Based on HW3(2), experiment the effect of L2 regularization on resolving overfitting.

		Train with L2 regularization	Testing
No of parameters =	Deep and thing		
	Shallow and fat		
	In-between version		
No of parameters =	Deep and thing		
	Shallow and fat		
	In-between version		

L1 regularization

L1 regularization:

$$\|\theta\|_1 = |w_1| + |w_2| + \ldots$$

$$\mathrm{L}'(\theta) = L(\theta) + \lambda \frac{1}{2}\|\theta\|_1 \qquad \frac{\partial \mathrm{L}'}{\partial w} = \frac{\partial \mathrm{L}}{\partial w} + \lambda\,\mathrm{sgn}(w)$$

Update:

$$w^{t+1} \to w^t - \eta \frac{\partial \mathrm{L}'}{\partial w} = w^t - \eta\left(\frac{\partial \mathrm{L}}{\partial w} + \lambda\,\mathrm{sgn}(w^t)\right)$$

$$= w^t - \eta \frac{\partial \mathrm{L}}{\partial w} - \underline{\eta\lambda\,\mathrm{sgn}(w^t)} \quad \text{Always delete}$$

$$= (1 - \eta\lambda) w^t - \eta \frac{\partial \mathrm{L}}{\partial w} \quad \text{...... L2}$$

Practice – L1 regularization

- Run "1. 2.3. L1_Regularization.ipynb", observe the overfitting problem.

HW3 (5)

- Based on HW3(2), experiment the effect of L1 regularization on resolving overfitting.

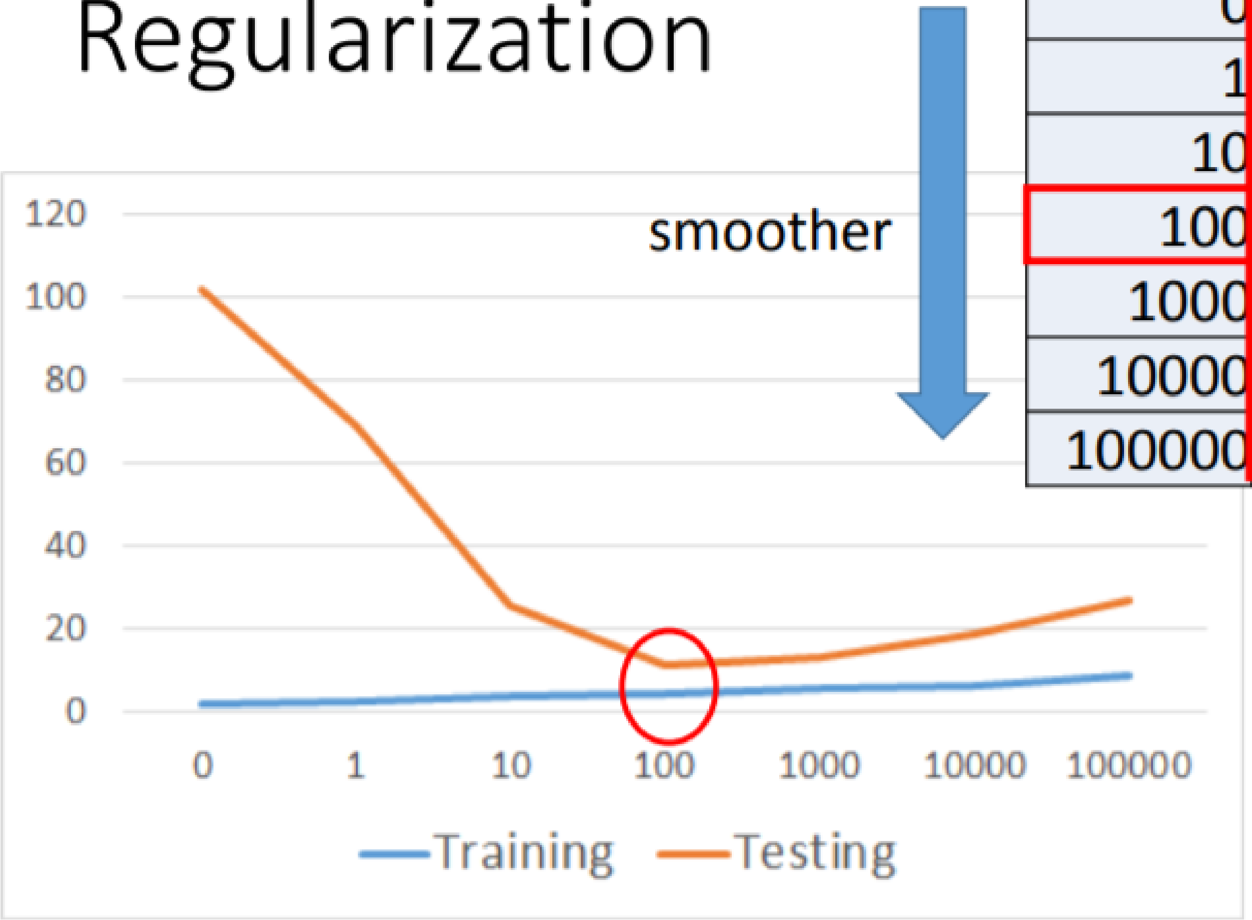
		Train with L1 regularization	Testing
No of parameters =	Deep and thing		
	Shallow and fat		
	In-between version		
No of parameters =	Deep and thing		
	Shallow and fat		
	In-between version		

Practice – Small initial weights

- Run "1. 2.3. Initialize_small_weights.ipynb". By initializing NN weight using small values we can also do regularization.

Regularization

Regularization



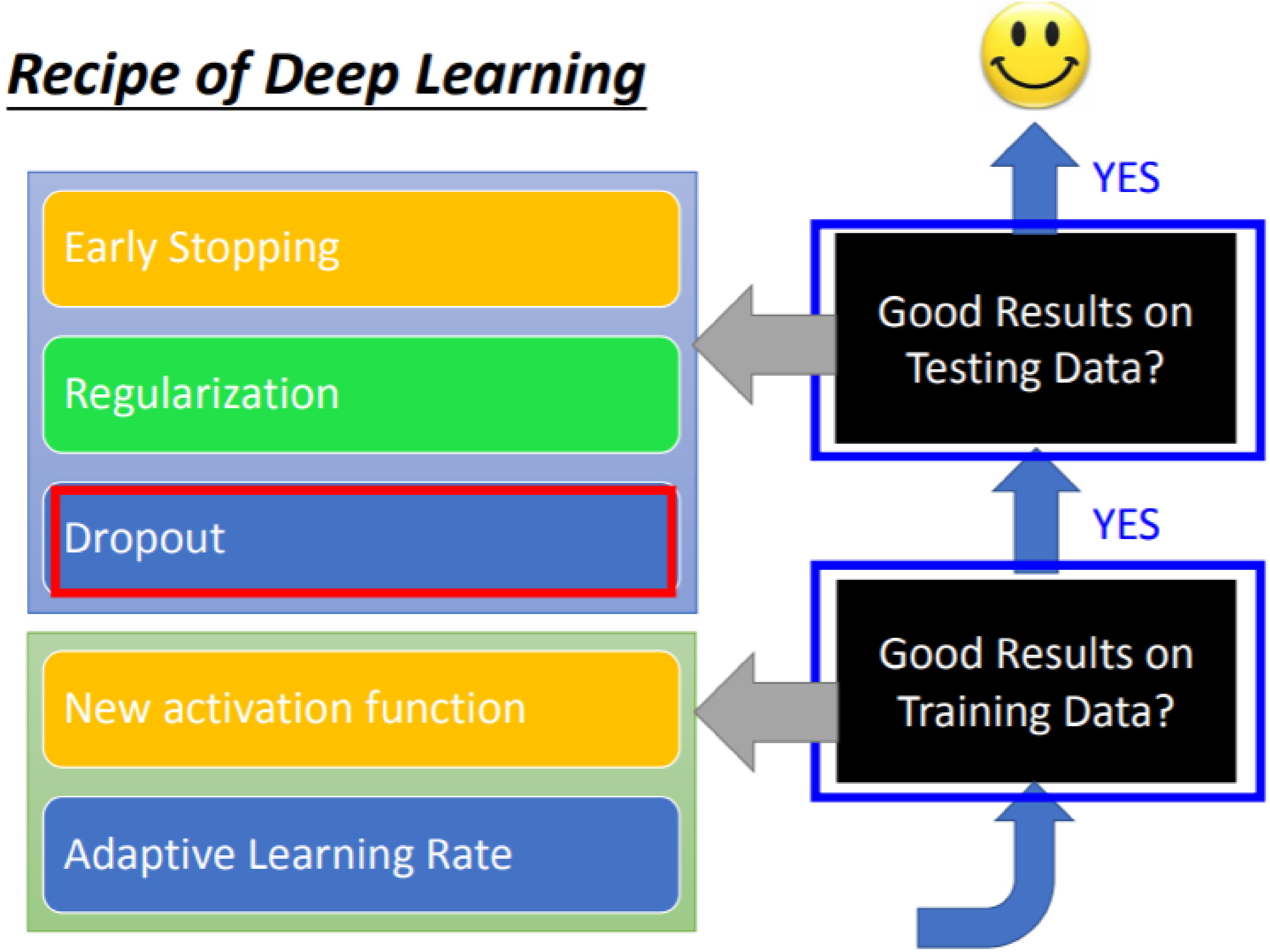
λ	Training	Testing
0	1.9	102.3
1	2.3	68.7
10	3.5	25.7
100	4.1	11.1
1000	5.6	12.8
10000	6.3	18.7
100000	8.5	26.8

How smooth?

Select λ obtaining the best model

Reference: 李弘毅 ML Lecture 1 https://youtu.be/CXgbekl66jc

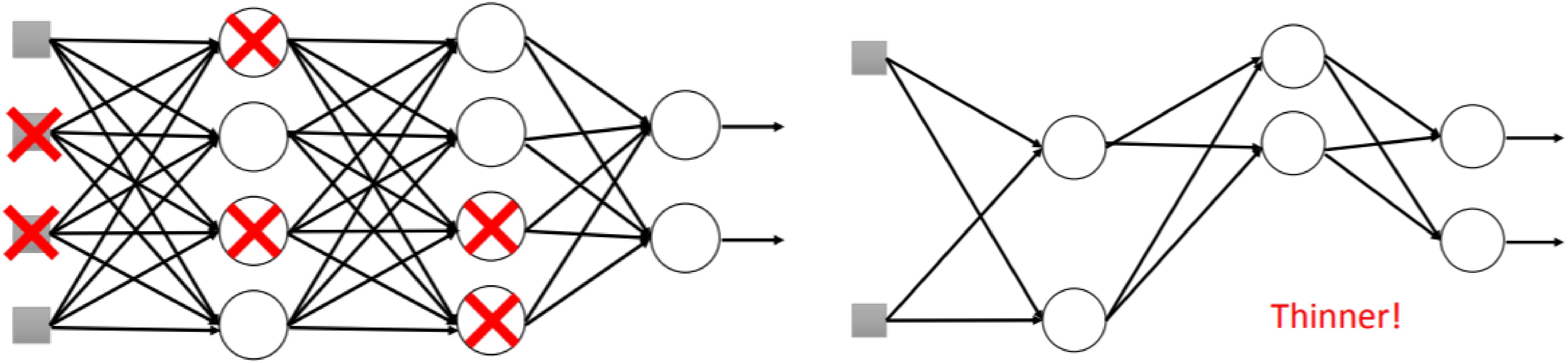
Drop out



Reference: 李弘毅 ML Lecture 9-1 https://youtu.be/xki61j7z-30

Drop out

Each time before updating θ, each neuron has $p\%$ to dropout. So the *NN* structure is changed (become thinner). That is, for each mini-batch, we resample the dropout neurons.

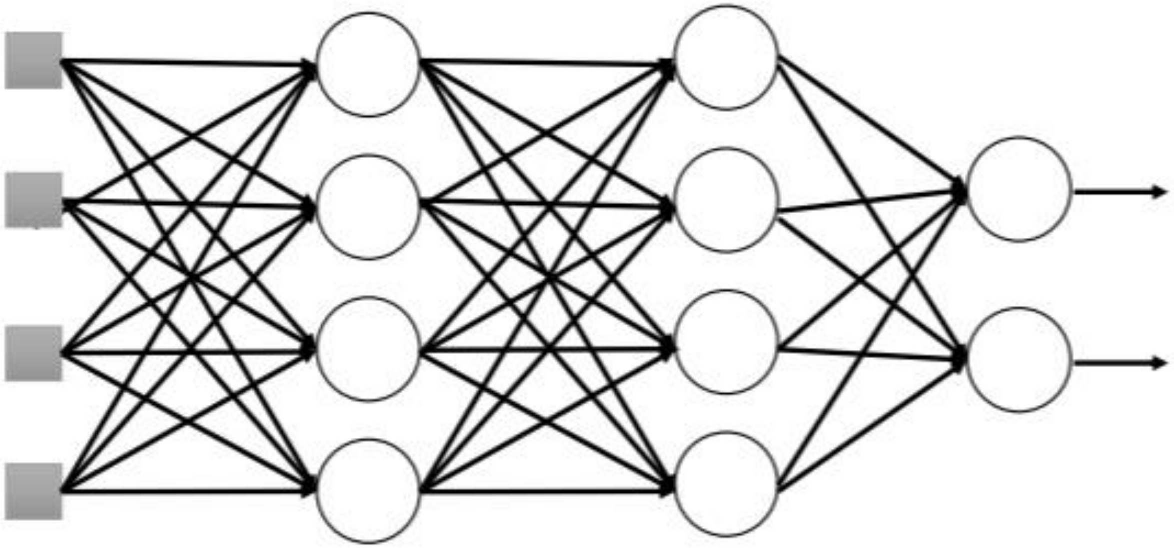


Reference: 李弘毅 ML Lecture 9-1 https://youtu.be/xki61j7z-30

Drop out

No neuron drop out at test stage. All weights time $1 - p\%$

Testing:



Reference: 李弘毅 ML Lecture 9-1 https://youtu.be/xki61j7z-30

Practice – Drop out

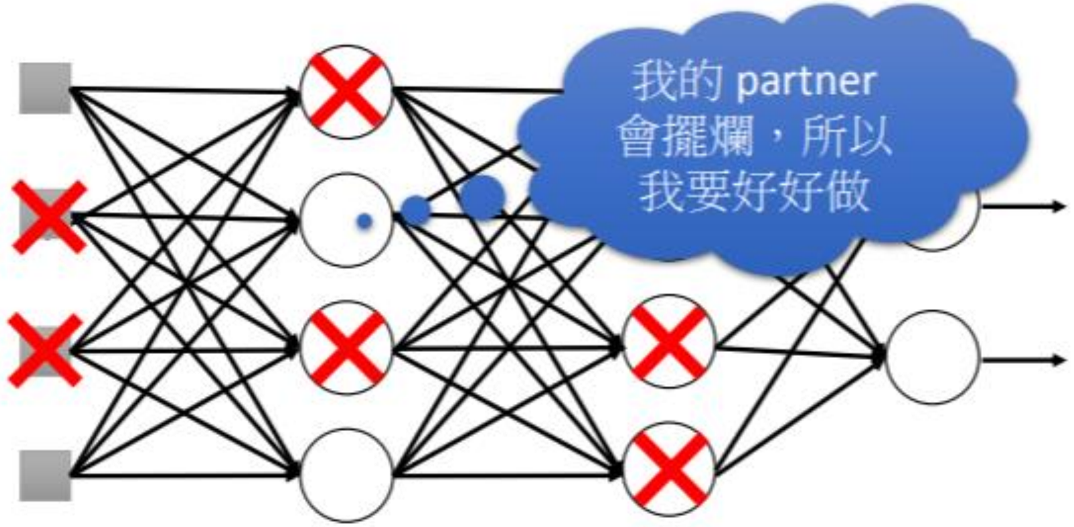
- Run “1. 2.3. Dropout.ipynb” .

HW3 (6)

- Based on HW3(2), experiment the effect of drop out on resolving overfitting.

		Train using drop out	Testing
No of parameters =	Deep and thing		
	Shallow and fat		
	In-between version		
No of parameters =	Deep and thing		
	Shallow and fat		
	In-between version		

Why drop out makes NN perform better?

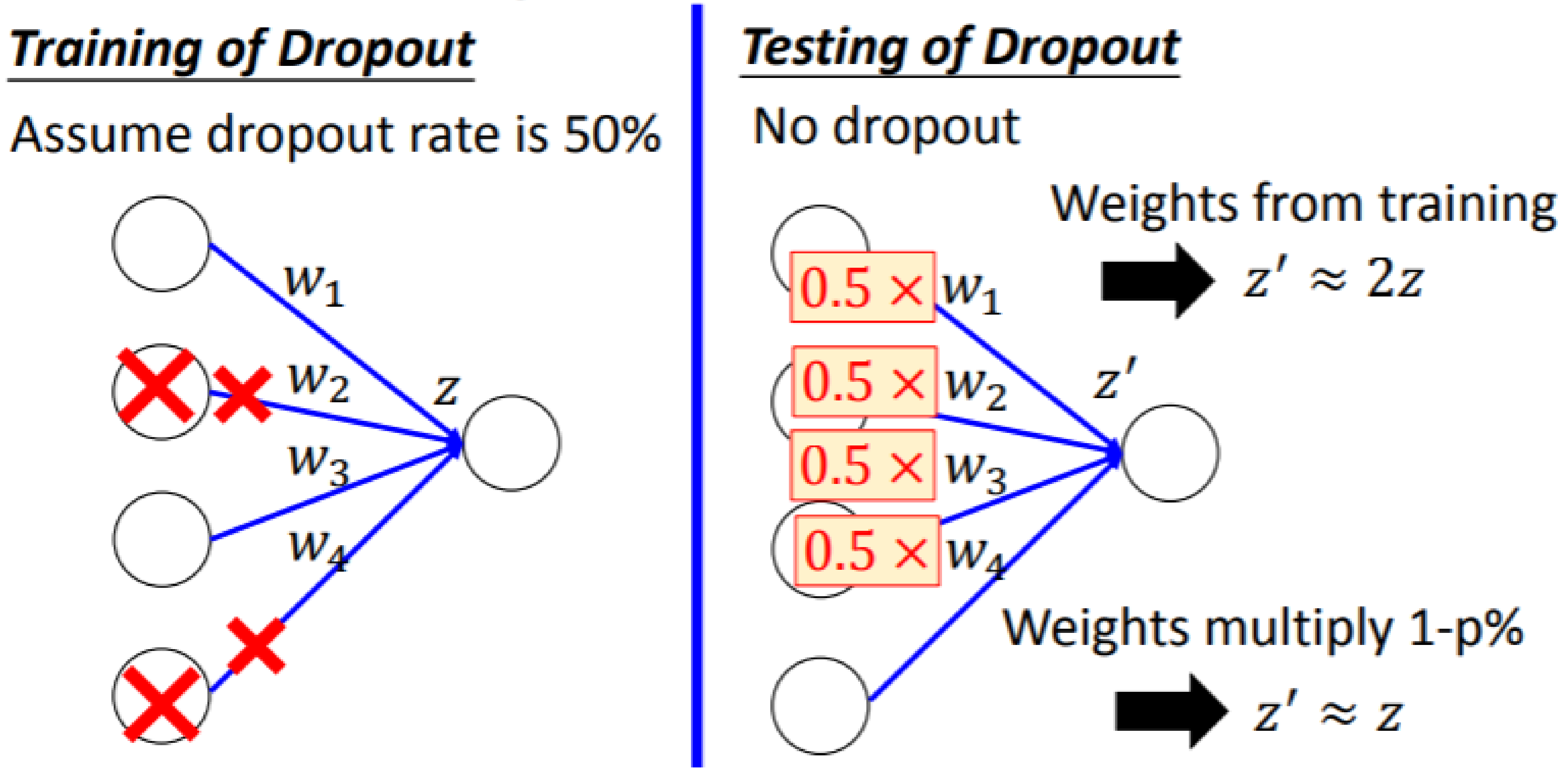


- When teams up, if everyone expect the partner will do the work, nothing will be done finally.
- However, if you know your partner will dropout, you will do better.
- When testing, no one dropout actually, so obtaining good results eventually.

Reference: 李弘毅 ML Lecture 9-1 https://youtu.be/xki61j7z-30

Why multiply weights by (1-p)% during testing?

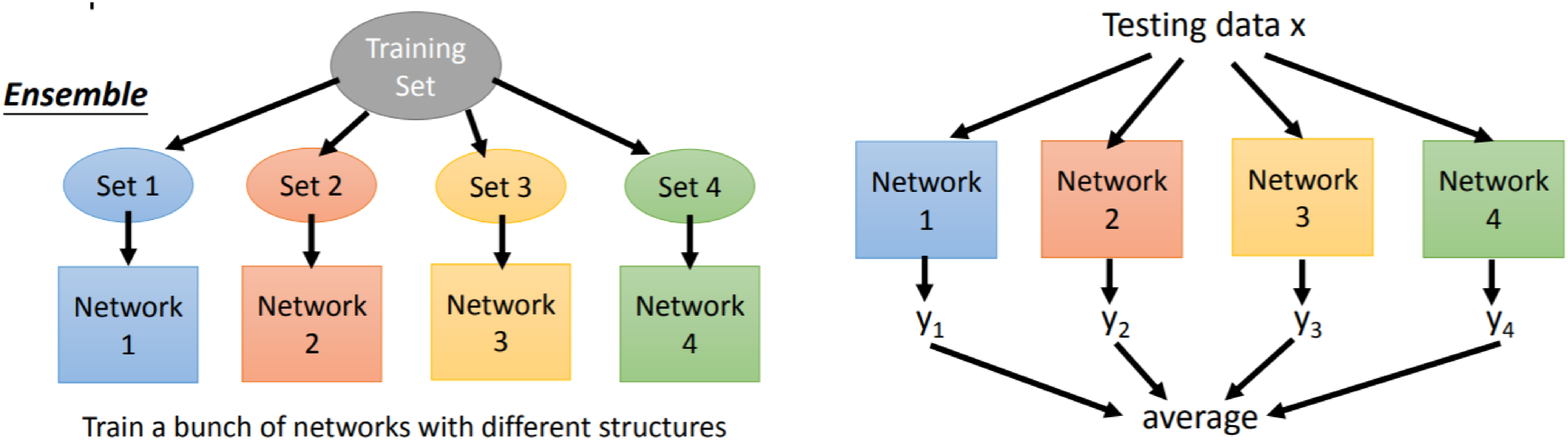
- Why the weights should multiply (1-p)% (dropout rate) when testing?



Reference: 李弘毅 ML Lecture 9-1 https://youtu.be/xki61j7z-30

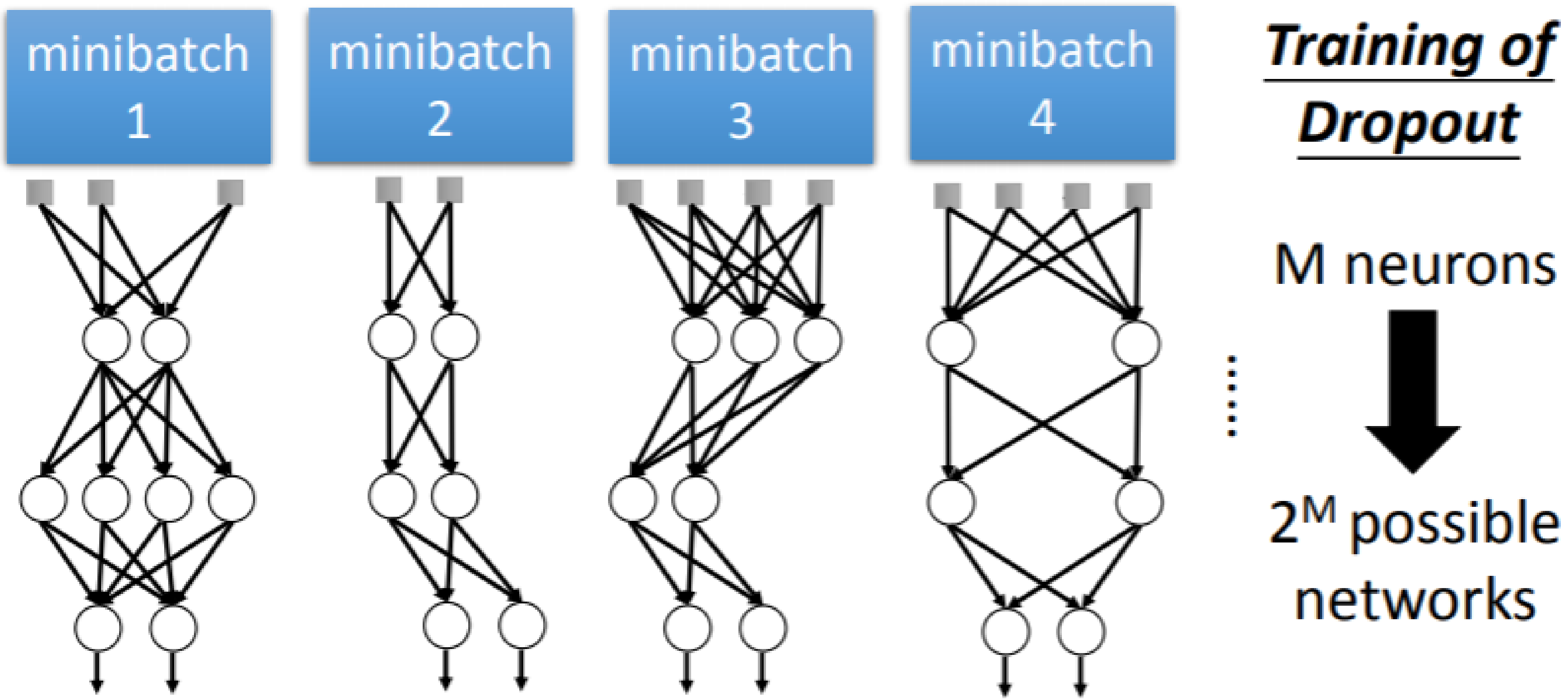
Why drop out makes NN performs better?

Drop out can be seen as an ensemble method.



Reference: 李弘毅 ML Lecture 9-1 https://youtu.be/xki61j7z-30

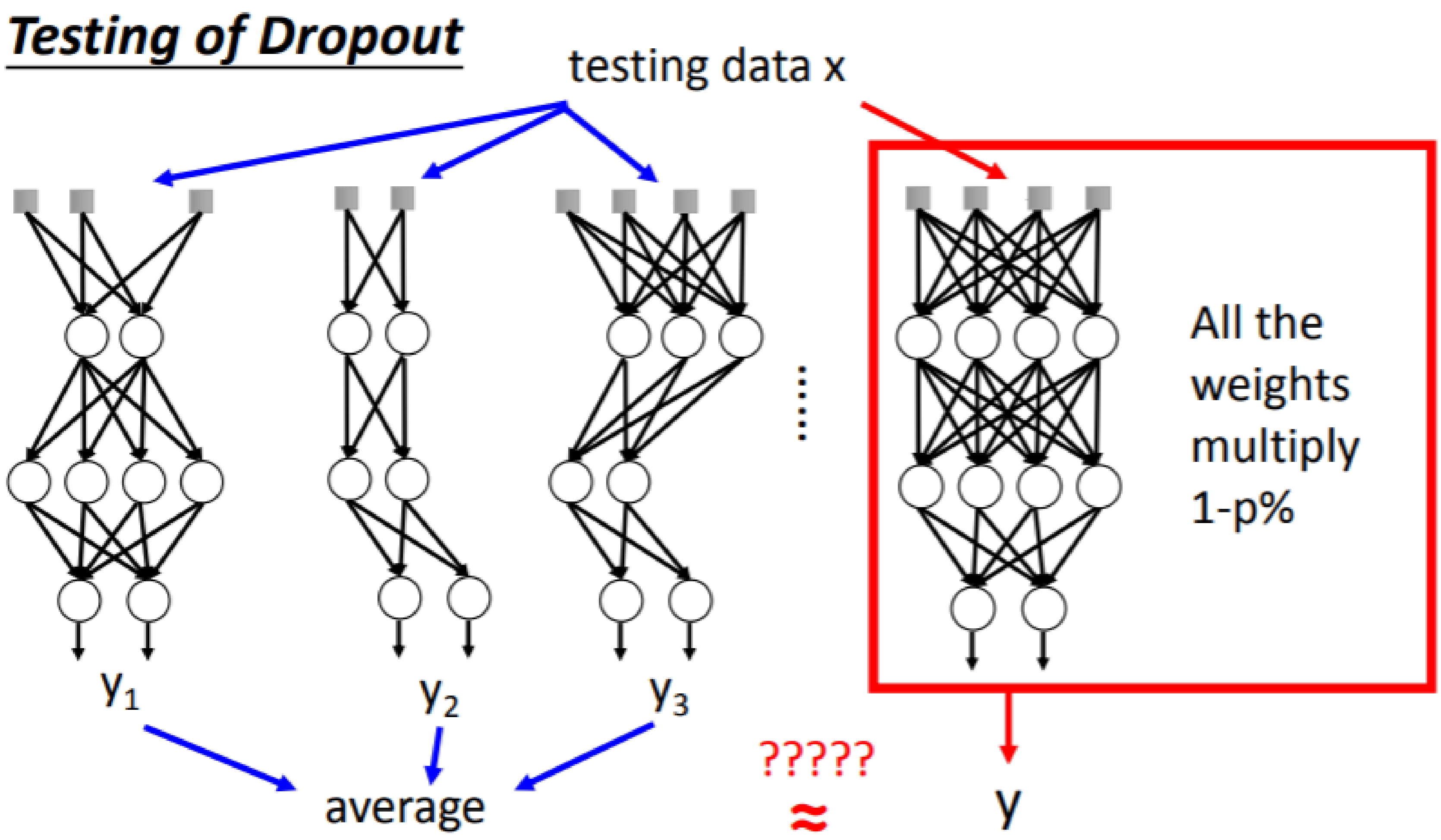
Why drop out makes NN performs better?



- ➢Using one mini-batch to train one network
- ➢Some parameters in the network are shared

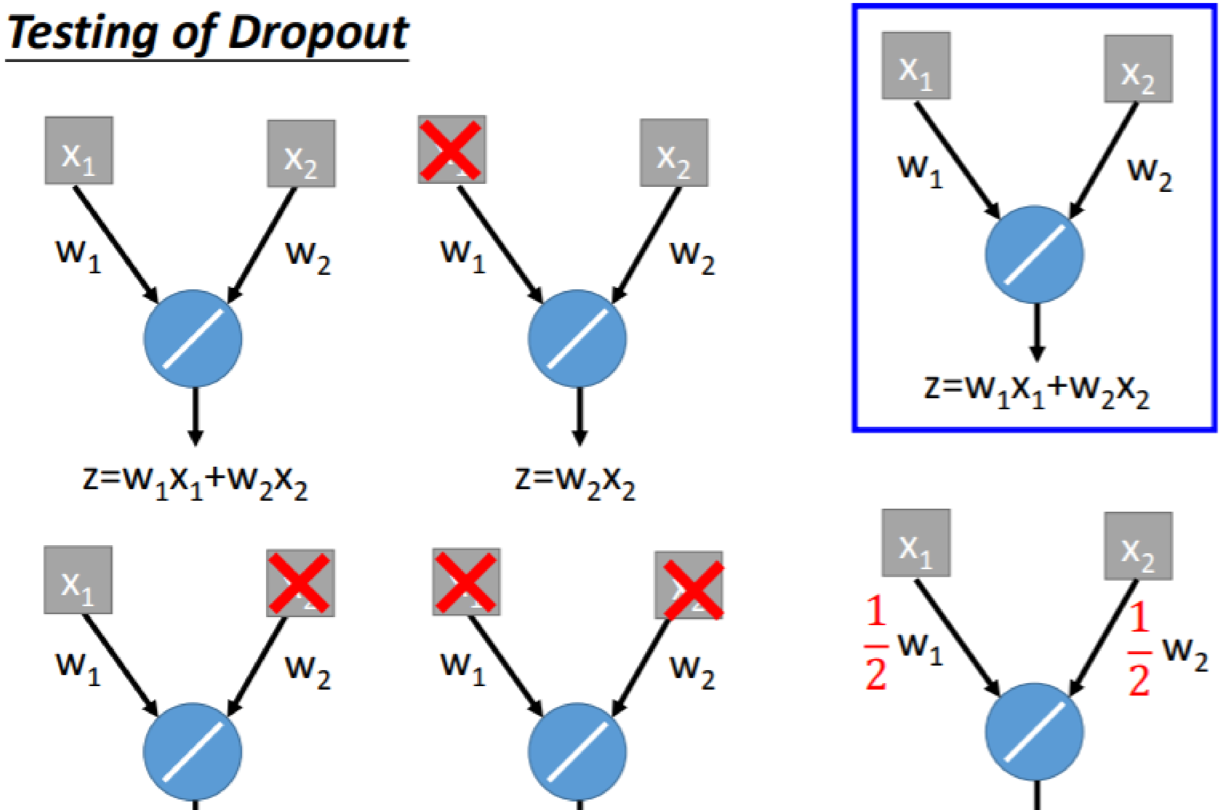
Reference: 李弘毅 ML Lecture 9-1 https://youtu.be/xki61j7z-30

Why multiply weights by (1-p)% during testing?



Reference: 李弘毅 ML Lecture 9-1 https://youtu.be/xki61j7z-30

Why multiply weights by (1-p)% during testing?



$$z = \frac{1}{2}w_1x_1 + \frac{1}{2}w_2x_2$$

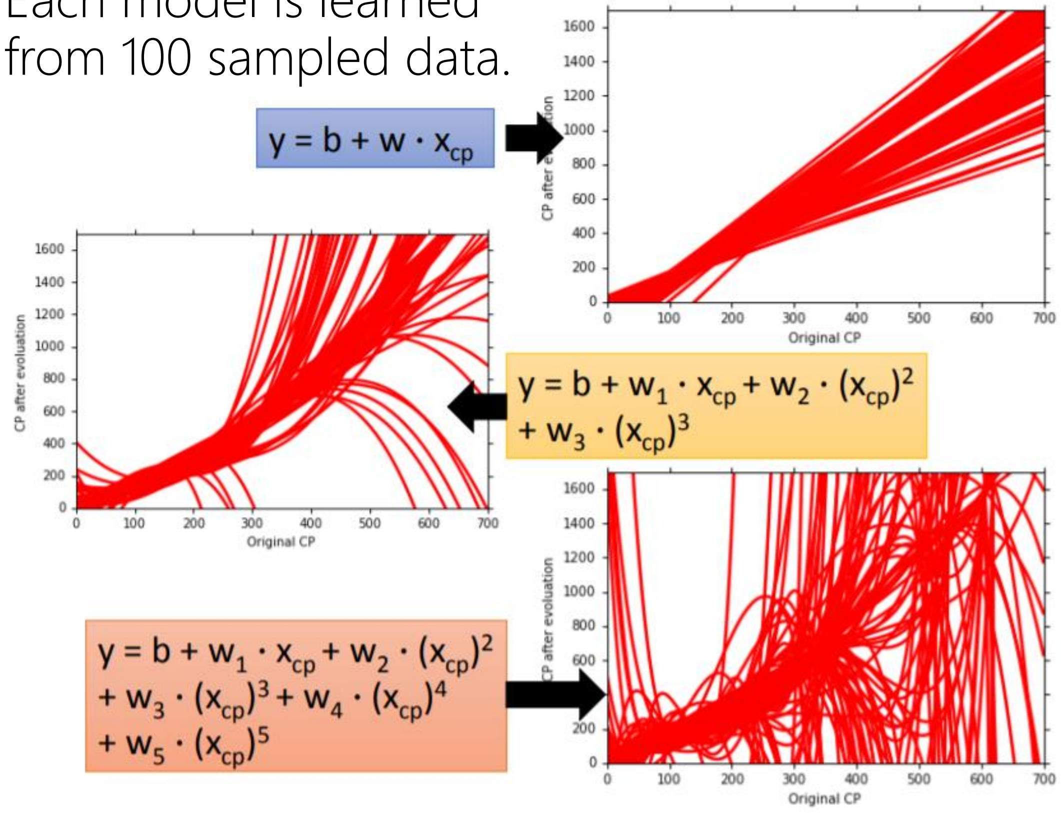
Reference: 李弘毅 ML Lecture 9-1 https://youtu.be/xki61j7z-30

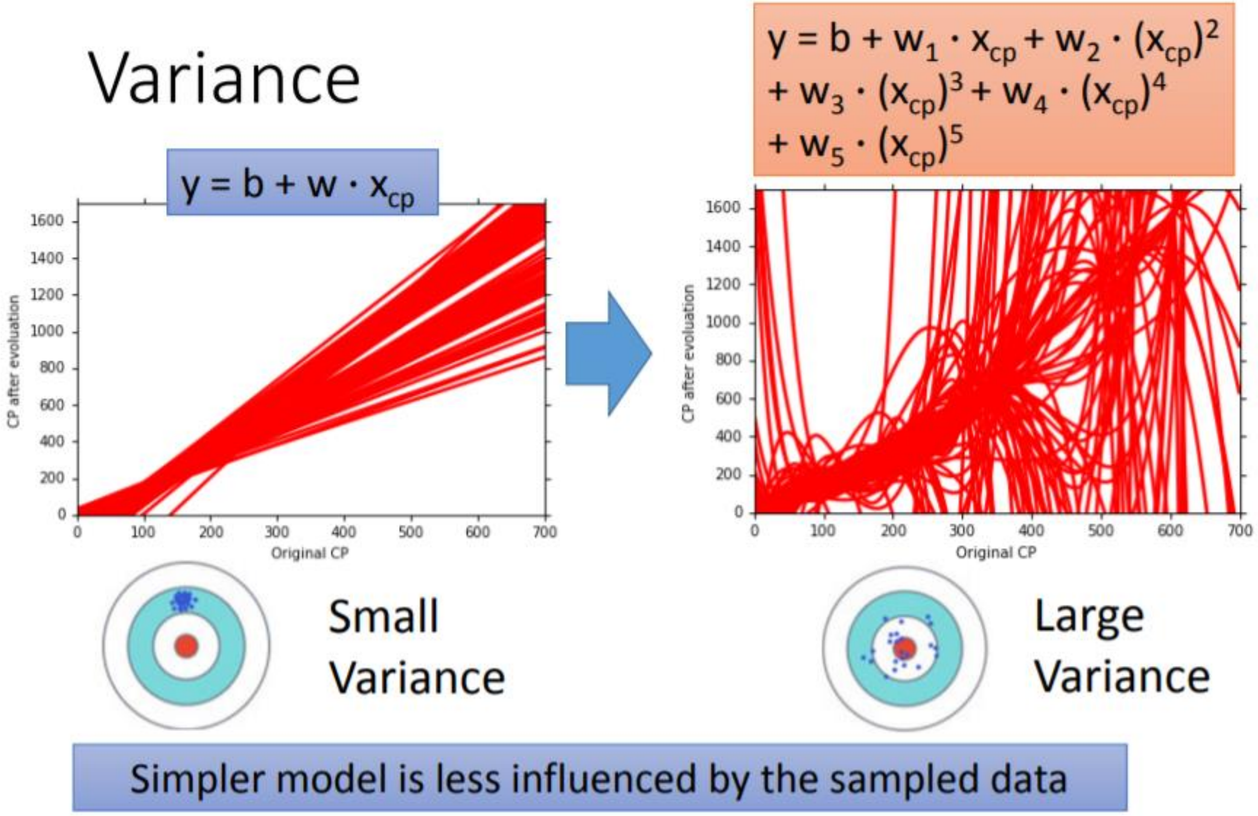
Report model's performance using bias and variance

Model's variance

Simpler model is less influenced by the sampled data and has smaller variance.

Each model is learned from 100 sampled data.





Reference: 李弘毅 ML Lecture 2 https://youtu.be/D_S6y0Jm6dQ

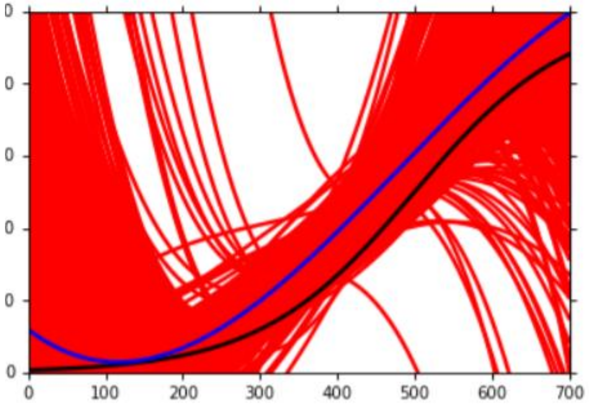
Bias and variance of ML models

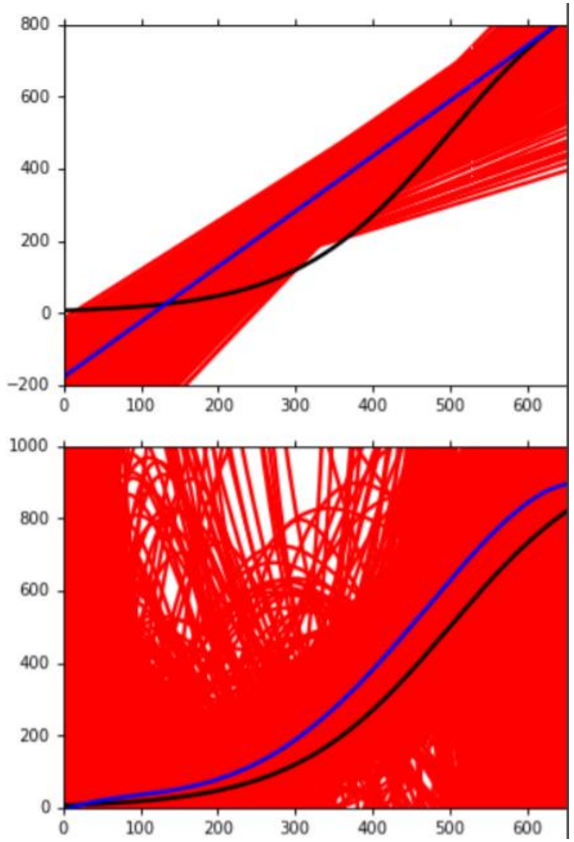
Simpler model has larger bias.

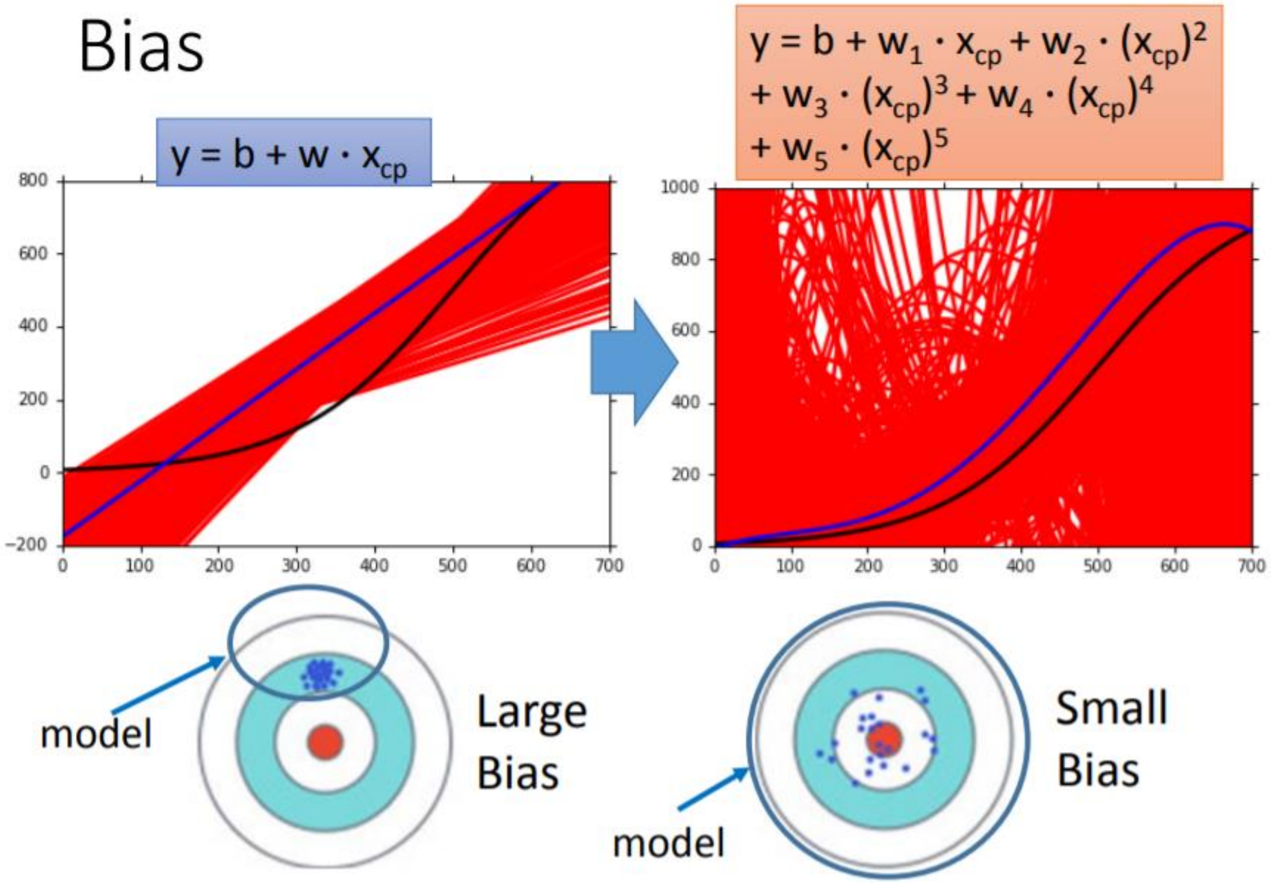
Black curve: the true function $\hat{f}$

Red curves: 5000 f^*

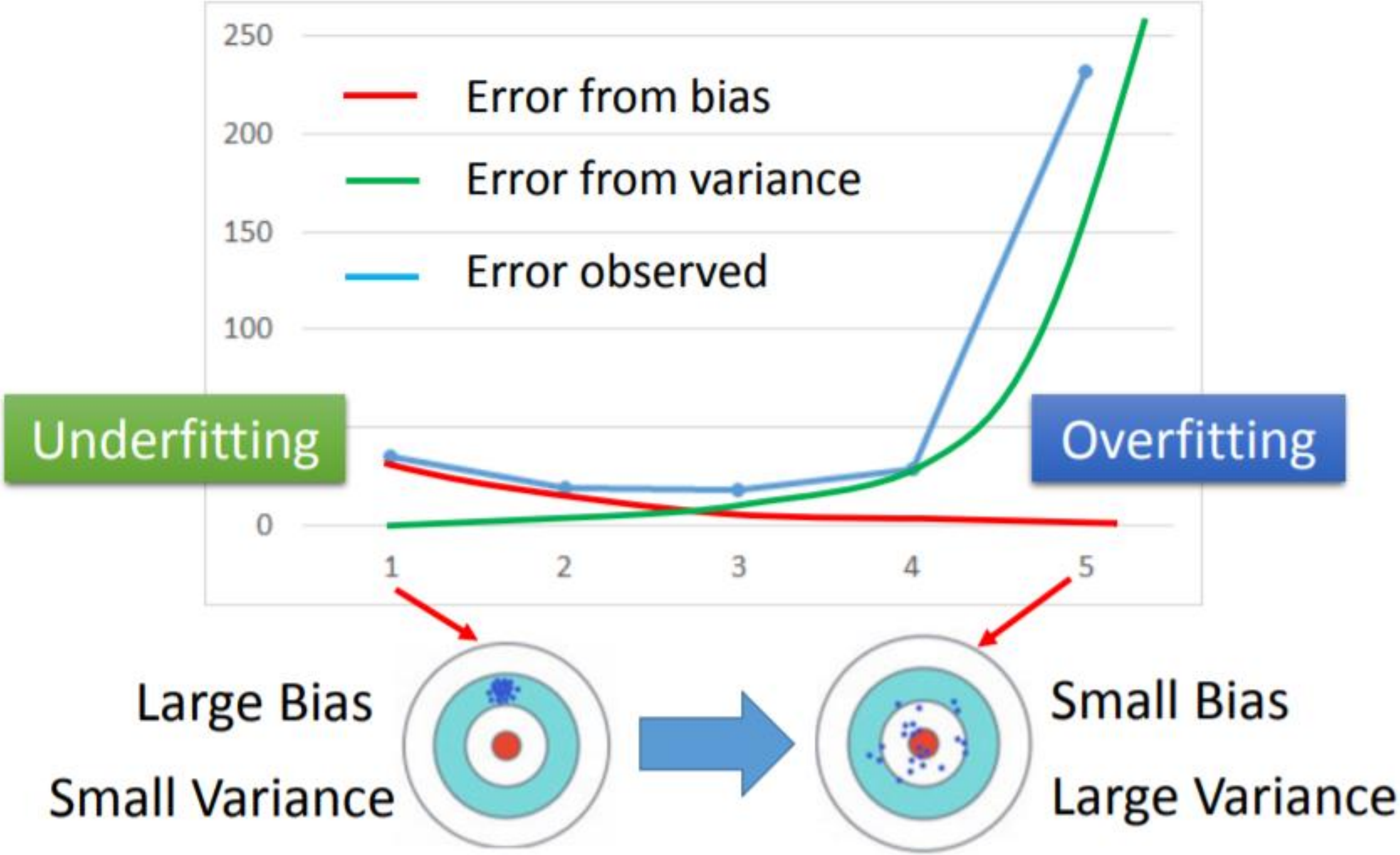
Blue curve: the average of 5000 f^* $= \bar{f}$







Errors of ML model



Reference: 李弘毅 ML Lecture 2 https://youtu.be/D_S6y0Jm6dQ

Practice – Variance of model prediction errors

- Run "1. 2.4. Variance of predicting error.ipynb" .

HW3 (7)

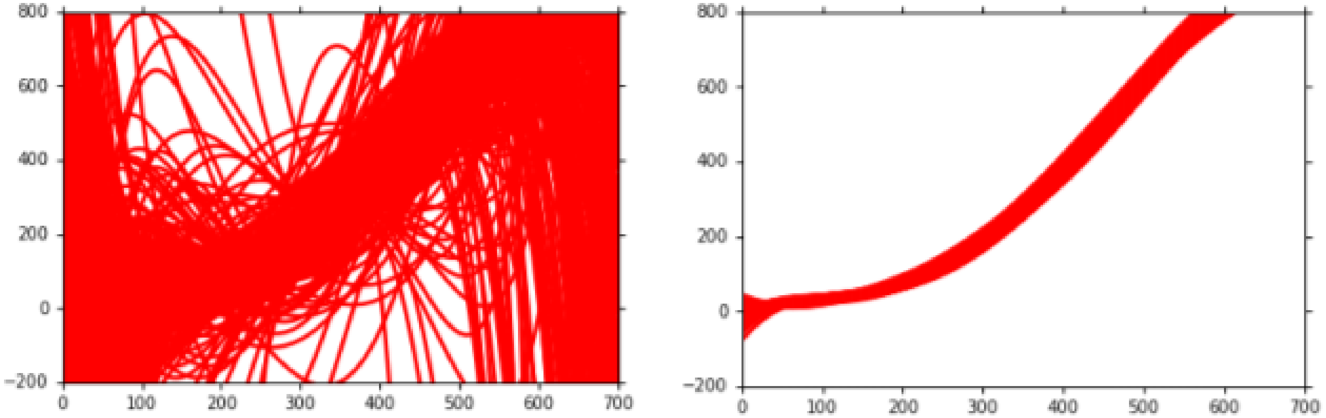
- Based on HW3(2), show the box plot and discuss the bias and variance of the NN models.

		Box plot
No of parameters =	Deep and thing	
	Shallow and fat	
	In-between version	
No of parameters =	Deep and thing	
	Shallow and fat	
	In-between version	

How to reduce model's variances?

① More data

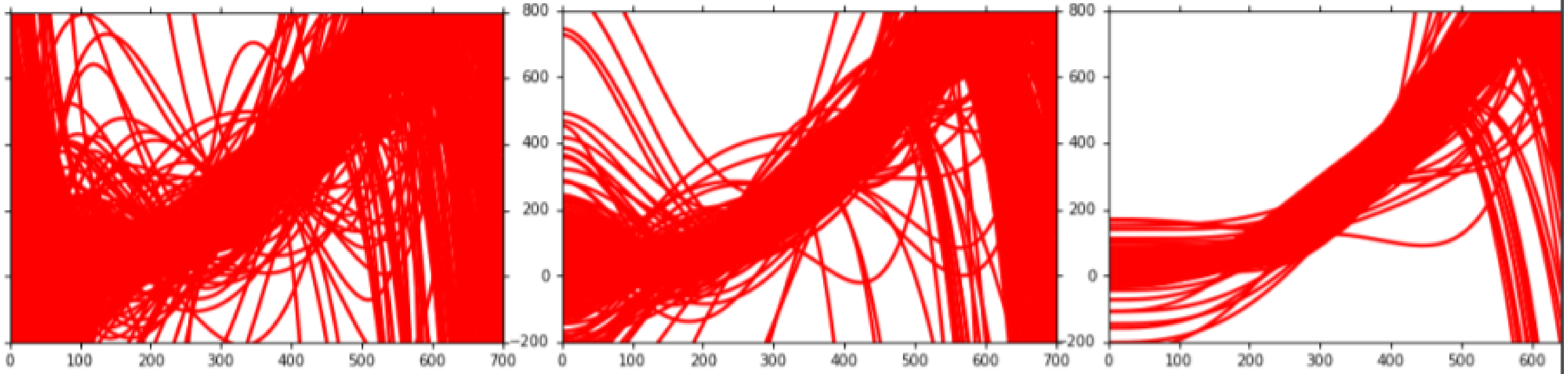
Very effective, but not always practical



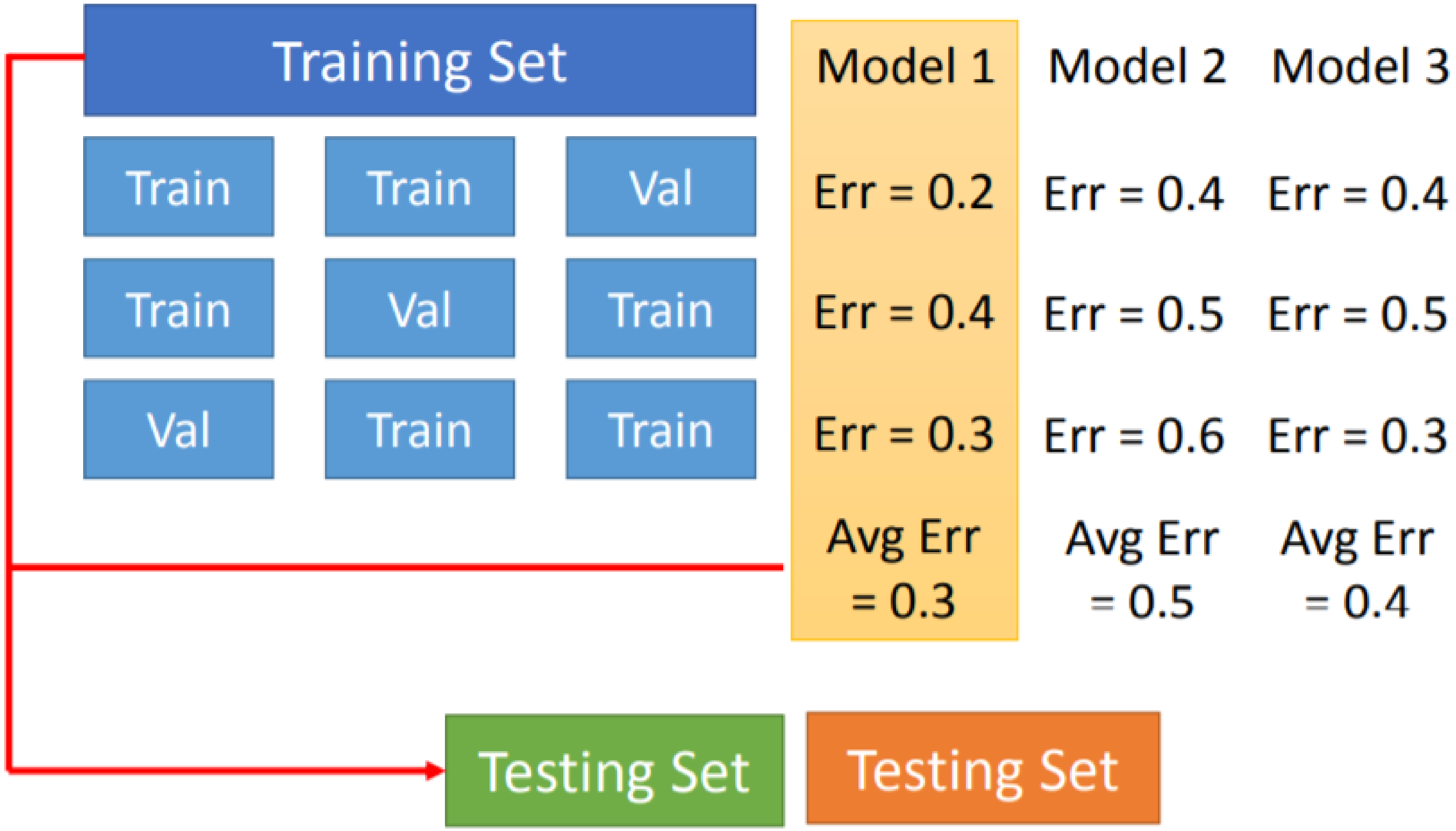
10 examples

100 examples

② Regularization



Training with cross validation



Reference: 李弘毅 ML Lecture 2 https://youtu.be/D_S6y0Jm6dQ