

Deep Neural Network & Recommendation Engine on TensorFlow



Gary Chen

Outline

Machine Learning Framework

Deep Neural Network(DNN) Tips

Recommender Engine in Matrix Factorization

Matrix Factorization with DNN



Machine Learning Framework



Machine Learning Framework

Deep Neural Network(DNN) Tips

Recommender Engine in Matrix Factorization

Matrix Factorization with DNN



Machine Learning 的種類



Regression

Semi-supervised
Learning

Transfer
Learning

Linear
Model

Unsupervised
Learning

Reinforcement
Learning

Deep
Learning

Decision Tree, SVM...,
Naive Bayse...

Non-Linear Model

classification

Supervised Learning

Machine Learning Framework



□ Machine Learning 3 Steps

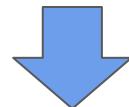
Step1

Define a set of functions



Step2

Evaluate and search



Step3

Pick the best function

特定的網絡架構

A set of functions, $f(\cdot)$
 $\{f(\theta_1), \dots, f(\theta^*), \dots, f(\theta_n)\}$

不斷修正 f 的參數

$f(\theta_{94}) \rightarrow$
 $f(\theta_{87}) \rightarrow$
 $f(\theta_{945}) \dots$

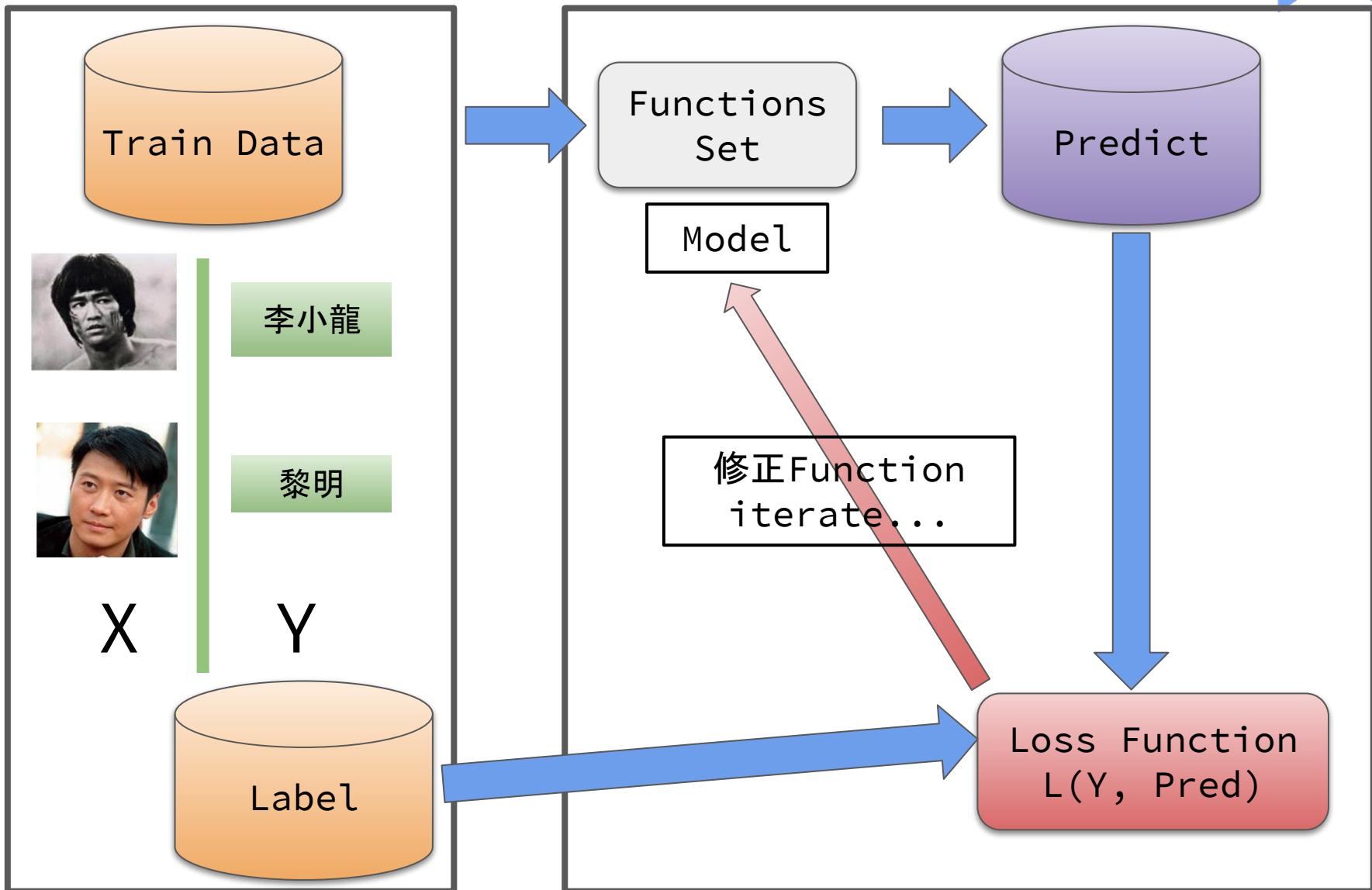
找到最適合的參數

$f(\theta^*)$

Machine Learning Framework



Machine Learning 3 Steps in Detail



Machine Learning Framework



快速理解回歸與分類

Machine Learning Framework

Case Study On **HunterXHunter**



Machine Learning Framework

Subject In Regression and Classification



Regression

西索對十二地支的實力評分？



Classification

如何判斷一個獵人是屬於什麼派系？



- Label: 連續型數值
- 可比大小

- Label: 離散型類別
- 沒有大小關係

Machine Learning Framework

Features



Regression

西索對十二地支的實力評分?

X ₁	念能力氣量(基本功)
X ₂	屬於什麼派系?
Y	0 ~ 100分

Classification

如何判斷一個獵人是屬於什麼派系?

X ₁	念能力氣量(基本功)
X ₂	水見式: 水變甜
:	:
X ₇	水見式: 水變多
X ₈	西索個性分析: 一味的單純
:	:
X ₁₃	西索個性分析: 反覆無常, 愛騙人
Y	強化系 ... 特質系(6種)

Machine Learning Framework

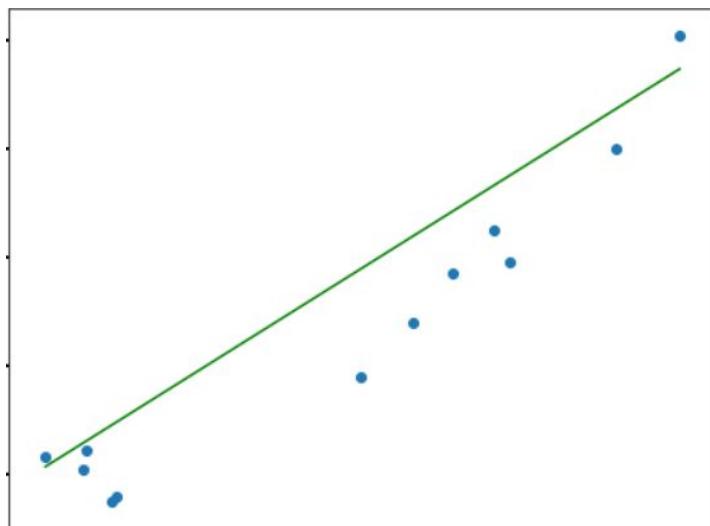
Step1: Define Function Set(Formulation)



Regression

西索對十二地支的實力評分?

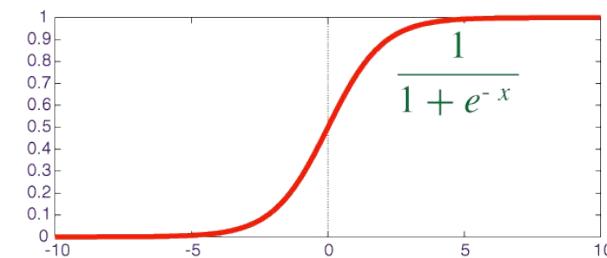
$$y = wx + b$$



Classification

如何判斷一個獵人是屬於什麼派系?

$$y = \sigma(wx + b)$$



sigmoid function

$\Rightarrow 0 \sim 1$

\Rightarrow probability

\Rightarrow one of activation functions

$$y = softmax(wx + b)$$

a method of normalization

$\Rightarrow 0 \sim 1$

\Rightarrow probability

Machine Learning Framework

Step2: Evaluate \Rightarrow Define The Loss Function!



Regression

$$\rightarrow z = wx_i + b$$

$$\frac{1}{n} \sum_i^n (\hat{y}_i - z)^2$$

mean square error

$$\frac{1}{n} \sum_i^n |\hat{y}_i - z|$$

mean absolute error

- 點對點的誤差平均
- 值域: real number, 任何值

Classification

$$\rightarrow z = \sigma(wx + b)$$

or

$$\rightarrow z = softmax(wx + b)$$

$$\frac{-1}{n} \sum_i^n (\hat{y}_i \cdot log(z))$$

cross entropy

- 資料分布的誤差
- 值域: output視為機率, 0 ~ 1 之間, 處理資料時需要做特殊處理
- cross entropy直觀理解

Machine Learning Framework

Step 3: Pick Best Function



Regression

Classification

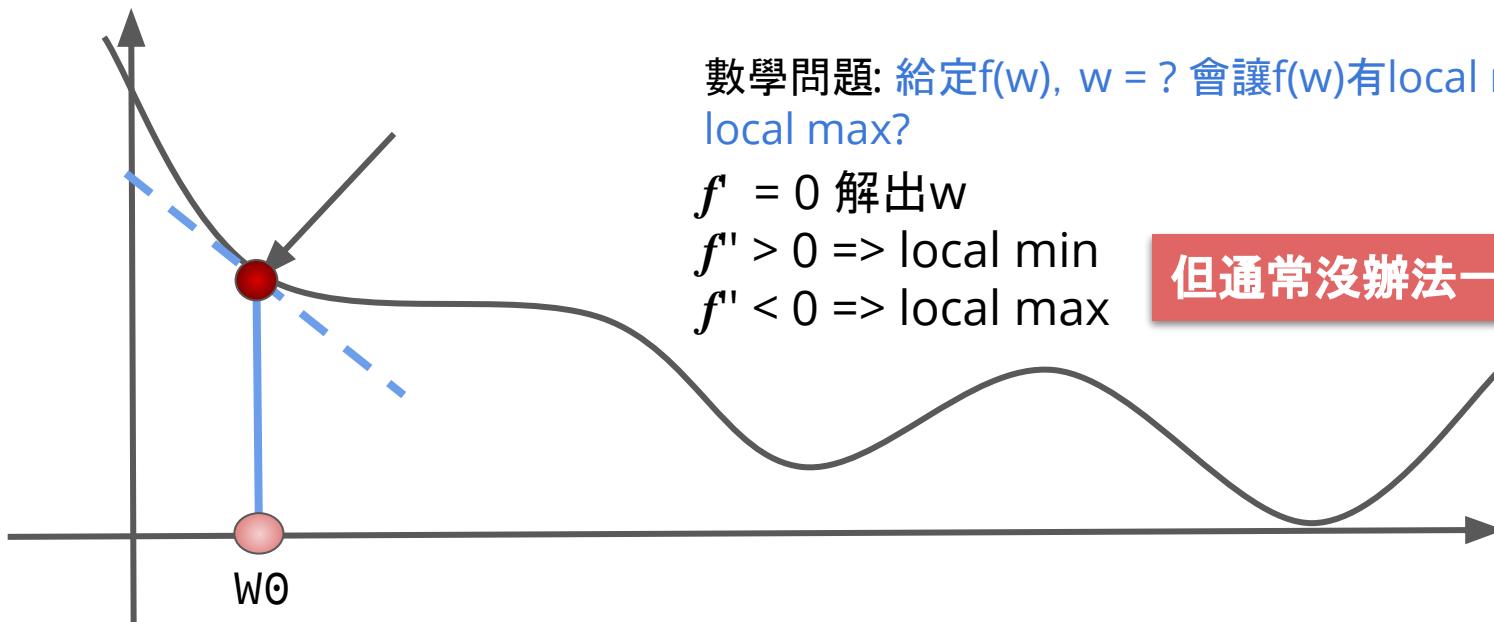
$$\operatorname{argmin}_{w,b} L(w, b)$$

最佳化問題

微分的定義

$$\lim_{\Delta w \rightarrow 0} \frac{L(w + \Delta w) - L(w)}{\Delta w}$$

w些微的變化，會讓L改變多少？



Machine Learning Framework

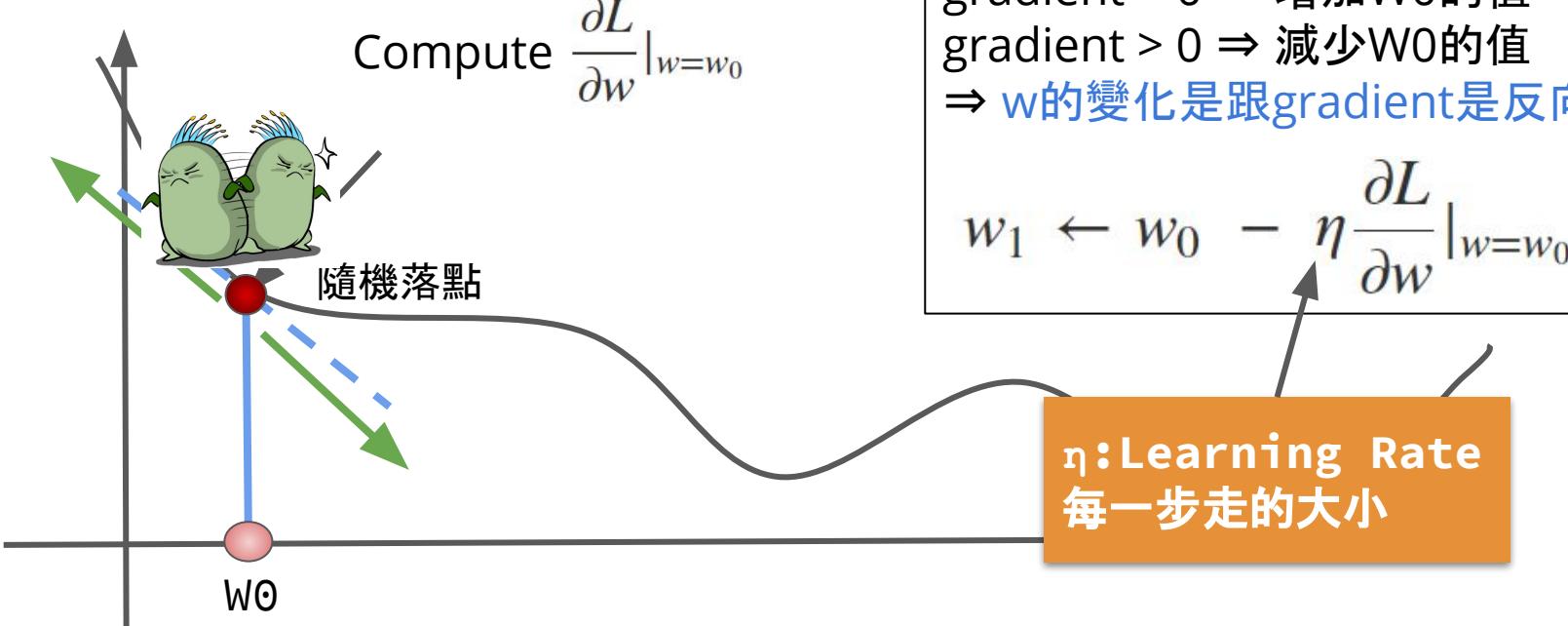
Step 3: Pick Best Function



Regression

Classification

$$\operatorname{argmin}_{w,b} L(w, b) \rightarrow \text{最佳化問題} \rightarrow \text{Gradient Descent}$$



Machine Learning Framework

Step 3: Pick Best Function



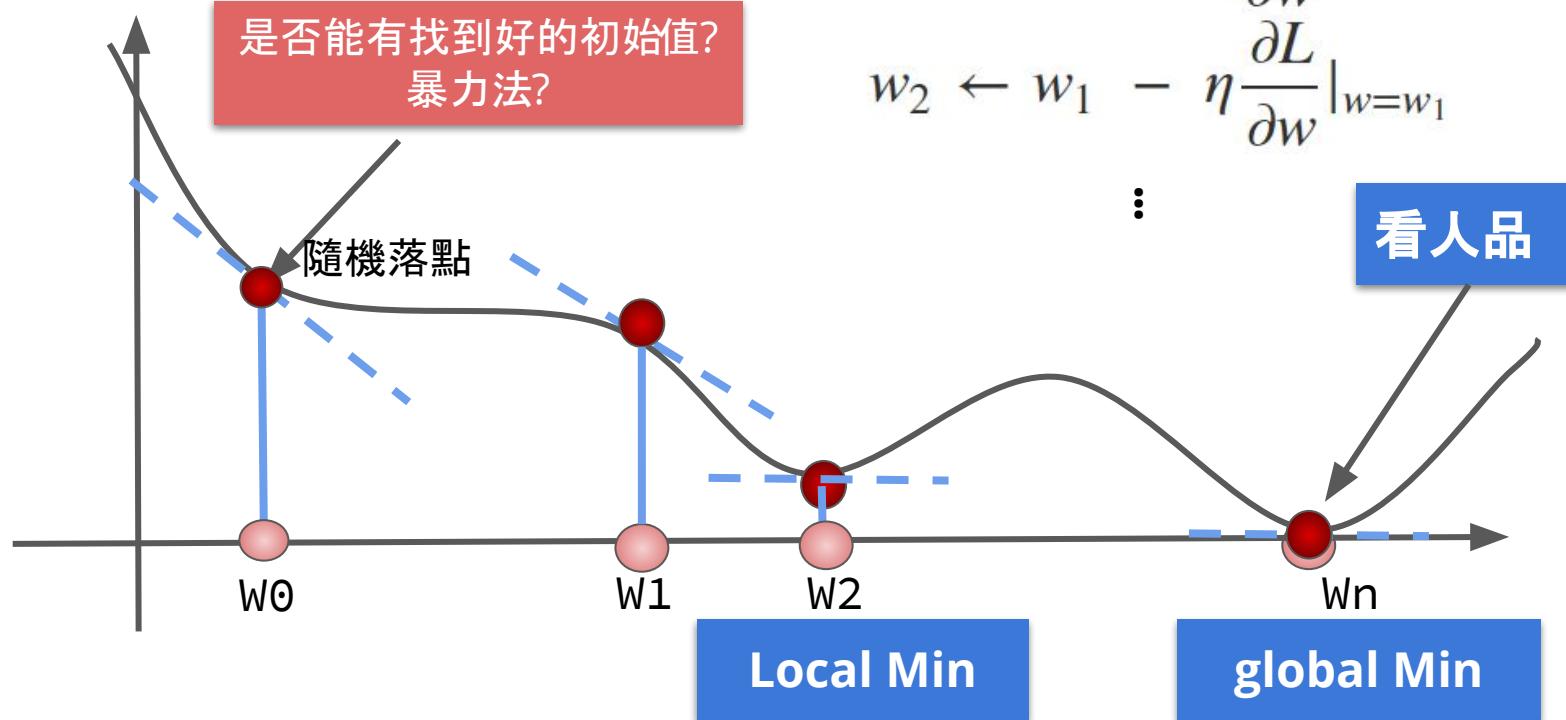
Regression

Classification

$$\operatorname{argmin}_{w,b} L(w, b)$$

最佳化問題

Gradient Descent



Machine Learning Framework

Step 3: Pick Best Function



蟻王用暴力法找出
尼特羅百式觀音掌
法當中的規律，突
破防守！



```
initFn = tf.glorot_uniform_initializer()  
initFn = tf.glorot_normal_initializer()  
init_var = tf.Variable(initFn(shape=[100, 80]))
```

Ref: [Xavier uniform initializer](#)

Machine Learning Framework

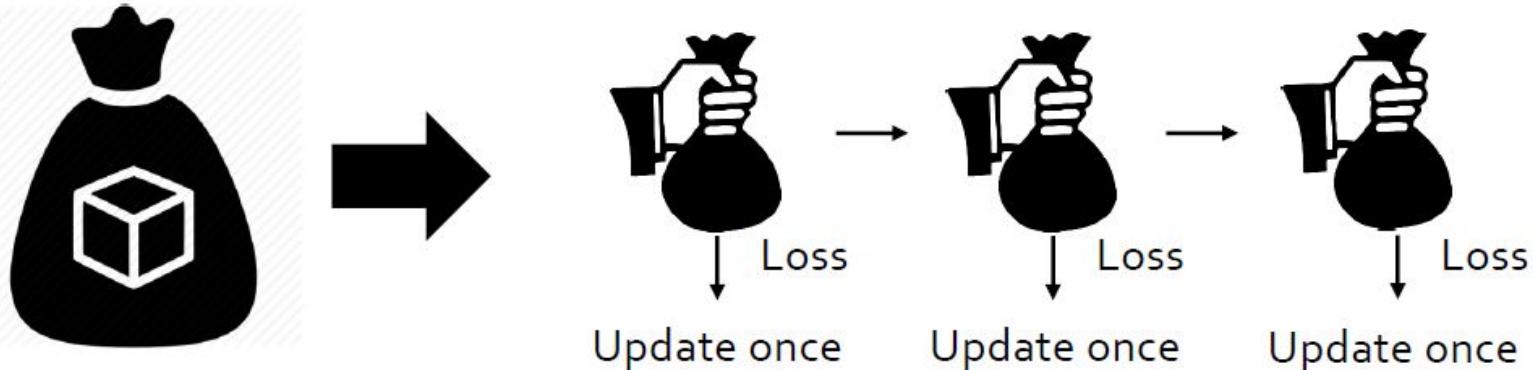
Step 3: Pick Best Function (Stochastic Gradient Descent)



Regression

Classification

- Shuffle training data
- Mini-batch update



- 假設有10000筆資料, batch = 100
 - Train一輪(one epoch)就更新了100次weights
 - 一次丟所有data下去只更新了一次, 沒有效率
- Batch size advice: 32, 64, 128, 256, 512, ...

Machine Learning Framework

回頭看看資料~



Regression

西索對十二地支的實力評分?

Data	Label
$x_1 \dots x_2$	y
...	78.5
...	87.8
...	35.0
...	64.0
...	33.64
...	90.2

Classification

如何判斷一個獵人是屬於什麼派系?

Data	Label
$x_1 \dots x_{13}$	y
...	1
...	2
...	3
...	4
...	5
...	6

離散的類別想辦法轉換成數字

Machine Learning Framework



分類資料如何處理

Classification

如何判斷一個獵人是屬於什麼派系？

Data	Label	
$x_1 \dots x_{13}$	y	
...	1	強化系
...	2	變化系
...	3	放出系
...	4	具現化系
...	5	操作系
...	6	特質系

這樣有點問題，各類別間產生了原本資料中沒有的距離的關係...



$\text{Distance}(\text{強化系}, \text{變化系}) = 2 - 1 = 1$
 $\text{Distance}(\text{強化系}, \text{放出系}) = 3 - 1 = 2$

解決方式？

Machine Learning Framework

分類資料如何處理



Classification

如何判斷一個獵人是屬於什麼派系?

Data	Label (One Hot Encoding)						
$x_1 \dots x_{13}$	y_1	y_2	y_3	y_4	y_5	y_6	
...	1	0	0	0	0	0	強化系
...	0	1	0	0	0	0	變化系
		
...	0	0	0	0	0	1	特質系

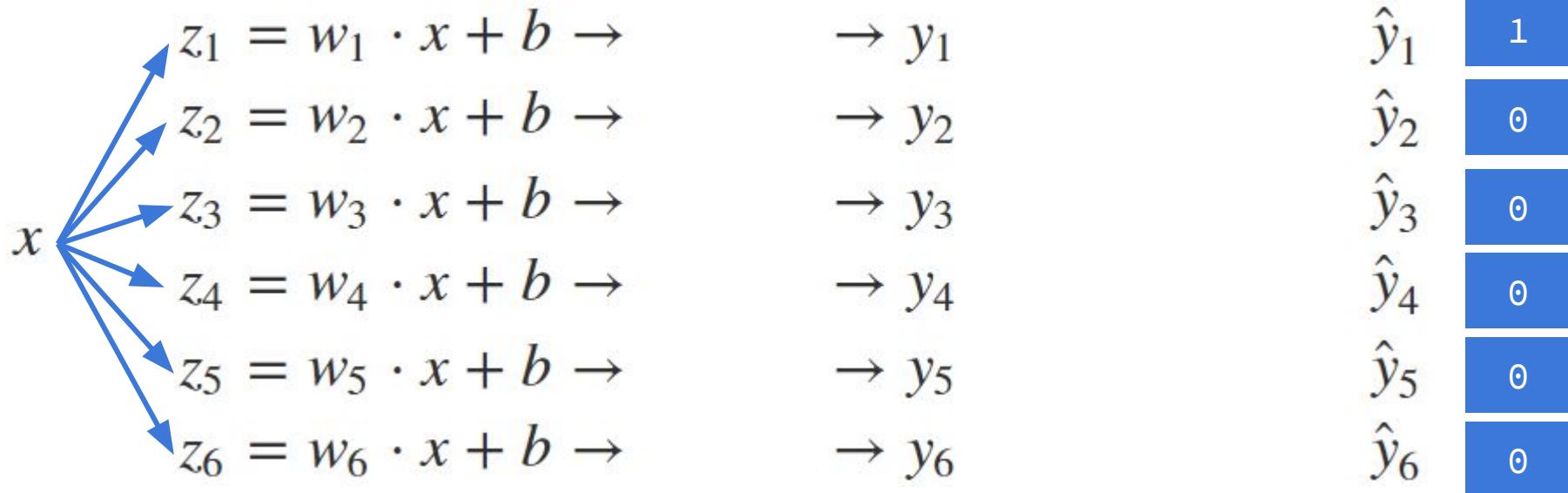
Label的形式是Vector, 怎麼處理?

Machine Learning Framework



如何判斷一個獵人是屬於什麼派系?

一個類別一個Function對付他!



Probability:

- $1 > y_i > 0 \quad y_i = P(C_i | x)$
- $\sum_i y_i = 1$

Sigmoid不滿足這個需求!

Machine Learning Framework

如何判斷一個獵人是屬於什麼派系? (Softmax)

Activation function: Softmax

$$\begin{aligned}x &\rightarrow z_1 = w_1 \cdot x + b & 5 \\&\rightarrow z_2 = w_2 \cdot x + b & 1 \\&\rightarrow z_3 = w_3 \cdot x + b & 3 \\&\rightarrow z_4 = w_4 \cdot x + b & -3 \\&\rightarrow z_5 = w_5 \cdot x + b & -3 \\&\rightarrow z_6 = w_6 \cdot x + b & 0\end{aligned}$$

$$\text{softmax}(z) = \frac{e^{z_i}}{\sum_{k=1}^n e^{z_k}}$$

$$\begin{array}{c|c}y_1 & 0.8 \\y_2 & 0.01 \\y_3 & 0.11 \\y_4 & 0 \\y_5 & 0 \\y_6 & 0\end{array}$$

Cross Entropy

$$-\sum_{i=1}^6 \hat{y}_i \ln y_i$$



強化系

$$\begin{array}{c|c}\hat{y}_1 & 1 \\ \hat{y}_2 & 0 \\ \hat{y}_3 & 0 \\ \hat{y}_4 & 0 \\ \hat{y}_5 & 0 \\ \hat{y}_6 & 0\end{array}$$

計算argmax, 最大的數字都在
index 0 ⇒ 命中!

Probability:

- $0 < y_i < 1 \quad y_i = P(C_i | x)$
- $\sum_i y_i = 1$

Machine Learning Framework

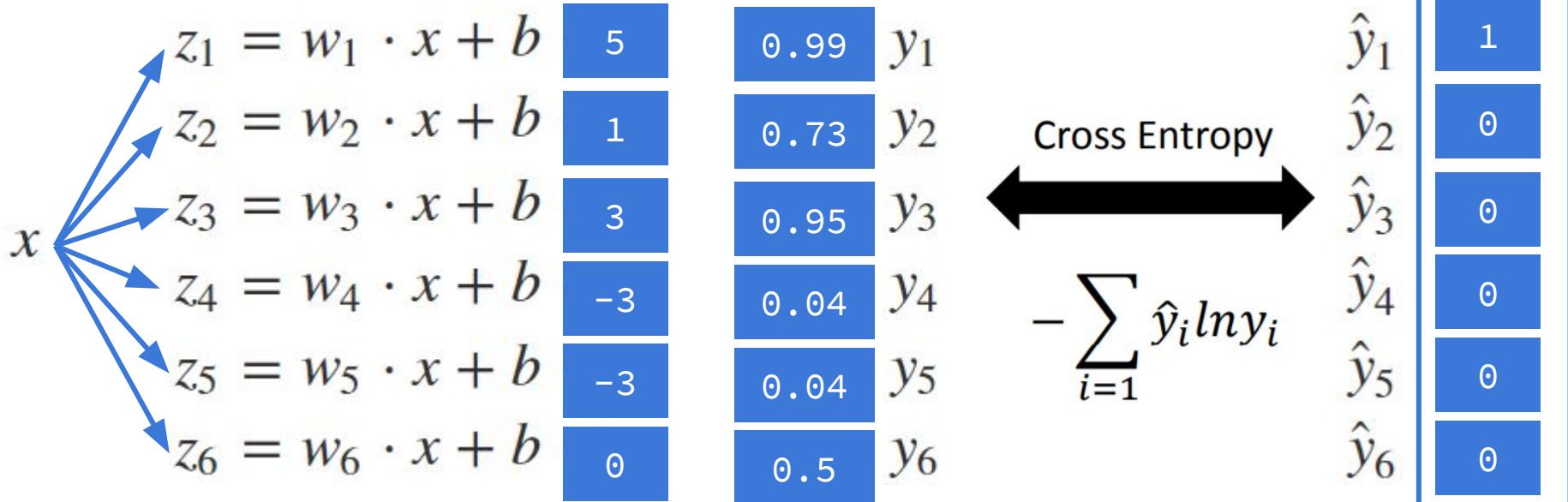
如何判斷一個獵人是屬於什麼派系? (Sigmoid)

Activation function: Sigmoid

$$\sigma(z) = \frac{1}{1 + \exp(-z)}$$



強化系



算argmax是沒有意義的! 要考慮的是每個class的分數

Machine Learning Framework

Softmax與Sigmoid的差別



$$\text{softmax}(z) = \frac{e^{z_i}}{\sum_{k=1}^n e^{z_k}}$$

$$\sigma(z) = \frac{1}{1 + \exp(-z)}$$

- 所有類別視為一個母體
- Single-label for one record

一個獵人只能屬於一個系別!

- 不考慮類別間的關係
- Multi-label for one record

一個獵人可以有多個系別
ex: 同時是強化系與特質系!

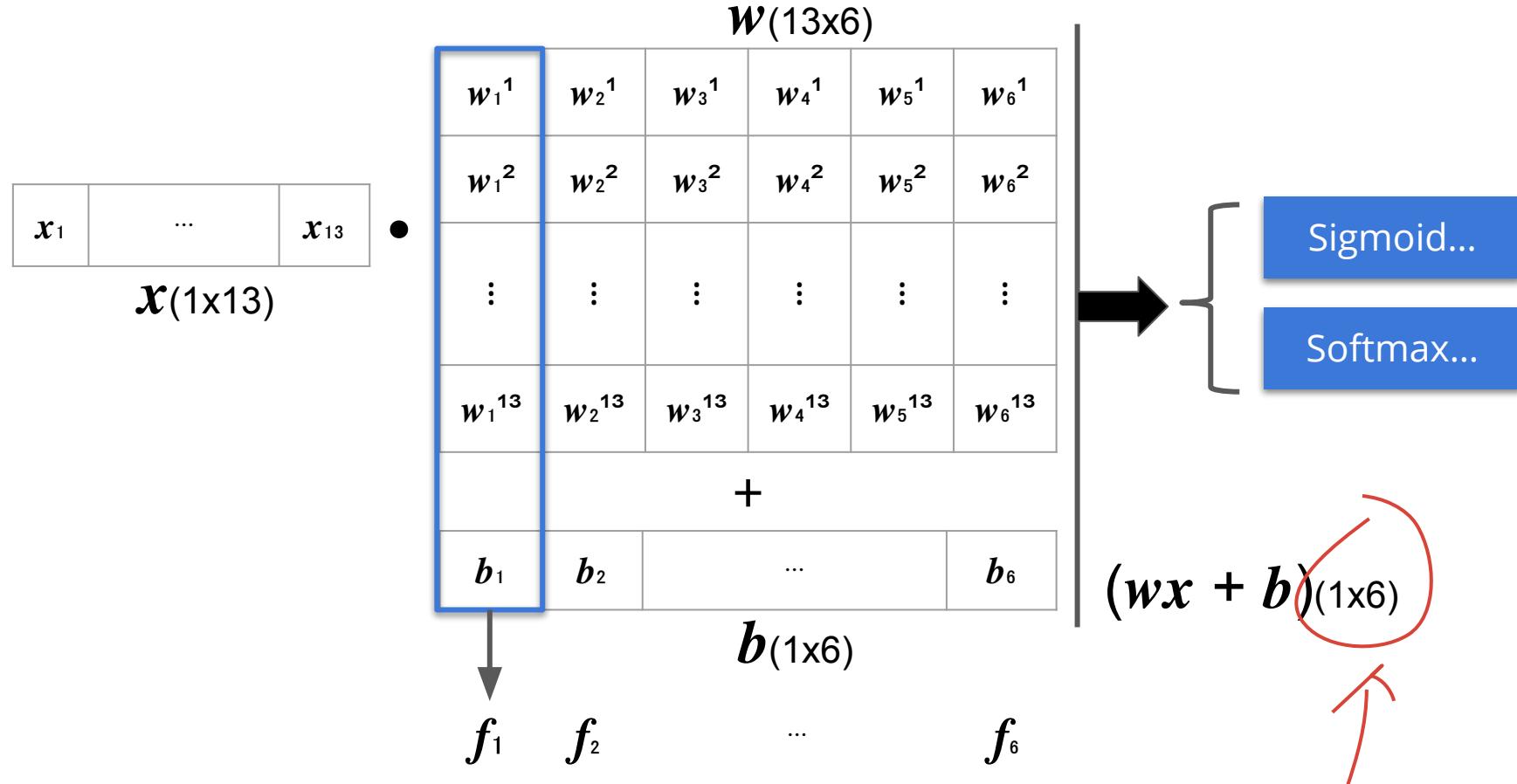
Computes sigmoid cross entropy given `logits`.

Measures the probability error in discrete classification tasks in which each class is independent and not mutually exclusive. For instance, one could perform multilabel classification where a picture can contain both an elephant and a dog at the same time.

Machine Learning Framework



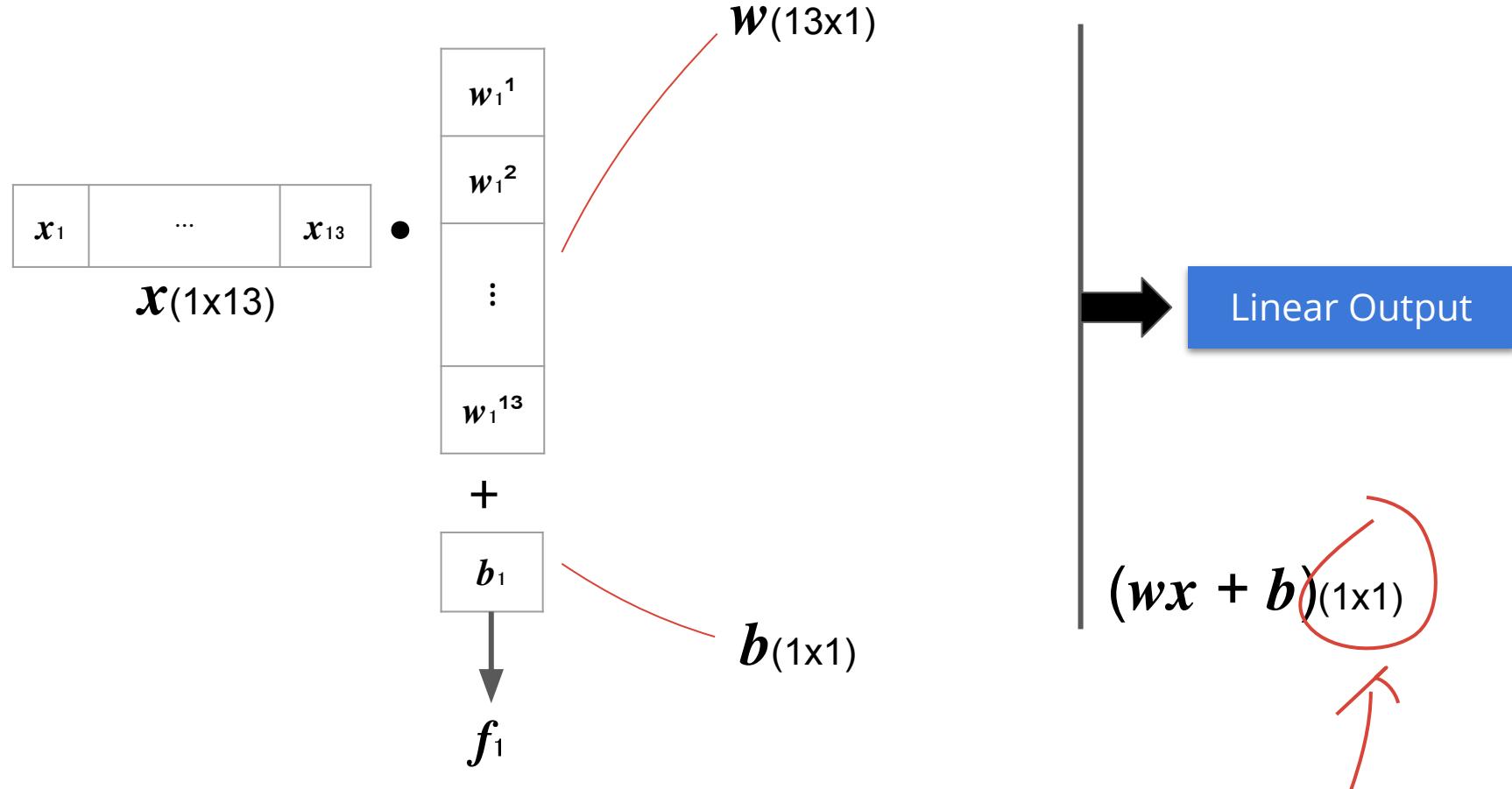
以向量運算表示(分類)~



Machine Learning Framework



以向量運算表示(回歸)~

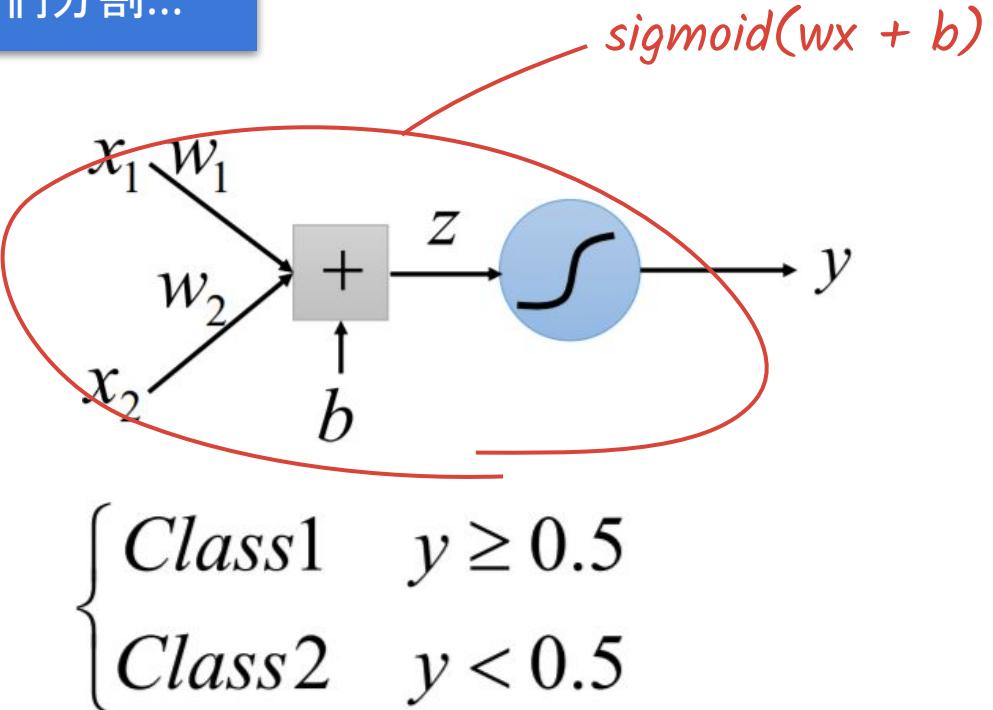
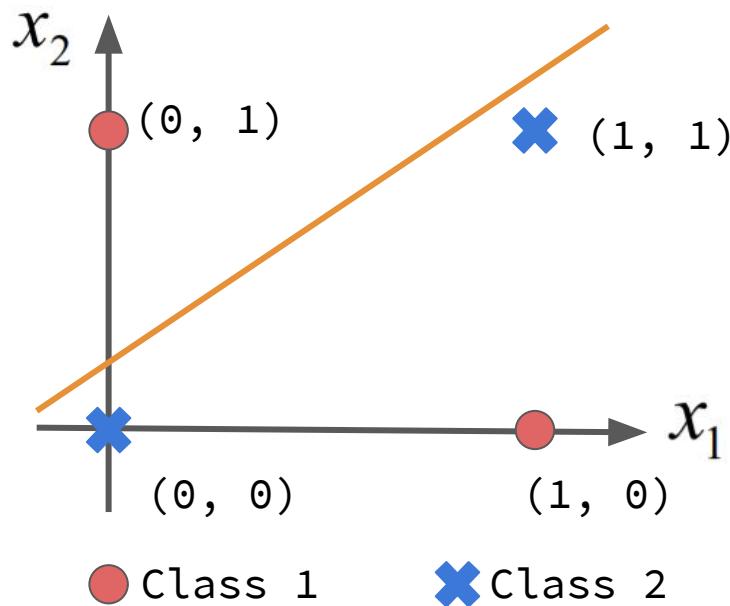


Machine Learning Framework



以分類來說, 常碰到的瓶頸(Limitation)

以下狀況似乎不能用一條線將他們分割...



Machine Learning Framework

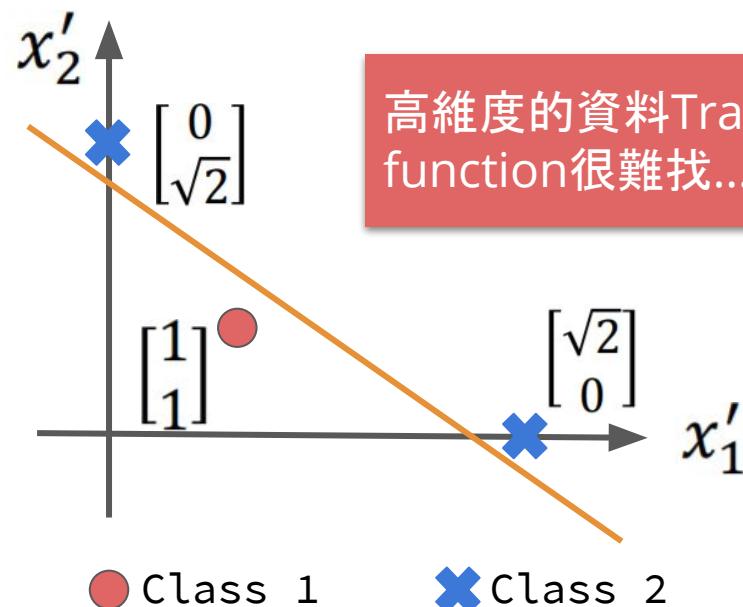
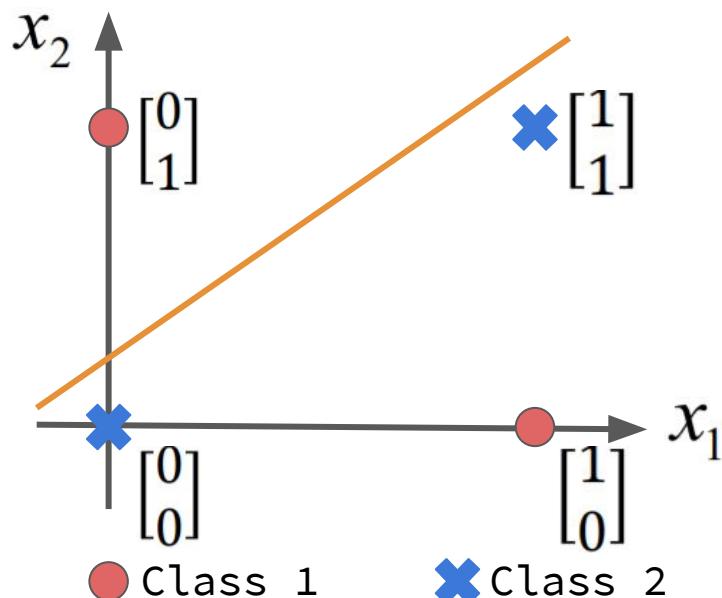
以分類來說，常碰到的瓶頸(Limitation)



Feature Transformation!

$$trans(x) = \begin{bmatrix} x \text{到}(0, 0) \text{ 的距離} \\ x \text{到}(1, 1) \text{ 的距離} \end{bmatrix}$$

$$x'_1: \text{distance to } \begin{bmatrix} 0 \\ 0 \end{bmatrix}$$
$$x'_2: \text{distance to } \begin{bmatrix} 1 \\ 1 \end{bmatrix}$$

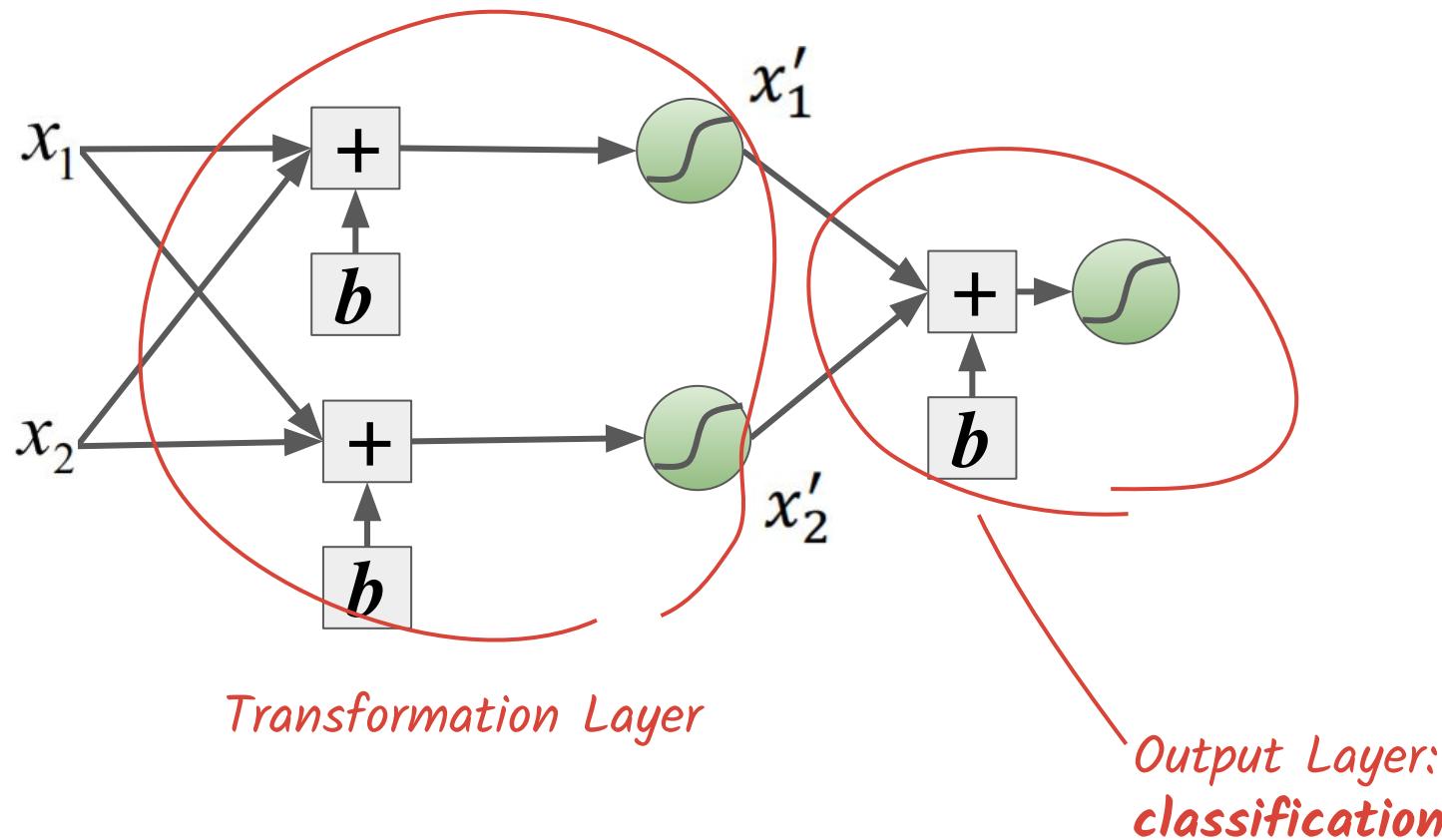


Machine Learning Framework



Cascading Functions

也許Machine可以自己learn feature transformation...



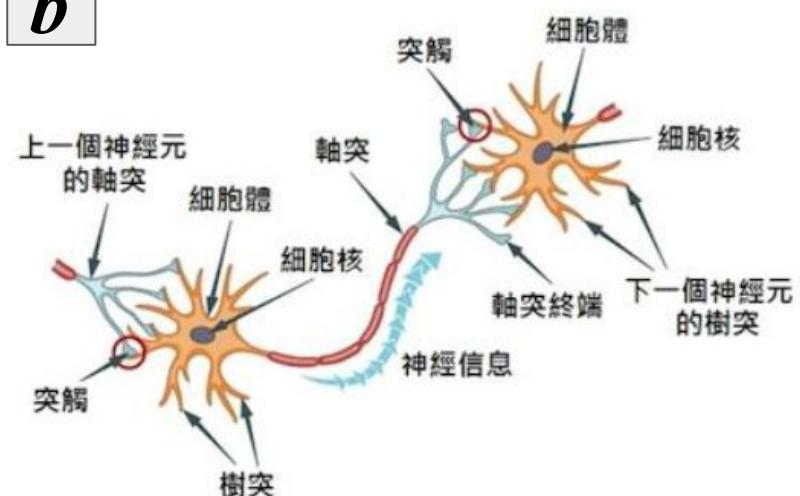
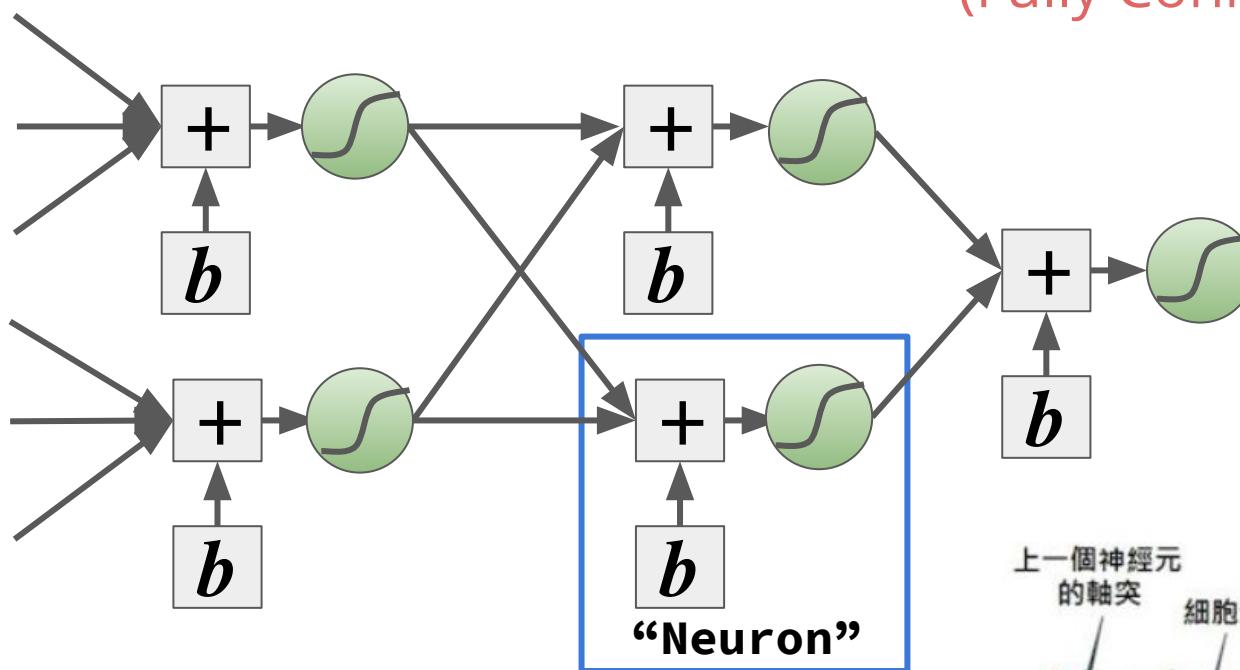
Machine Learning Framework



Cascading Functions

串聯 + 並聯

Deep Learning!
(Fully Connected Structure)



Deep Neural Network(DNN) Tips



Machine Learning Framework

Deep Neural Network(DNN) Tips

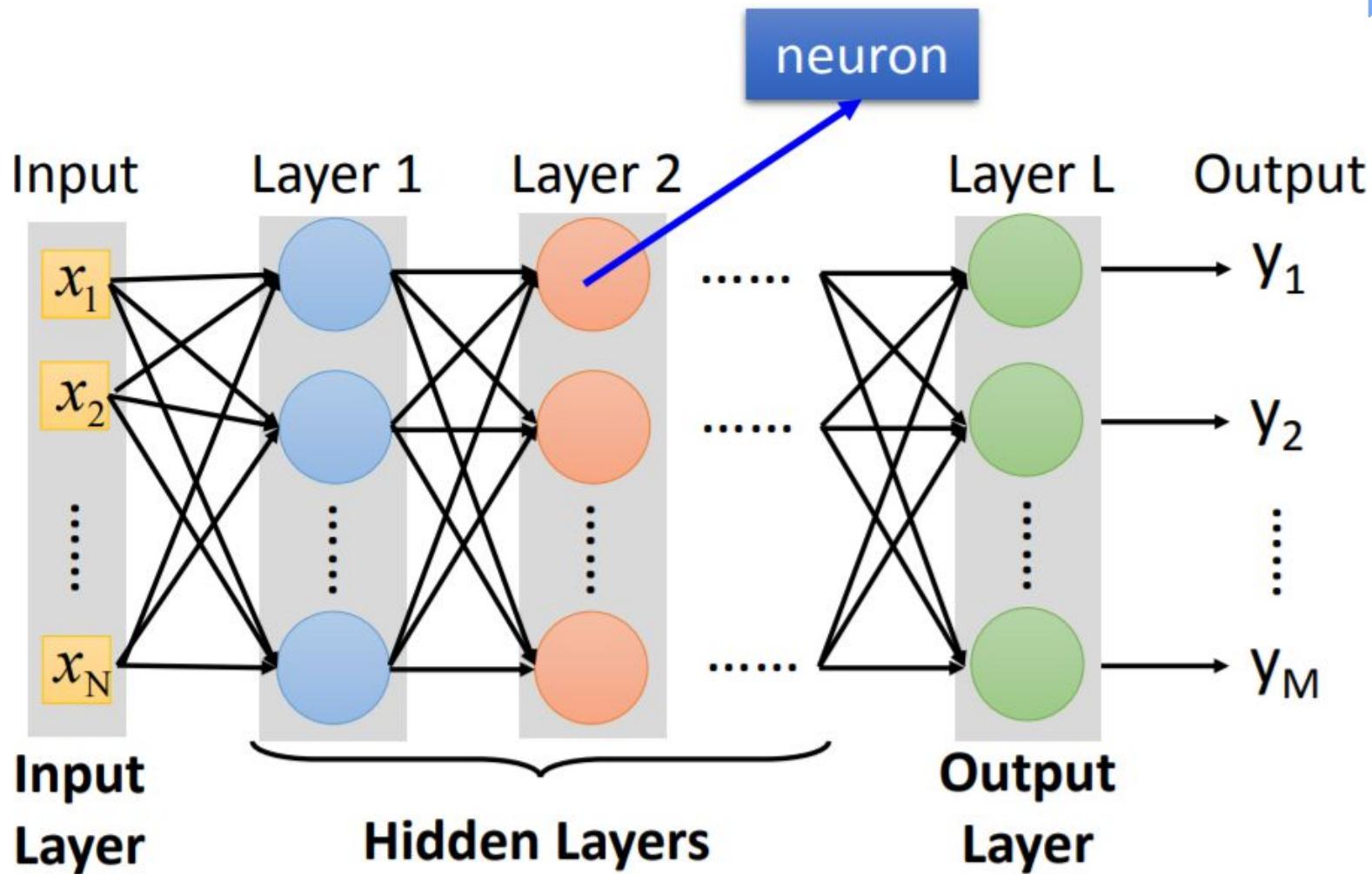
Recommenedation Engine in Matrix Factorization

Matrix Factorization with DNN



Deep Neural Network(DNN) Tips

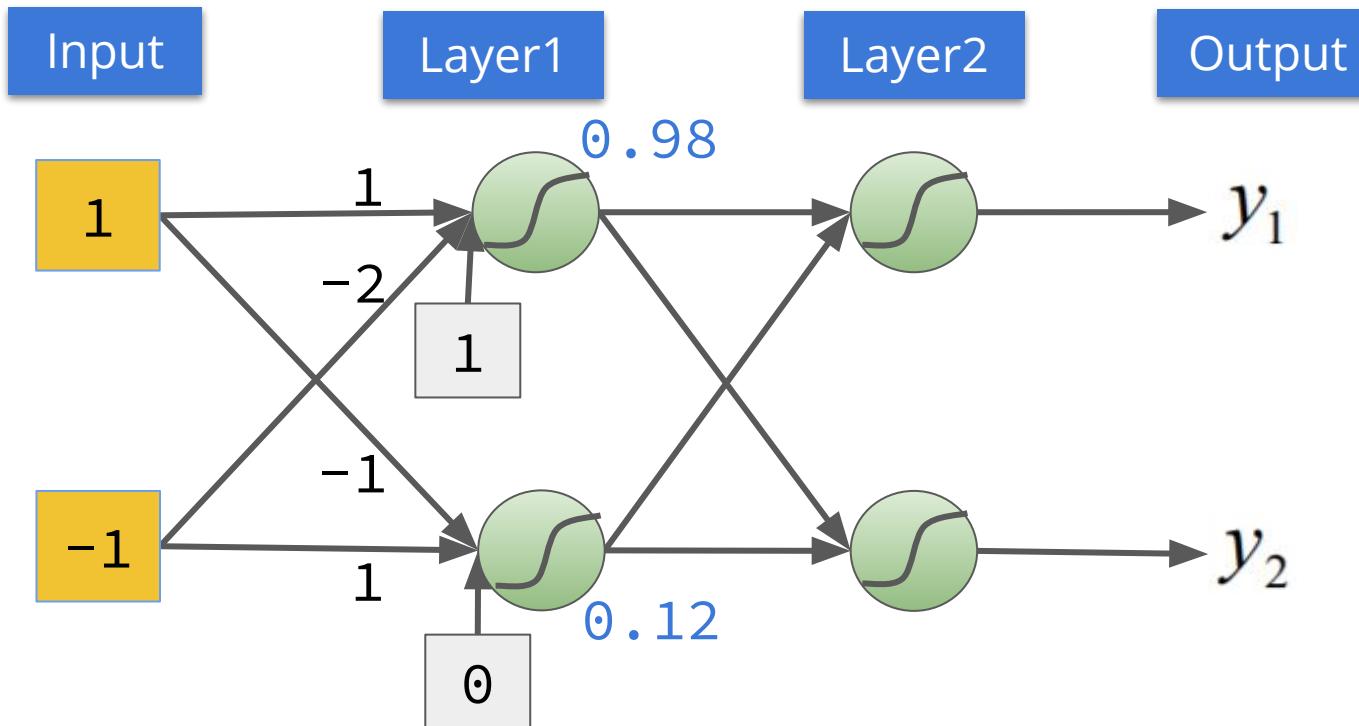
DNN Structure(Fully Connected)



Deep Neural Network(DNN) Tips



Matrix Operation



$$\sigma \left(\begin{matrix} 1 & -1 \\ 1 & -2 \\ -2 & 1 \end{matrix} + \begin{matrix} 1 & 0 \\ 1 & 0 \end{matrix} \right) = \begin{matrix} 0.98 & 0.12 \end{matrix}$$

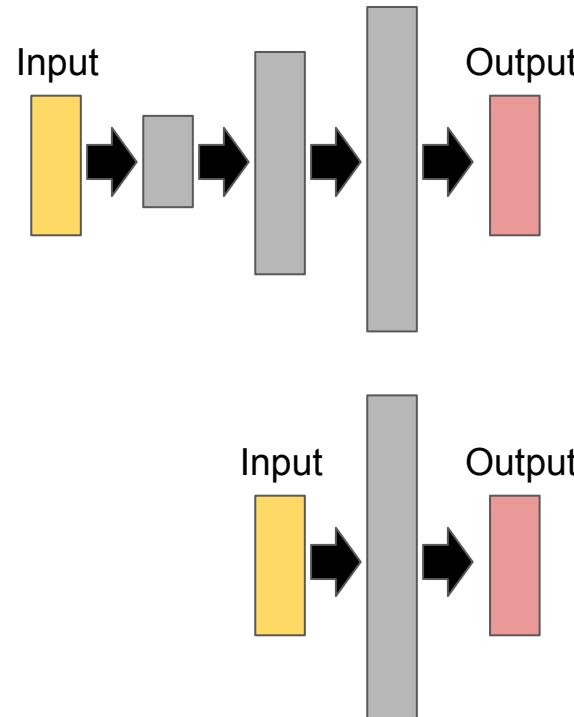
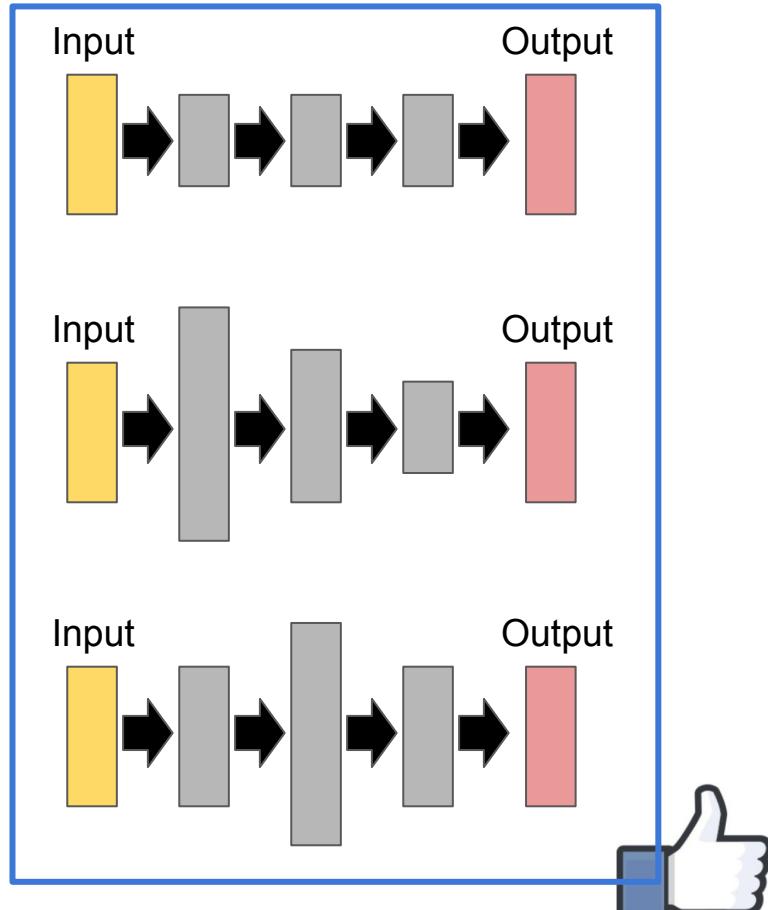
一層Layer
的組成元素
 σ, w, b

data weight bias

Deep Neural Network(DNN) Tips



Network的架構(哪種比較好?)



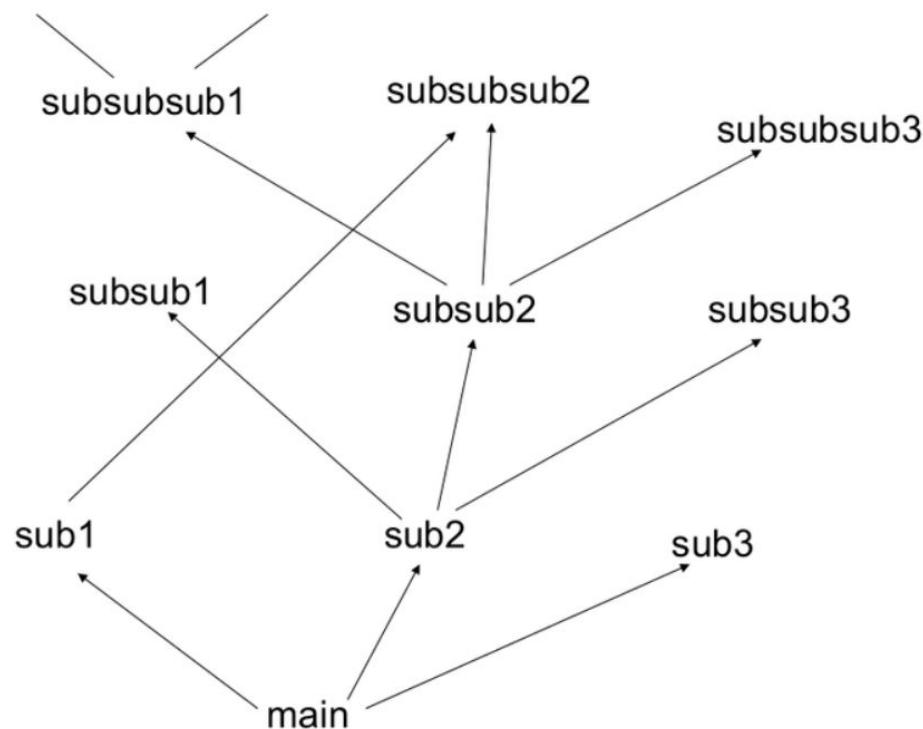
Deep Neural Network(DNN) Tips

Network的架構



- Deep Neural Network可以想成一種模組化的概念
- 不要一招打天下

Coding的概念：
不要在main重頭寫到尾!



Deep Neural Network(DNN) Tips

Network的架構

Modularization



Deep neural
networks learn
hierarchical feature
representations

這是**Data Driven**的模組化，你沒法決定每一層
Layer到底執行什麼功能，但是沒有模組化就沒有
機會！

Deep Neural Network(DNN) Tips



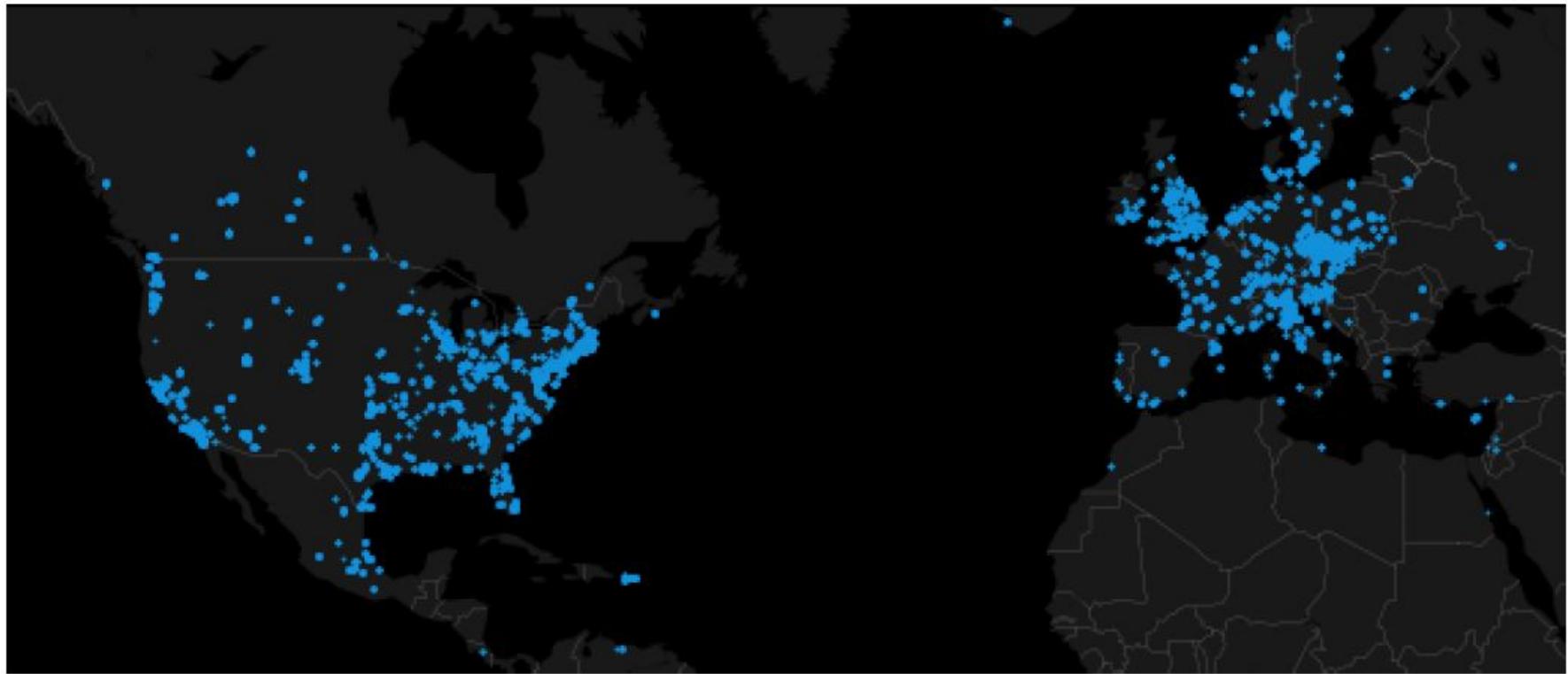
Classification:
說太多了，動手實作吧！

Deep Neural Network(DNN) Tips

直接來實作吧！



寶可夢過去出現的時間與地點紀錄 (dataset from Kaggle)



Ref: <https://www.kaggle.com/kostyabahshetsyan/d/seminiy/predictmall/pokemon-geolocation-visualisations/notebook>

Deep Neural Network(DNN) Tips

範例資料: Raw Data Overview



latitude	longitude	appearedTimeOfDay	appearedYear	terrainType	closeToWater	city	weather	temperature	windSpeed	cooc_151	class
20.525745	-97.460829	night	2016	14	0	Mexico_City	Foggy	25.5	4.79	0	16
20.523695	-97.461167	night	2016	14	0	Mexico_City	Foggy	25.5	4.79	0	133
38.90359	-77.19978	night	2016	13	0	New_York	Clear	24.2	4.29	0	16
47.665903	-122.312561	night	2016	0	1	Los_Angeles	PartlyCloudy	15.6	5.84	0	13
47.666454	-122.311628	night	2016	0	1	Los_Angeles	PartlyCloudy	15.6	5.84	0	133
-31.95498	115.853609	night	2016	13	0	Perth	PartlyCloudy	16.5	6.39	0	21
-31.954245	115.852038	night	2016	13	0	Perth	PartlyCloudy	16.5	6.4	0	66
26.235257	-98.197591	night	2016	13	0	Chicago	Clear	28	11.26	0	27
20.525554	-97.4588	night	2016	14	0	Mexico_City	Foggy	25.5	4.79	0	35
32.928558	-84.340278	night	2016	8	0	New_York	Clear	23.7	3.94	0	19
32.930646	-84.339867	night	2016	8	0	New_York	Clear	23.7	3.94	0	116
32.943651	-84.334443	night	2016	8	0	New_York	Clear	23.7	3.94	0	74
26.235552	-98.197249	night	2016	13	0	Chicago	Clear	28	11.26	0	16
20.52577	-97.460237	night	2016	14	0	Mexico_City	Foggy	25.5	4.79	0	19
26.236029	-98.196908	night	2016	13	0	Chicago	Clear	28	11.26	0	19
47.664333	-122.312645	night	2016	0	1	Los_Angeles	PartlyCloudy	15.6	5.84	0	19
20.526489	-97.460745	night	2016	14	0	Mexico_City	Foggy	25.5	4.79	0	16
53.611417	-113.369528	night	2016	12	0	Edmonton	Clear	8.9	1.47	0	13

問題: 會出現哪一隻神奇寶貝呢 ?

Deep Neural Network(DNN) Tips

範例資料：預測會出現哪一隻神奇寶貝？



- ❑ 時間: local.hour,local.month, DayofWeek...
- ❑ 天氣: temperature,windSpeed,pressure...
- ❑ 位置: longitude,latitude,pokestop...
- ❑ 環境: closeToWater,terrainType...
- ❑ 十分鐘前有無出現其他寶可夢
 - ❑ 例如: cooc_1=1十分鐘前出現過class=1隻寶可夢
- ❑ class 就是我們要預測目標

為了簡化複雜度，我們只挑出五種類別

No.4 小火龍



No.43 走路草



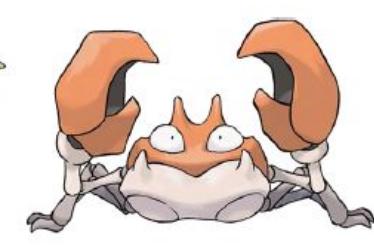
No.56 火爆猴



No.71 喇叭芽



No.98 大鉗蟹



Class 0

Class 1

Class 2

Class 3

Class 4

Data Preprocessing Tips



- ❑ Neural network運算只接受數字(numeric)
- ❑ 如何處理非數值資料?
 - ❑ Categorical variables
 - ❑ Ordinal variables
- ❑ 不同feature的數值範圍差異是否會有影響?
 - ❑ 溫度: 0 ~ 40度
 - ❑ 距離: 0 ~ 10000公尺

Data Preprocessing Tips



- ❑ Categorical variables
 - ❑ One Hot Encoding: 回想獵人的系別, 強化系、變化系...
 - ❑ 避免產生不必要的距離關係!
- ❑ 特殊Case: 地址
 - ❑ 台中市大馬路無尾巷
⇒ 轉成經緯度 {25.04, 121.61}

Deep Neural Network(DNN) Tips

Data Preprocessing Tips



- ❑ Ordinal variables(順序資料)
 - ❑ Def: 跟數值一樣有大小關係(順序關係)卻又像類別變數
 - ❑ For example: 衣服size, {S, M, L, XL}
 - ⇒ {1, 2, 3, 4}
 - ⇒ 當作類別變數One Hot Encoding
 - S: [1, 0, 0, 0]
 - M: [0, 1, 0, 0]
 - ...
- ❑ Create a new feature using mean or median



UID	Age
P1	0-17
P2	0-17
P3	55+
P4	26-35

→

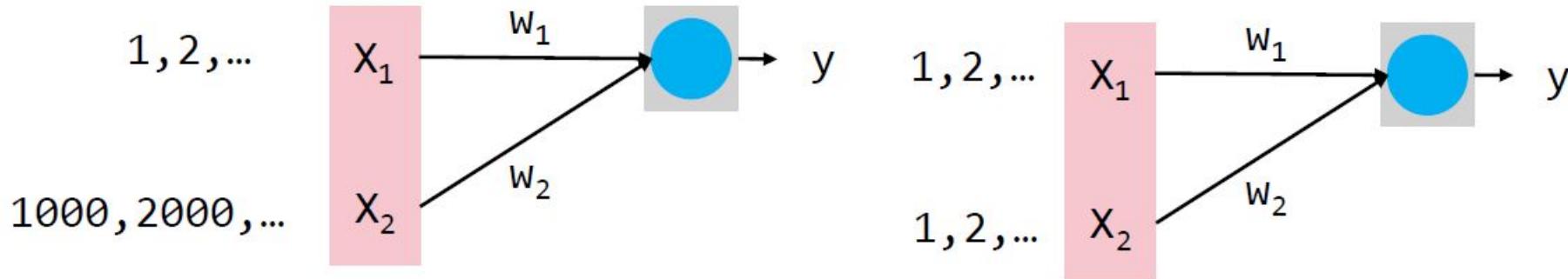
UID	Age
P1	15
P2	15
P3	70
P4	30

Deep Neural Network(DNN) Tips

Data Preprocessing Tips



- 對於numeric欄位，處理不同數值範圍
 - 非常建議normalize, 不要讓幾個變數主導整個function！



w_2 的修正(Δw)對於 loss 的影響比較大



Data Preprocessing Tips



- ❑ Normalize methods
 - ❑ Min max scaler to $[0, 1]$ $\frac{x_i - \min}{\max - \min}$ (注意outlier)
 - ❑ Scale to standard normal distribution $\mu = 0, \sigma = 1$
(Z-score standardize)
 - ❑ 以上兩種方式沒有guideline說什麼狀況下使用最好

Deep Neural Network(DNN) Tips

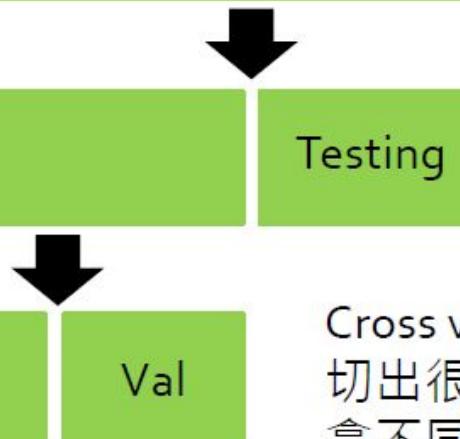
How to Treat Your Data?



- ❑ Split train, valid, test data
 - ❑ Valid data用來挑選model
 - ❑ Test data檢驗模型的generalization
- ❑ 為簡化複雜度, Demo用traing + valid

理論上

手邊收集到的資料



挑選出最好的模型後，拿 testing 檢驗 generalization

Cross validation:
切出很多組的 (training, validation) 再拿不同組訓練模型，挑選最好的模型

Deep Neural Network(DNN) Tips

寶可夢Deep Neural Network Tutorial



filename	tutorial_dnn_practice.ipynb
lab filename	lab_tutorial_dnn_practice.ipynb
input	200 features
label	5 class
neural network	DNN

Lab File For You

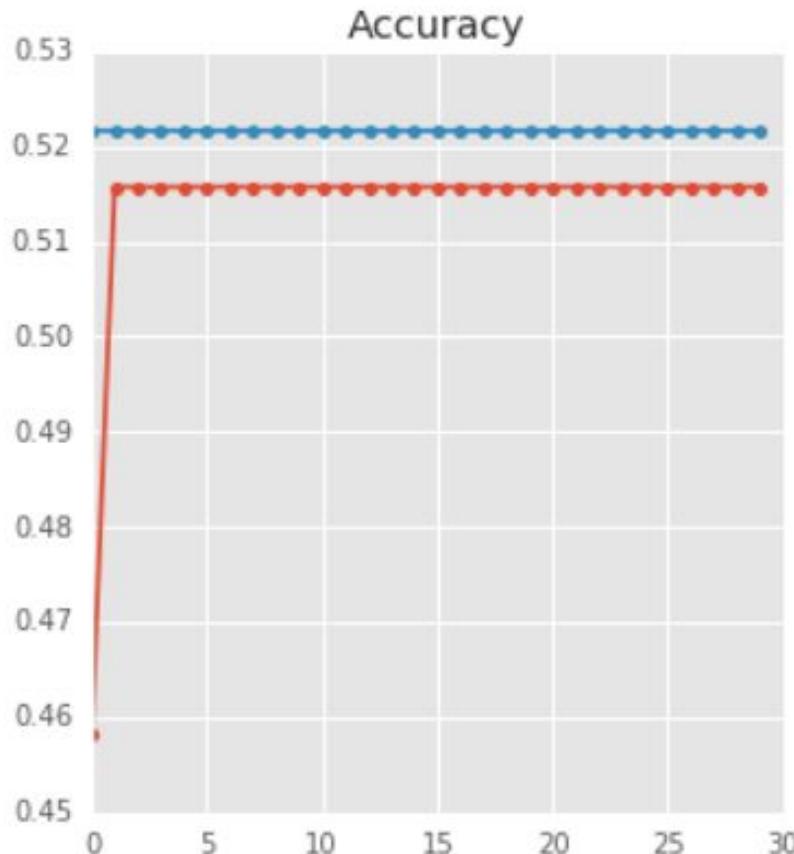
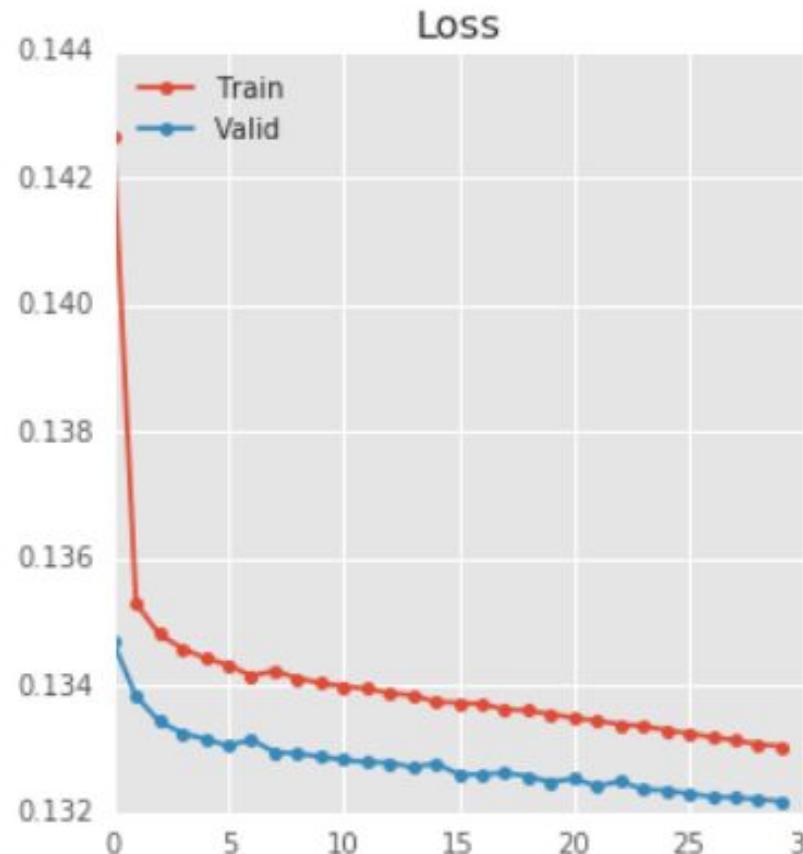
Deep Neural Network(DNN) Tips

Result of ModelMSE



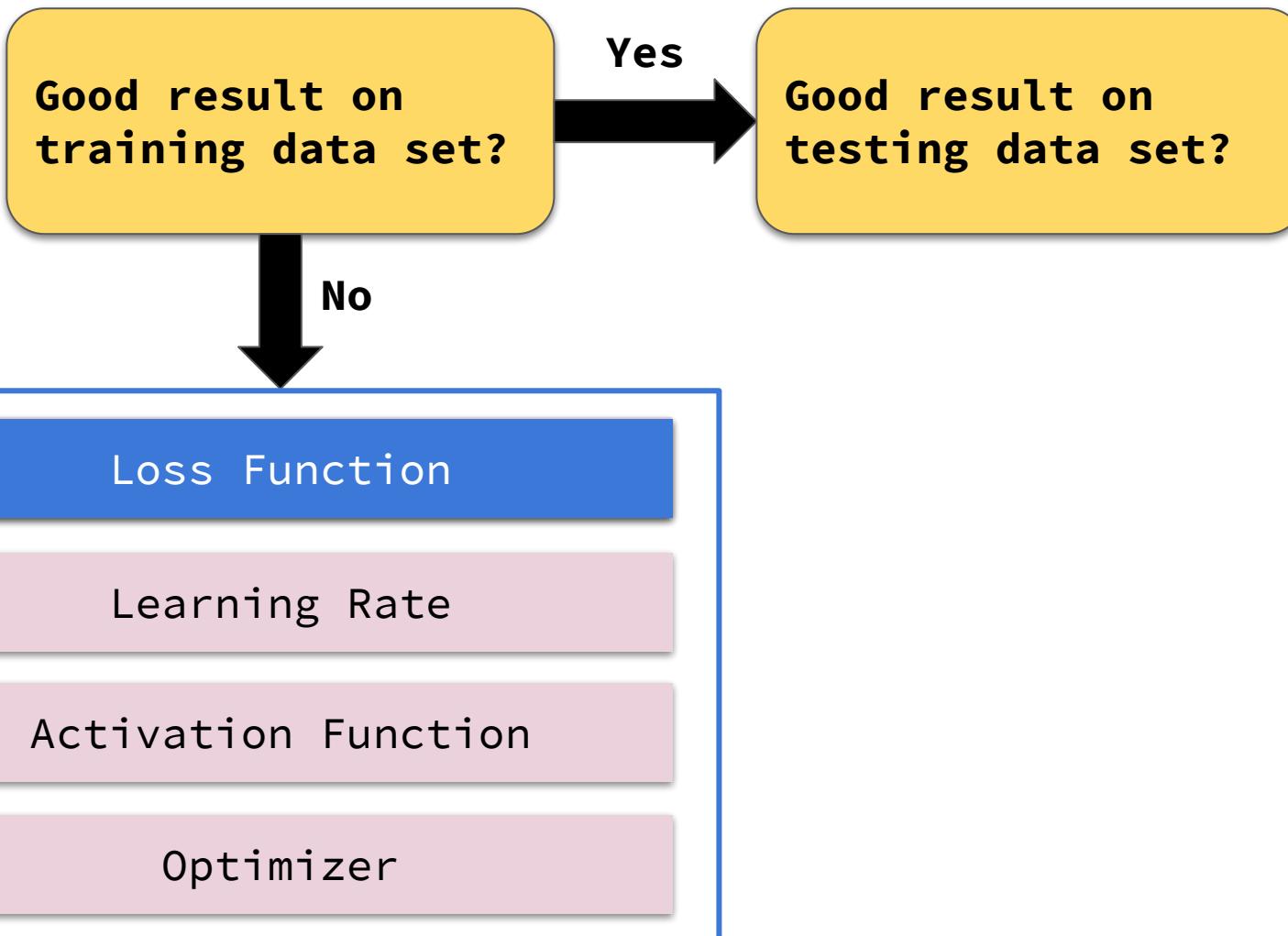
mean square error model!

30/30. train loss: 0.133, valid loss: 0.132, train acc: 0.516, valid acc: 0.522



Accuracy about 0.5, 好像不太妙啊...

Deep Neural Network(DNN) Tips



Deep Neural Network(DNN) Tips



Lab: lab_dnn_tutorial_practice.ipynb (15 minutes)

搜尋 "Do:" 就可以找到可能要修改的位置

lab filename	lab_tutorial_dnn_practice.ipynb
target	將loss function由MSE修改成cross entropy

Classification With Cross Entropy

- + learning_rate: 0.01
- + loss function: softmax cross entropy
- + optimizer: 基本款 => GradientDescentOptimizer
- + activation function: sigmoid

Deep Neural Network(DNN) Tips

Tensorflow Code(Loss Function)



```
with tf.variable_scope("loss"):  
    # loss function 改用 cross entropy  
    self.loss = tf.reduce_mean(tf.nn.softmax_cross_entropy_with_logits(labels=self.labels, logits=nets))
```

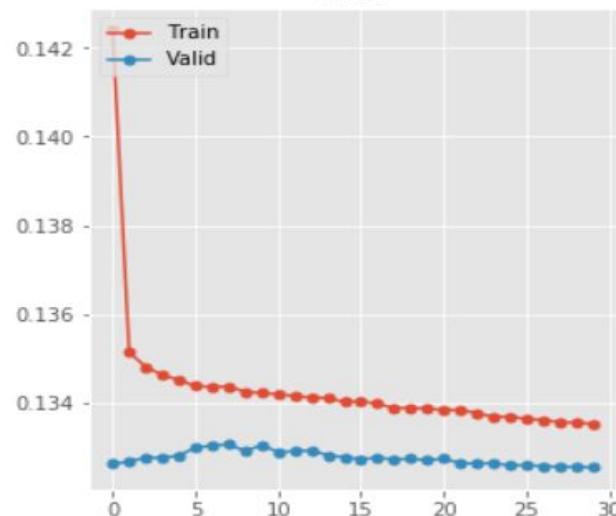
- ❑ loss function: `tf.nn.softmax_cross_entropy_with_logits`
 - ❑ 這個function會自動幫你的prediction加上softmax

Deep Neural Network(DNN) Tips

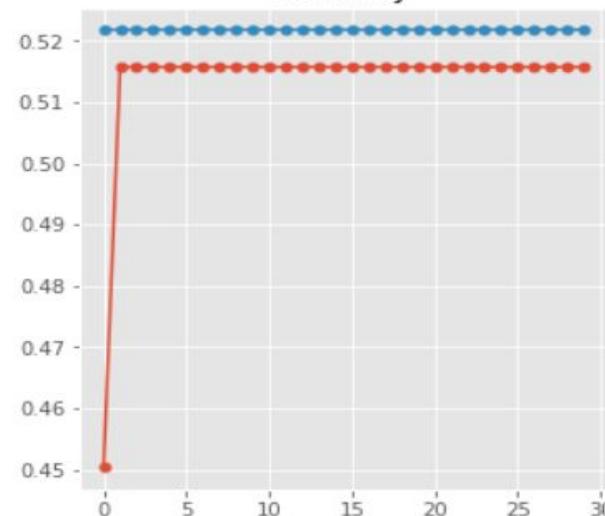
Result - MSE vs Cross Entropy



Loss



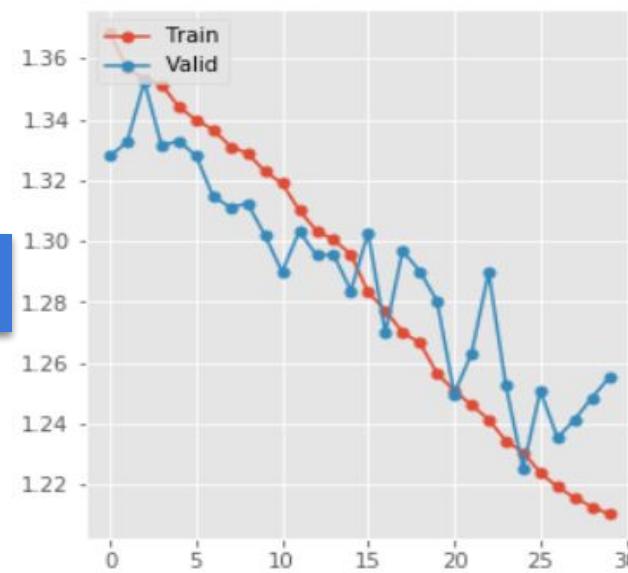
Accuracy



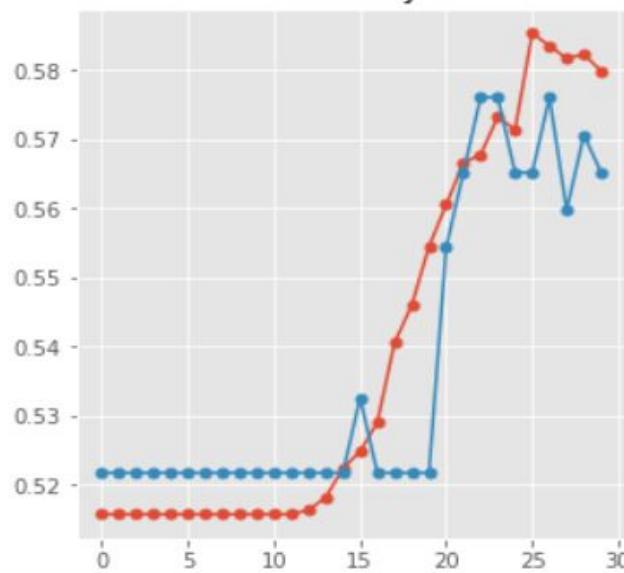
MSE

Cross Entropy

Loss

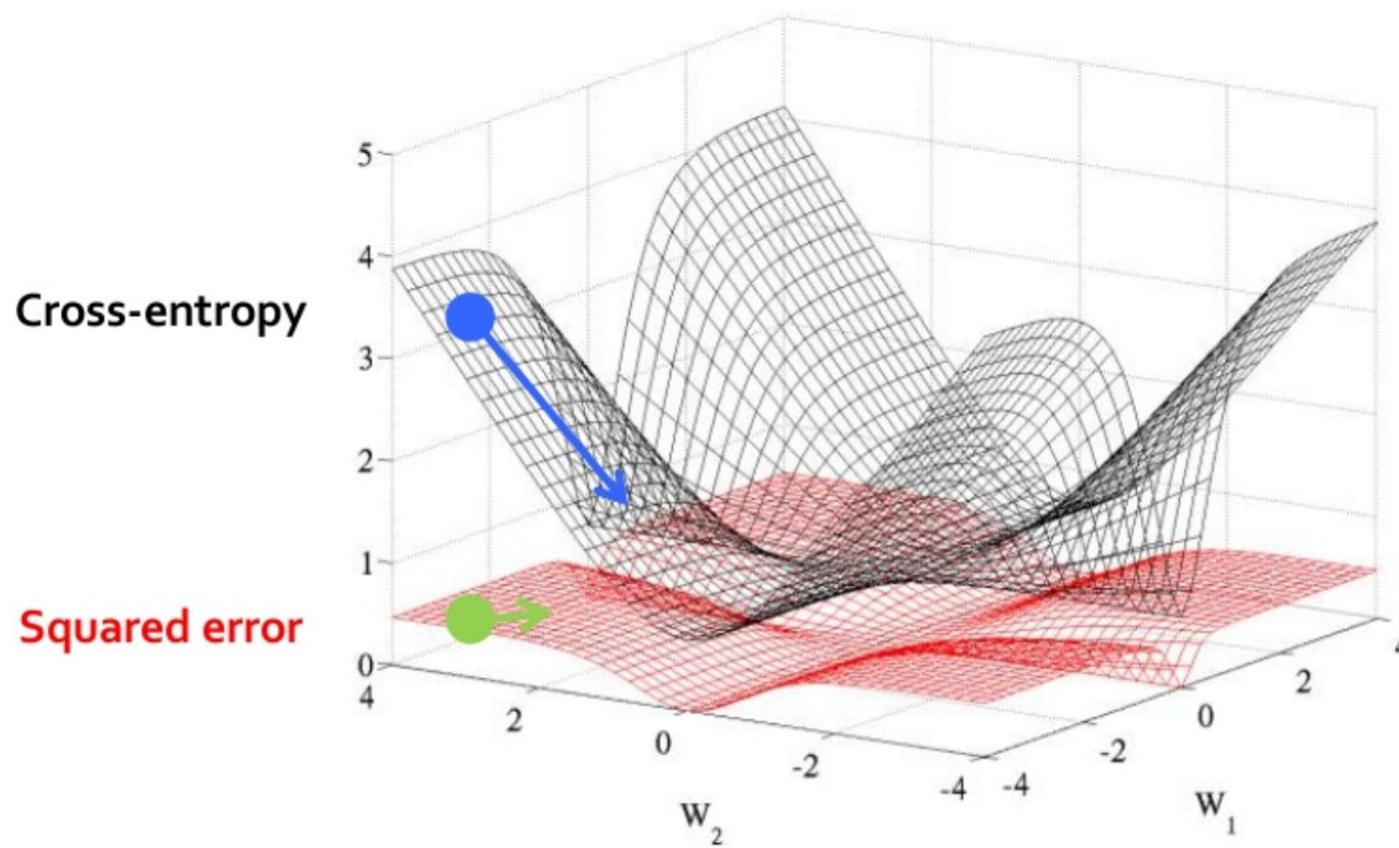


Accuracy



Deep Neural Network(DNN) Tips

Classification Loss Function: MSE or Cross Entropy



The error surface of logarithmic functions is steeper than that of quadratic functions.

Deep Neural Network(DNN) Tips

How to Select Loss function



- ❑ Classification常用cross entropy
 - ❑ 搭配softmax or sigmoid當作output layer的activation function
- ❑ Regression常用mean squared/absolute error
 - ❑ 其他還有...
 - ❑ MAPE: mean absolute percentage error
 - ❑ SMAPE: symmetric mean absolute percentage error

$$\frac{100}{n} \sum_i^n \frac{|actual_i - prediction_i|}{actual_i}$$

Deep Neural Network(DNN) Tips



Lab: lab_dnn_tutorial_practice.ipynb (5 ~ 8 minutes)

搜尋 "Do:" 就可以找到可能要修改的位置

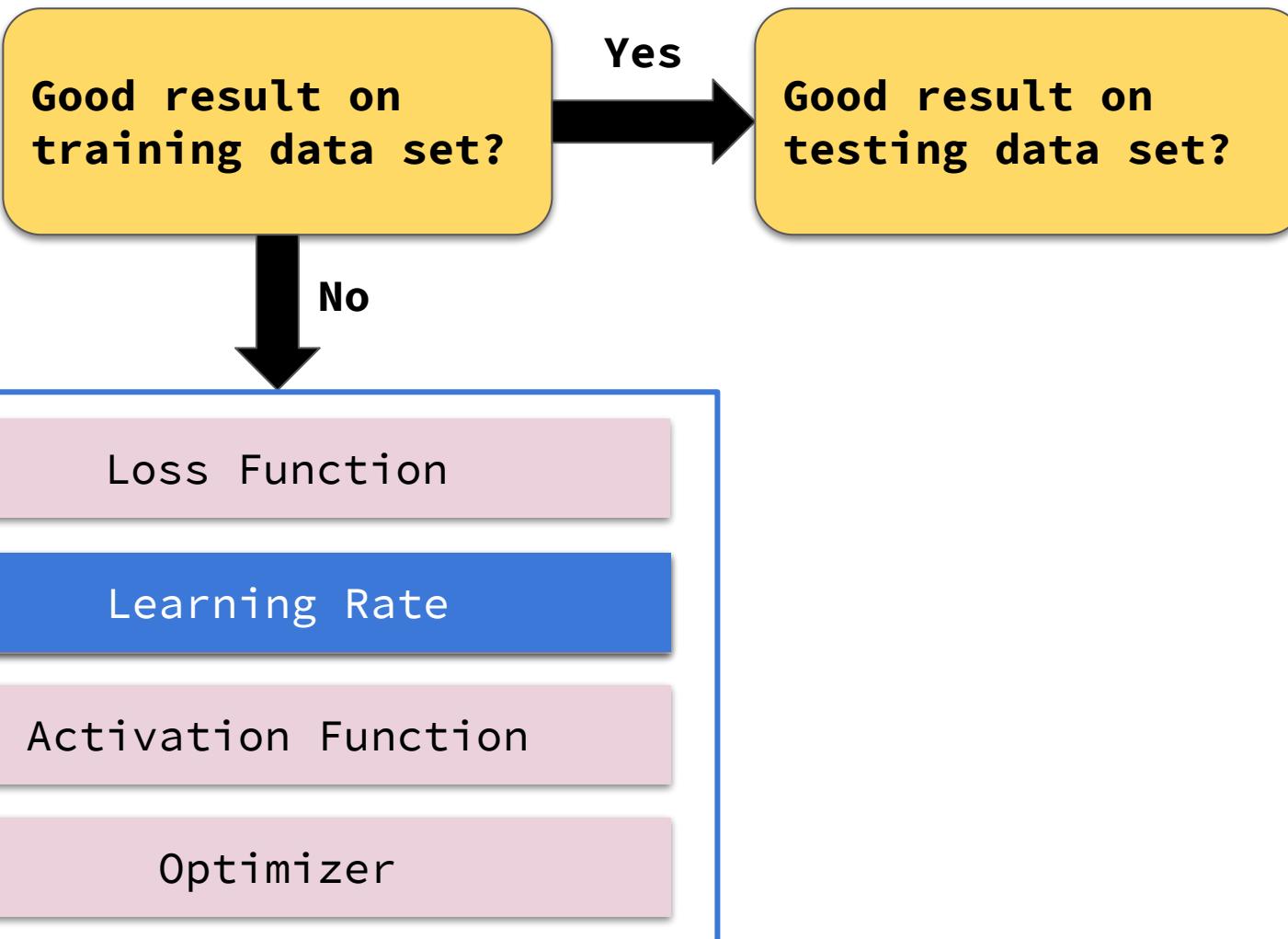
lab filename	<code>lab_tutorial_dnn_practice.ipynb</code>
target	嘗試不同的learning rate: [0.1, 0.01, 0.001]

Try Learning Rate!

- + learning_rate: 0.1, 0.01, 0.001
- + loss function: softmax cross entropy
- + optimizer: 基本款 => GradientDescentOptimizer
- + activation function: sigmoid

Deep Neural Network(DNN) Tips

Tune Your Learning Rate



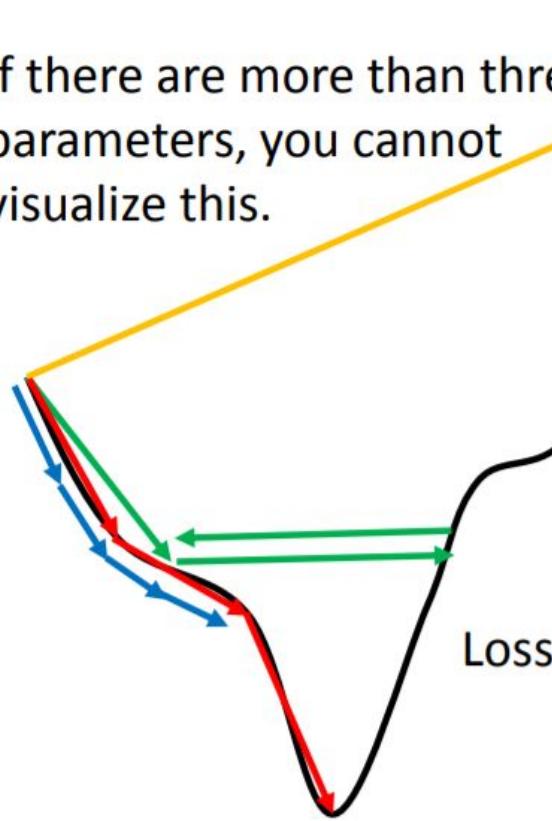
Deep Neural Network(DNN) Tips

Tune Your Learning Rate

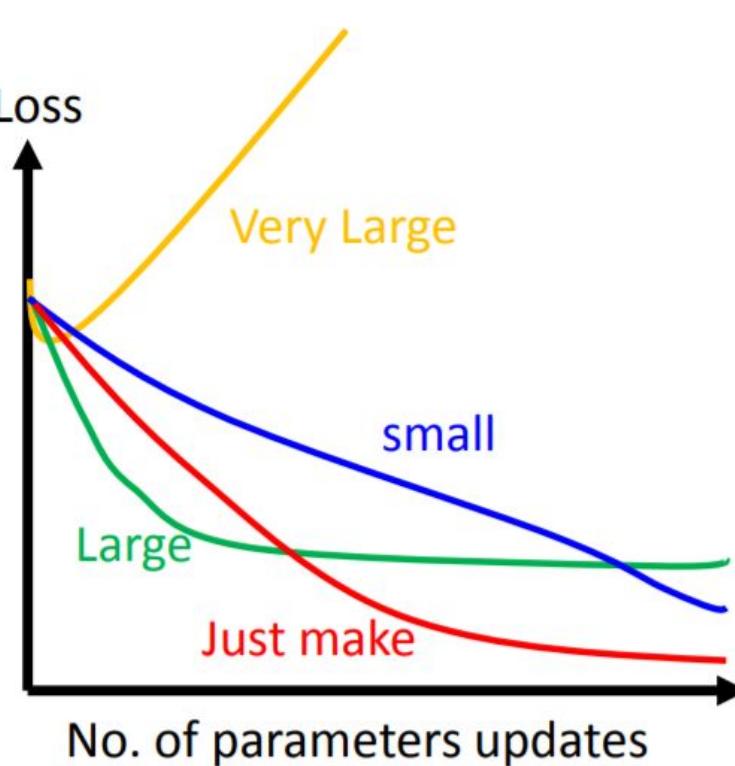


Learning Rate

If there are more than three parameters, you cannot visualize this.



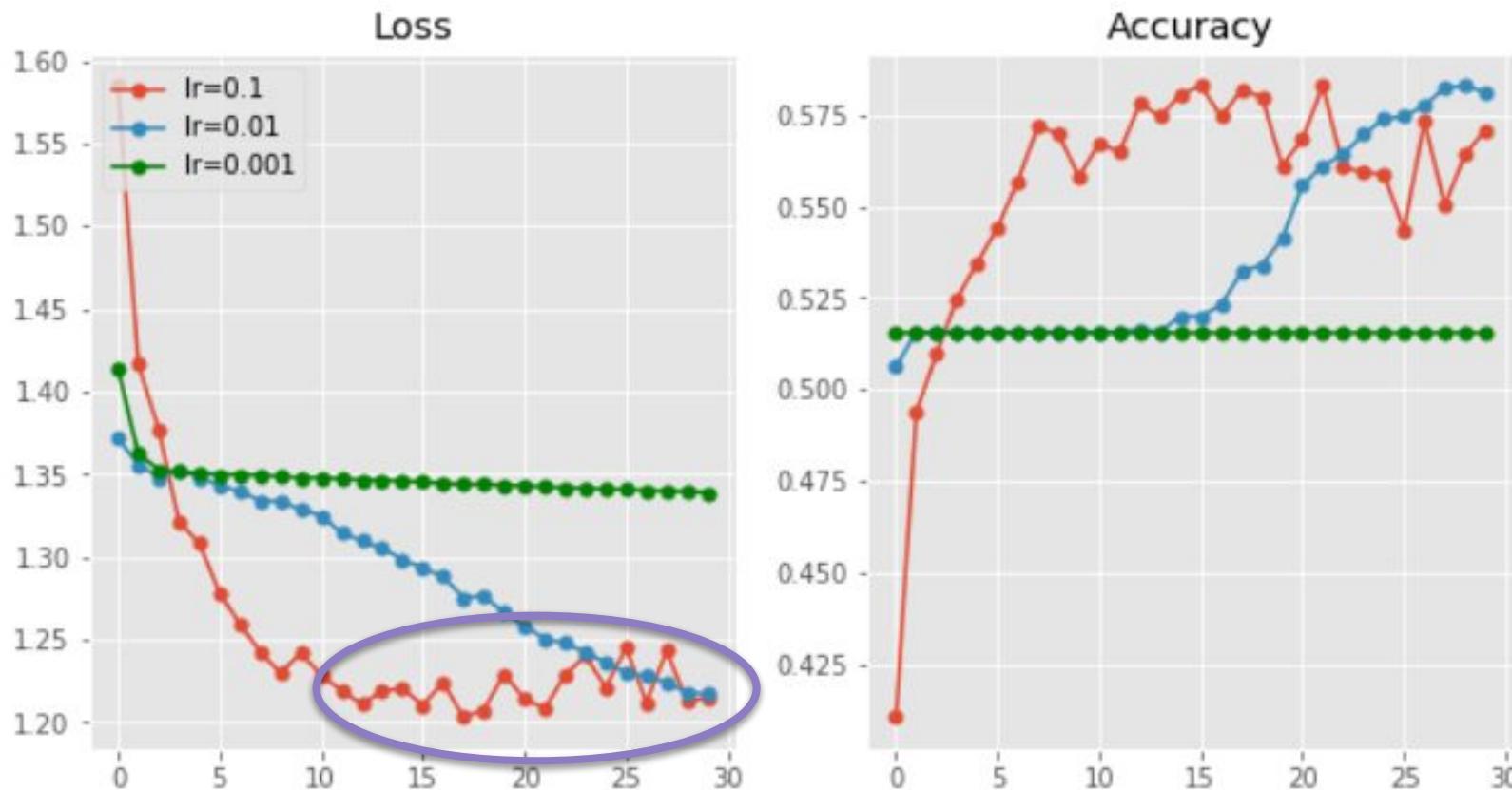
$\theta^i = \theta^{i-1} - \eta \nabla L(\theta^{i-1})$
Set the learning rate η carefully



But you can always visualize this.

Deep Neural Network(DNN) Tips

Tune Your Learning Rate



觀察loss, 這樣的震盪比表示learning rate可能太大, 以這個狀況, 0.01是最好的選擇

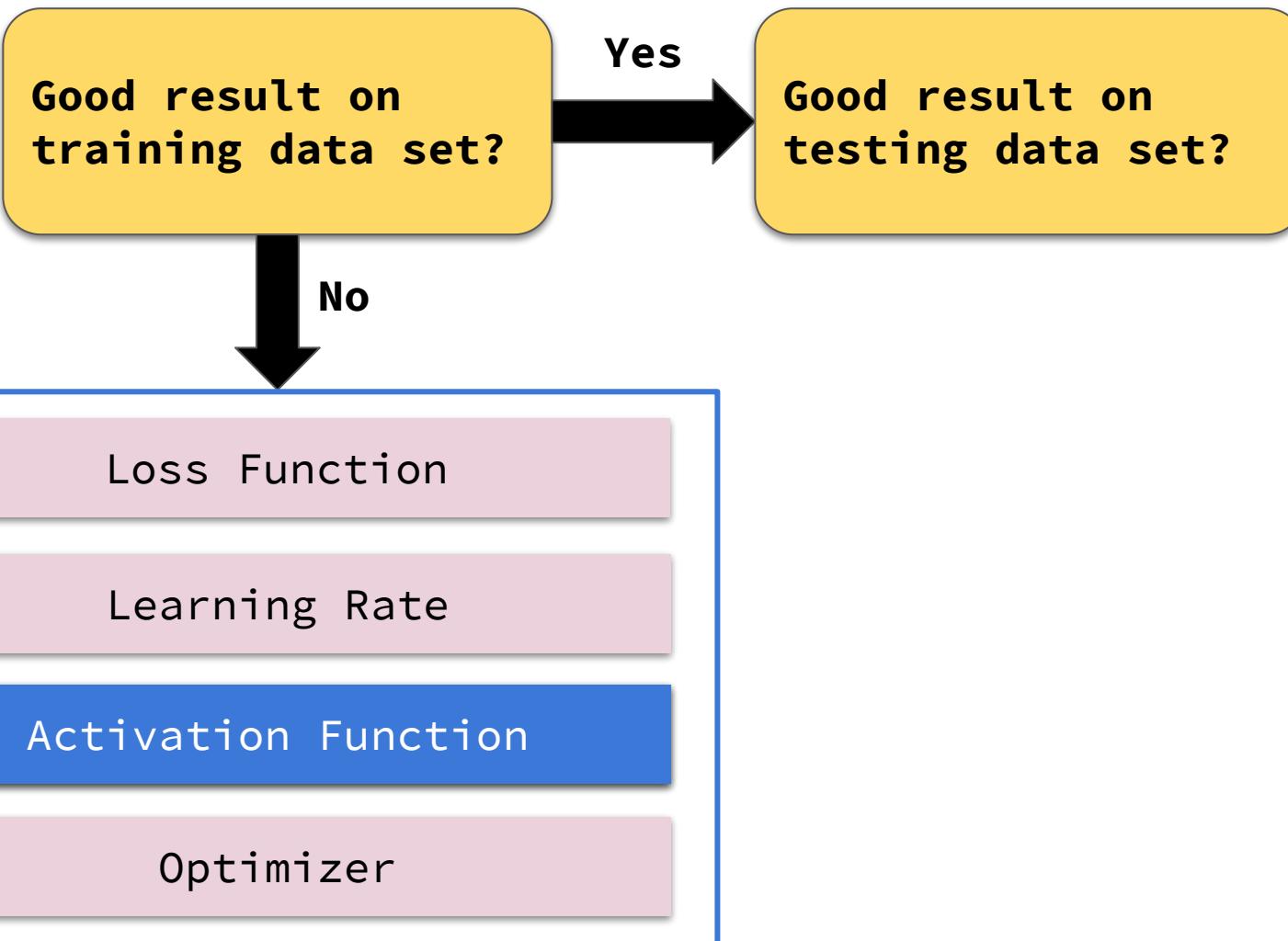
Tune Your Learning Rate



- ❑ 大多要試試看才知道，通常不會大於0.1
- ❑ 調整的步伐大概是一次5 ~ 10倍
 - ❑ $0.1 \Rightarrow 0.01 \Rightarrow 0.001$
 - ❑ $0.001 \Rightarrow 0.01 \Rightarrow 0.1 \Rightarrow 0.5$
 - ❑ ~~$0.01 \Rightarrow 0.012 \Rightarrow 0.015 \Rightarrow$ 最高0.87分不能再高~~

Deep Neural Network(DNN) Tips

Choose Your Activation Function

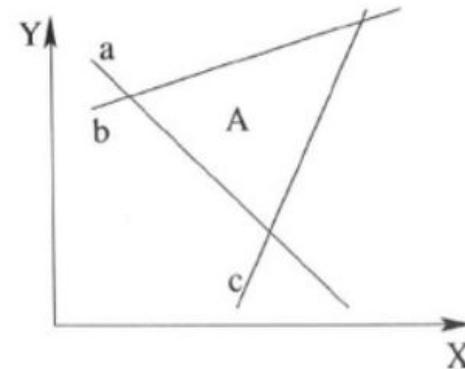
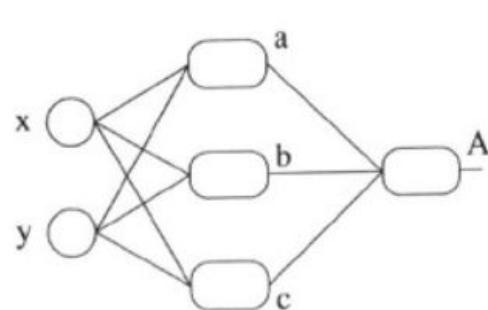


Deep Neural Network(DNN) Tips

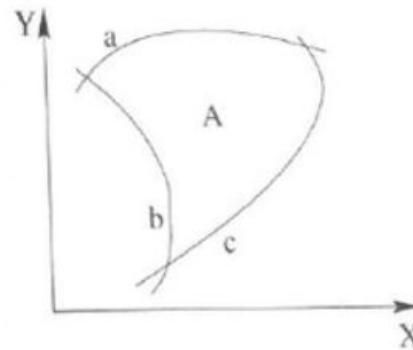
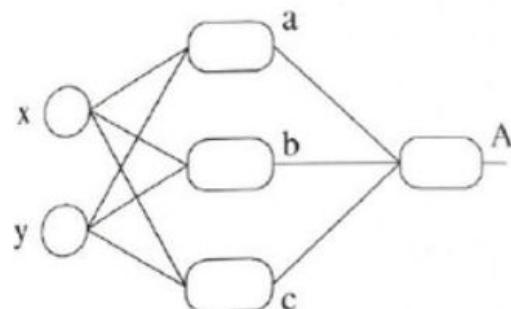
Activation Function的初衷



- 把你的Neural network變成non-linear, 加強fit能力



with step activation function



with sigmoid activation function

Deep Neural Network(DNN) Tips



傳統的Activation Function

Sigmoid

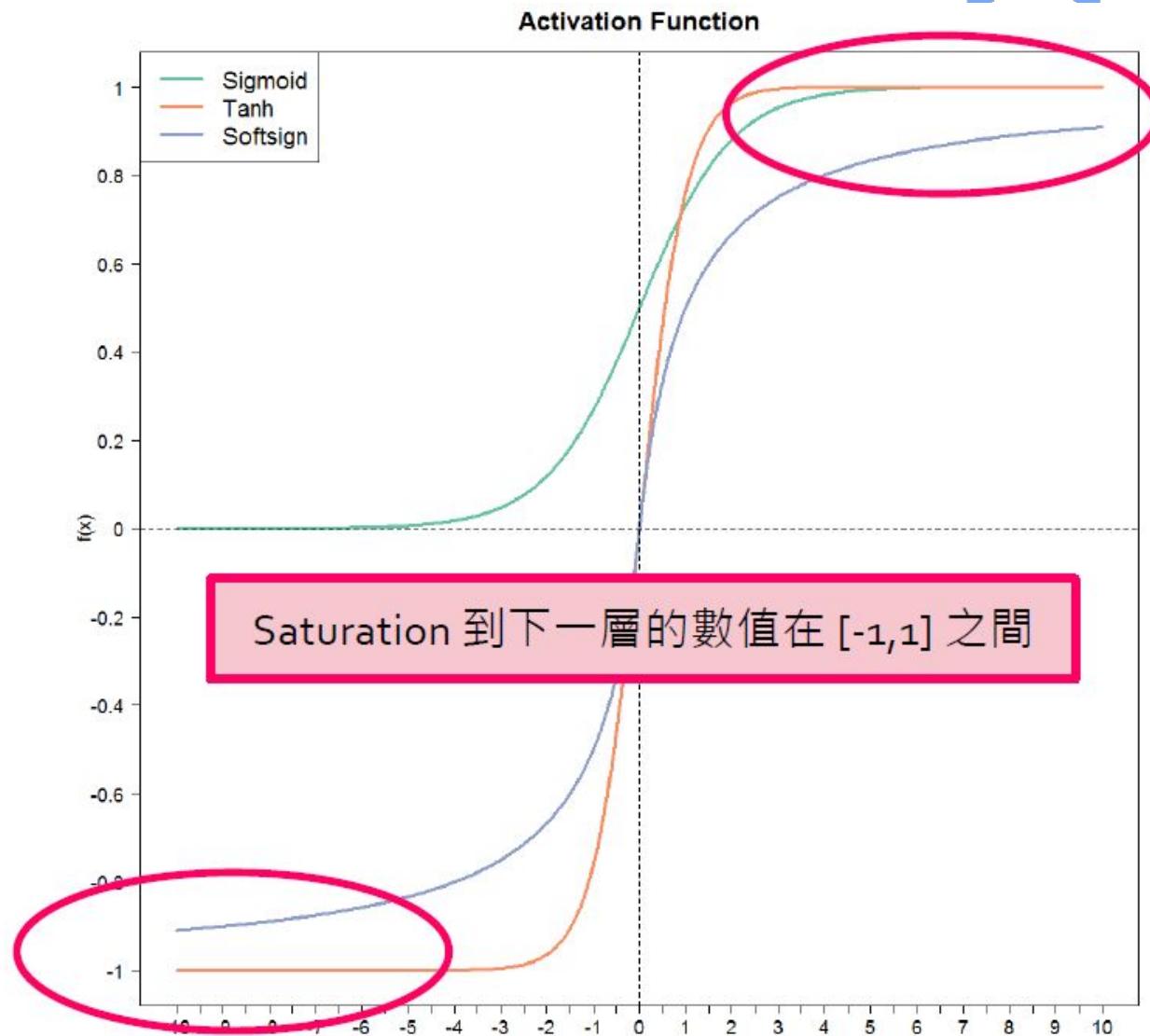
$$f(x) = \frac{1}{(1+e^{-x})}$$

Tanh

$$f(x) = \frac{(1-e^{-2x})}{(1+e^{-2x})}$$

Softsign

$$f(x) = \frac{x}{(1+|x|)}$$



Deep Neural Network(DNN) Tips

傳統的Activation Function



Sigmoid

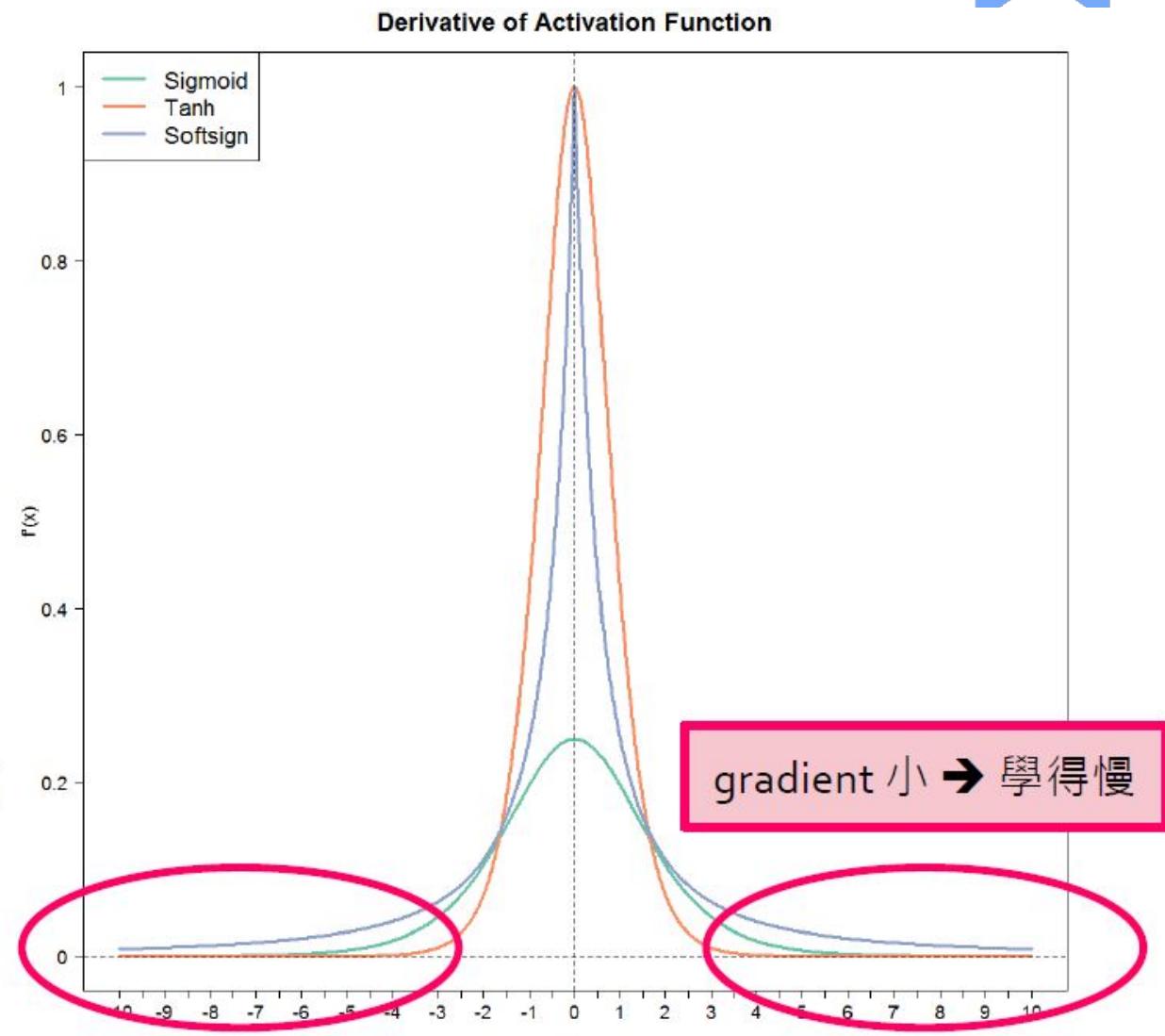
$$df/dx = \frac{e^{-x}}{(1+e^{-x})^2}$$

Tanh

$$df/dx = 1 - f(x)^2$$

Softsign

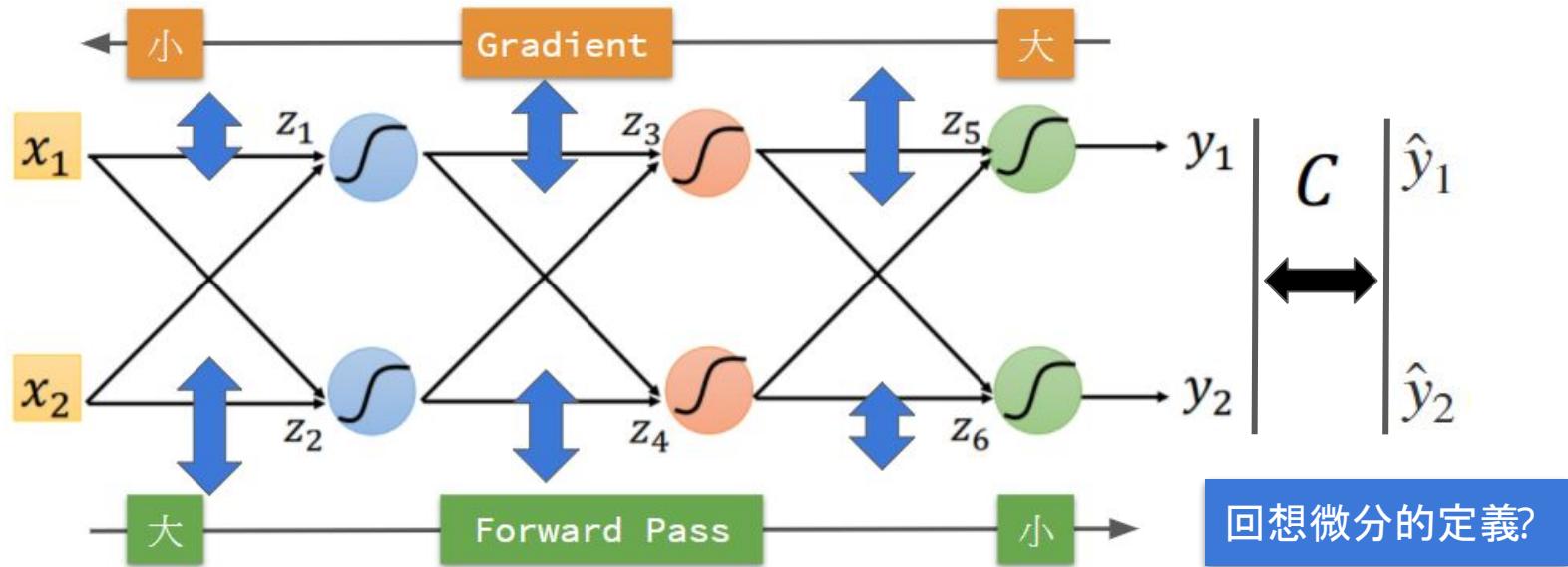
$$df/dx = \frac{1}{(1+|x|)^2}$$





傳統的Activation Function

- ❑ Vanishing gradient problem
 - ❑ 原因: 每一層的input被壓縮到一個相對很小的output range



- ❑ 結果: Gradient小無法有效地學習
- ❑ Sigmoid, Tanh, Softsign都有這樣的特性
- ❑ 特別不適用於深的深度學習模型

Deep Neural Network(DNN) Tips



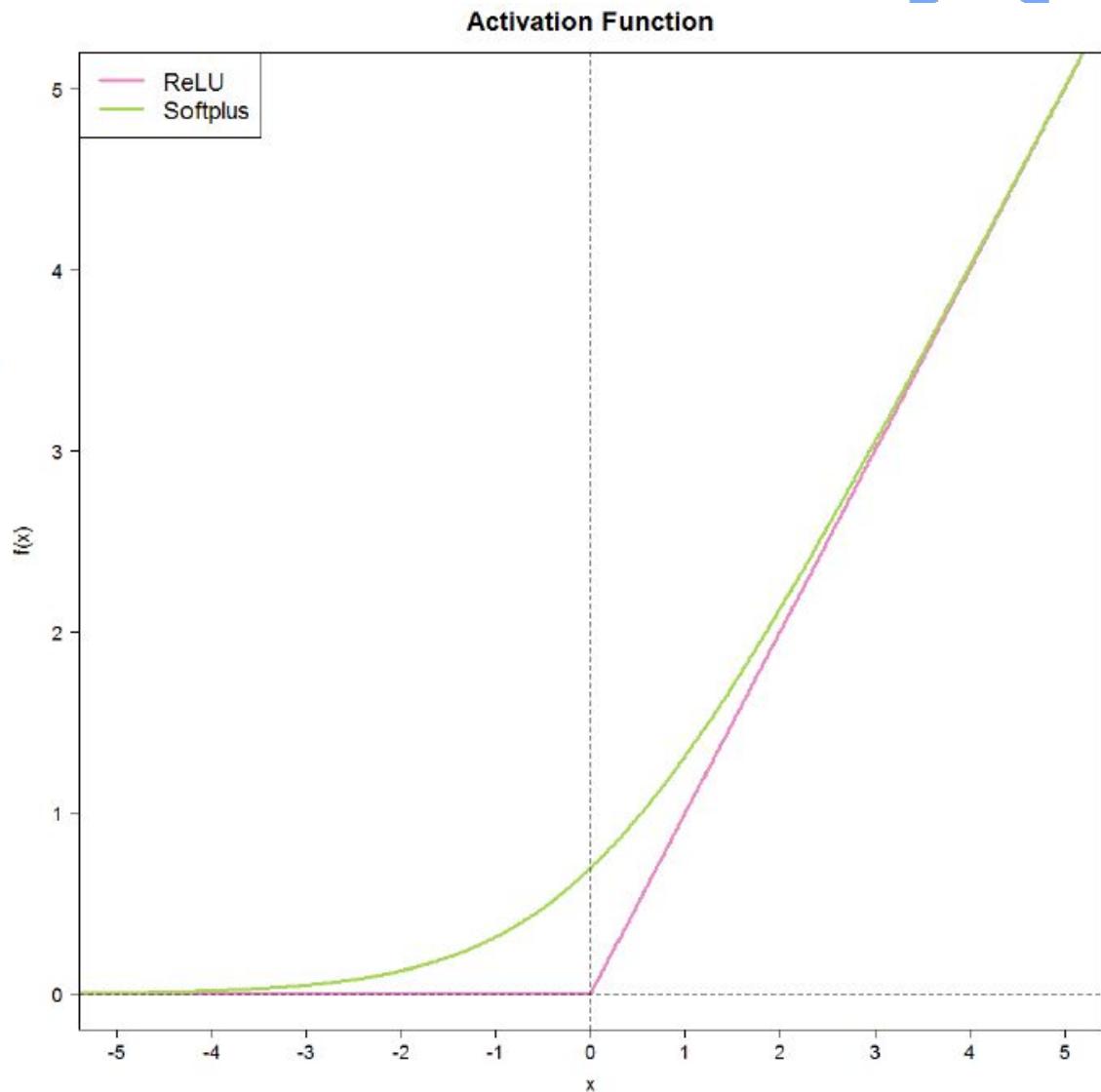
比較潮的Activation Function

□ ReLU

- $f(x) = \max(0, x)$
- $df/dx = 1 \text{ if } x > 0,$
 0 otherwise.

□ Softplus

- $f(x) = \ln(1+e^x)$
- $df/dx = e^x/(1+e^x)$

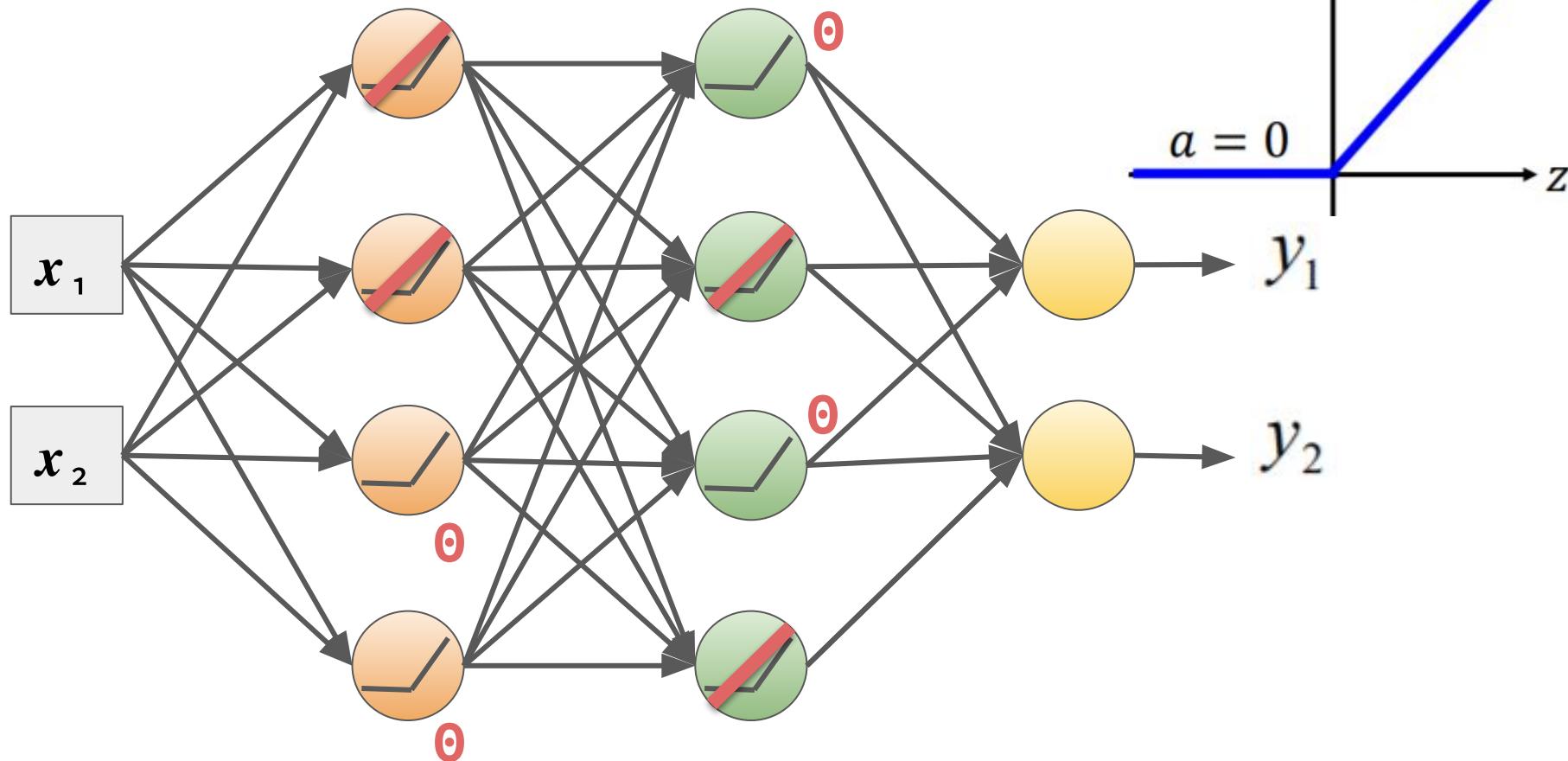


Deep Neural Network(DNN) Tips



比較潮的Activation Function

ReLU: Rectified Linear Unit(線性整流函數)

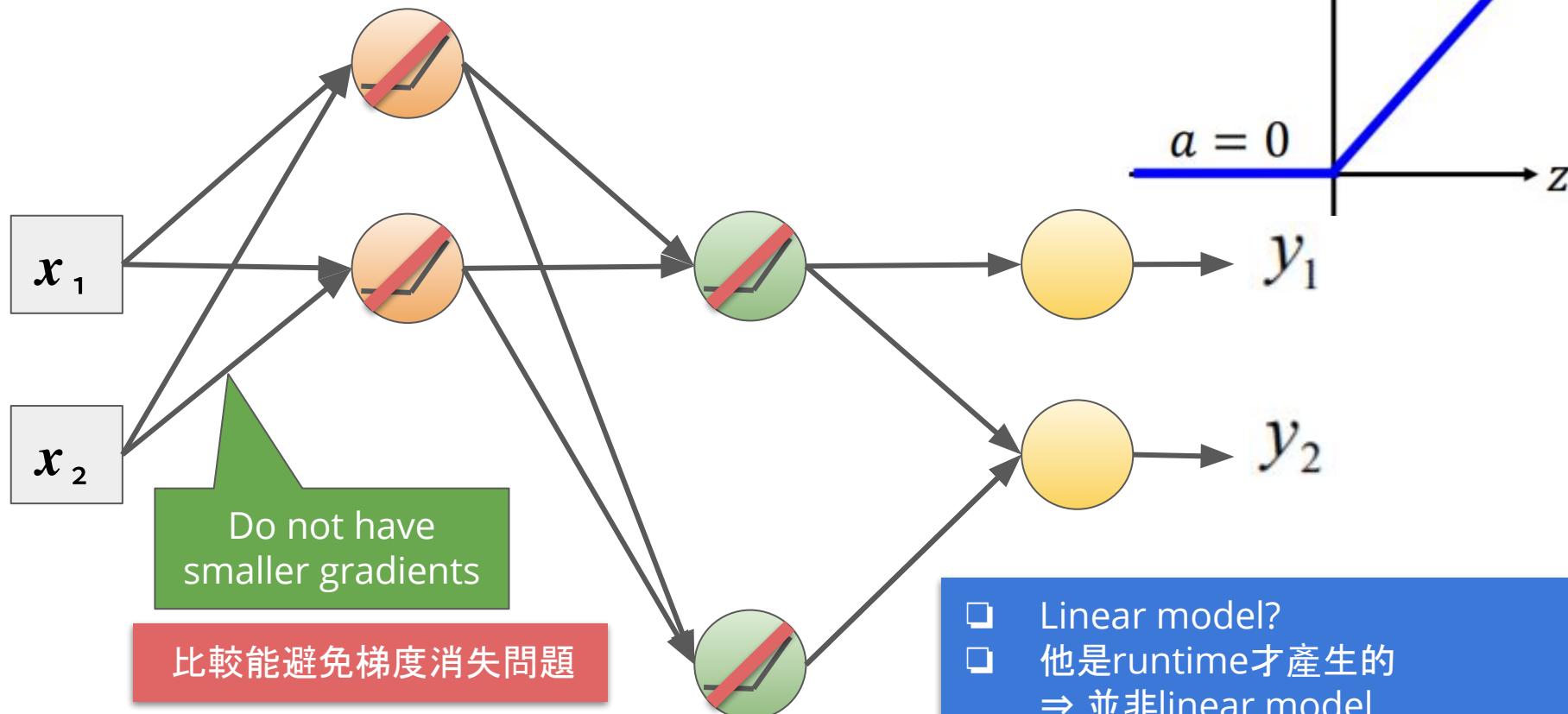


Deep Neural Network(DNN) Tips



比較潮的Activation Function

ReLU: Rectified Linear Unit(線性整流函數)

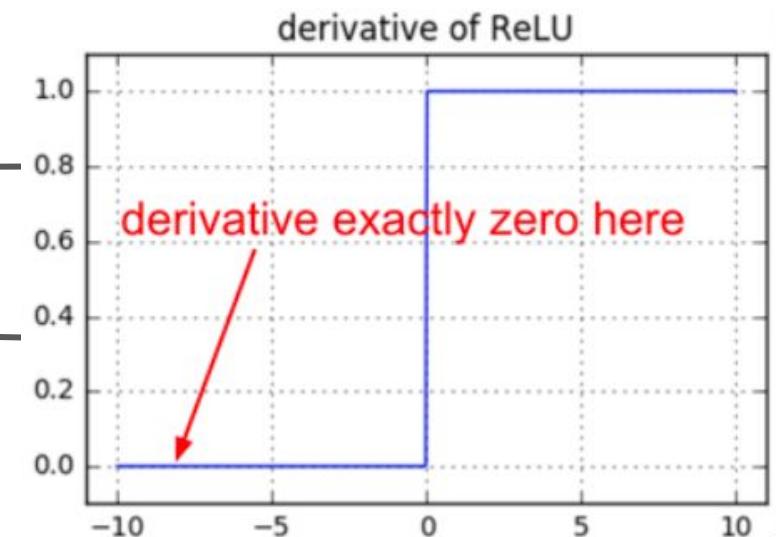
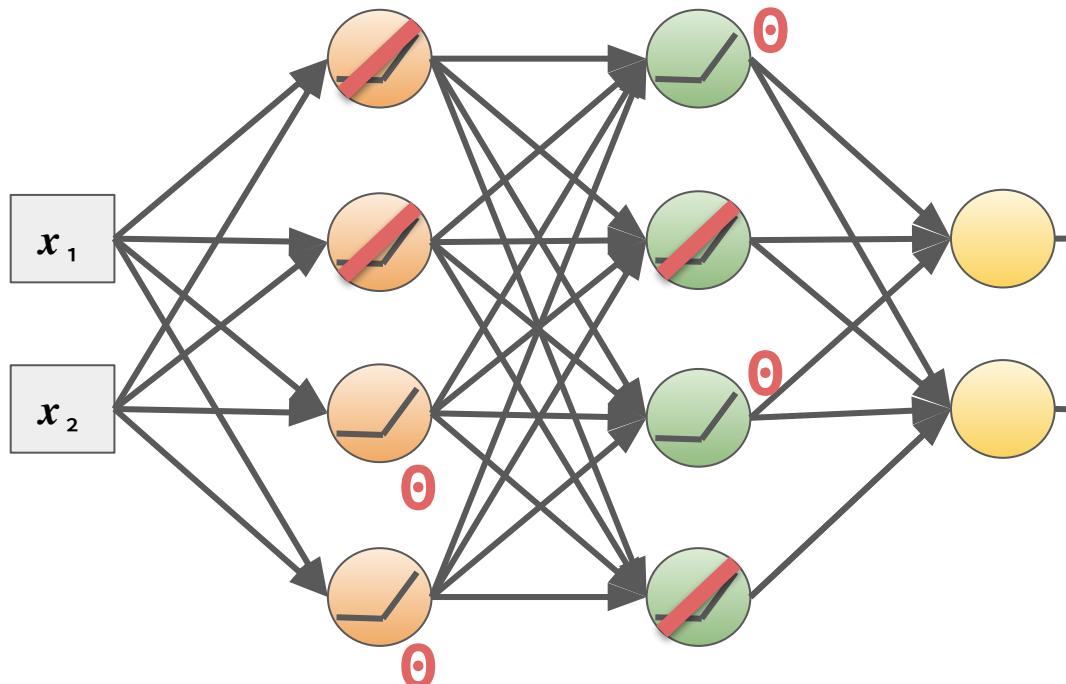


Deep Neural Network(DNN) Tips



比較潮的Activation Function

ReLU: 還是有缺點...



❑ Dying ReLU problem

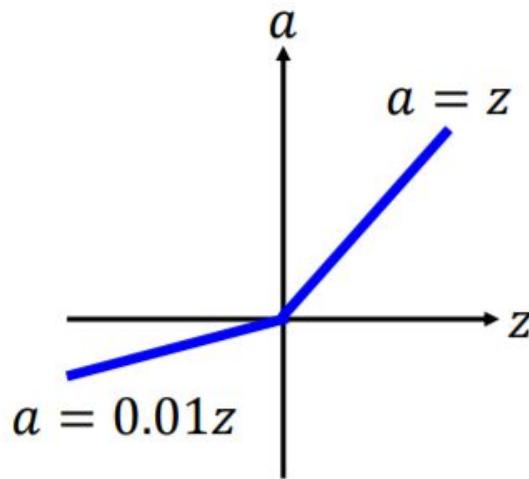
Deep Neural Network(DNN) Tips

比較潮的Activation Function

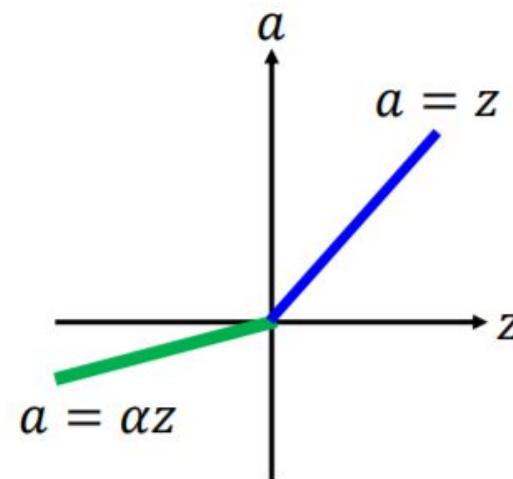
ReLU - variant



Leaky ReLU



Parametric ReLU



實際上Dying ReLU problem:
 適當的 weight init
 適當的 learning rate

α also learned by gradient descent

Deep Neural Network(DNN) Tips



Lab: lab_dnn_tutorial_practice.ipynb (5 ~ 8 minutes)

搜尋 "Do:" 就可以找到可能要修改的位置

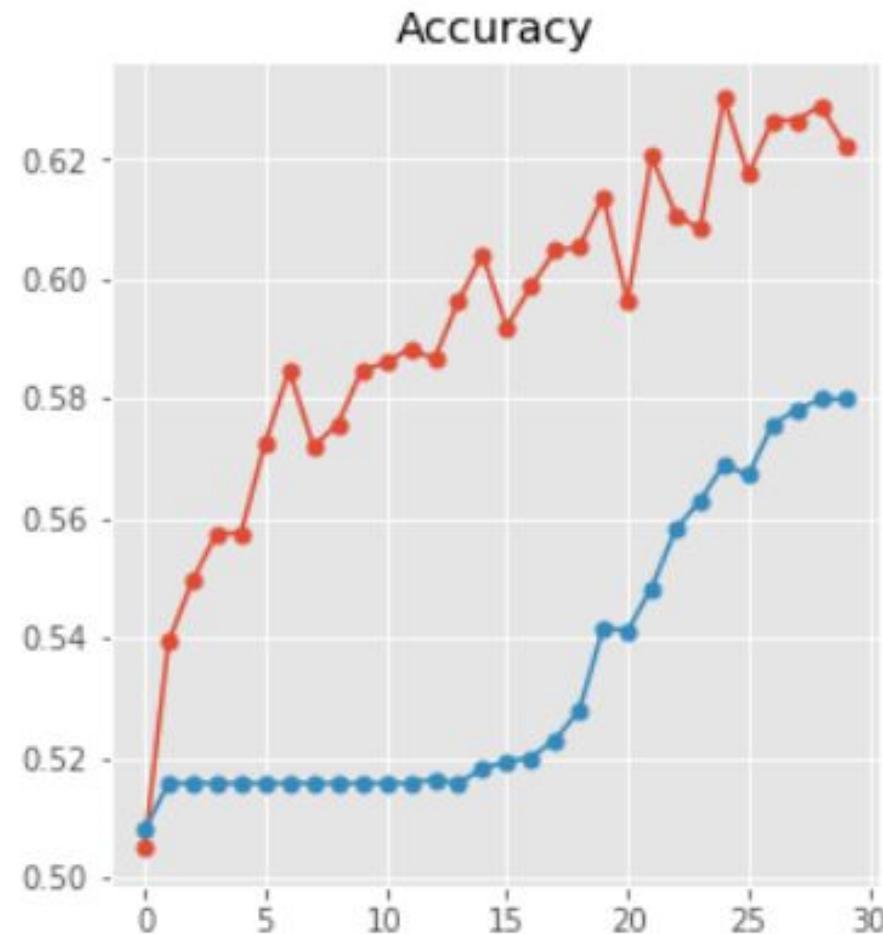
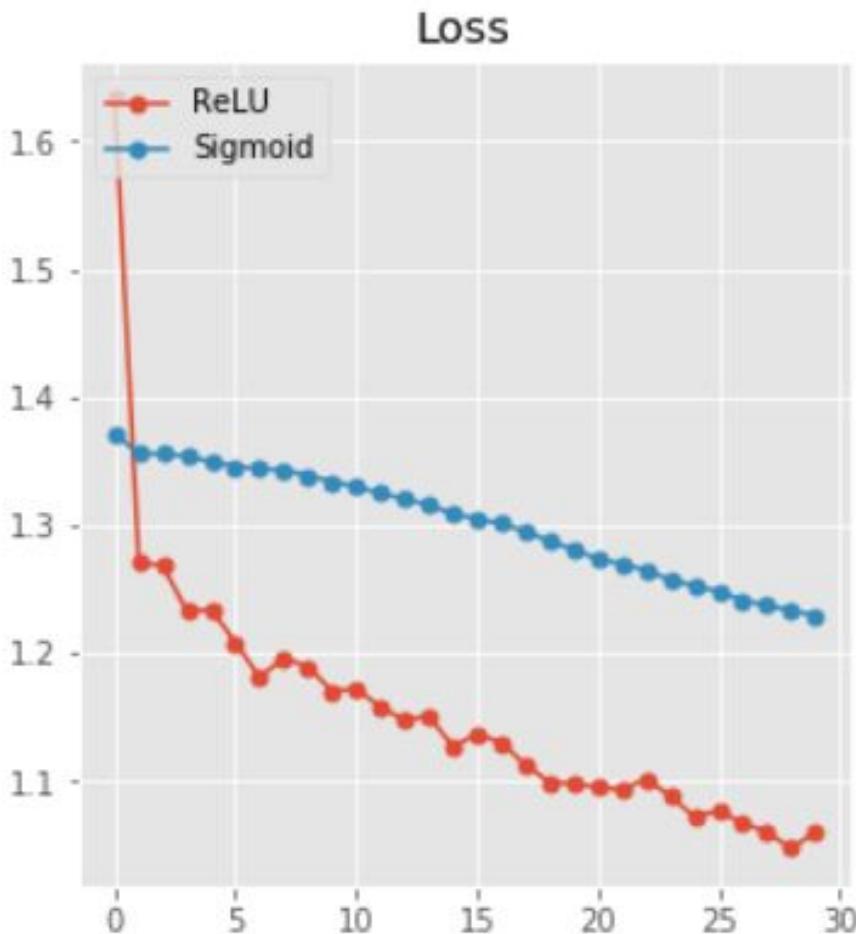
lab filename	lab_tutorial_dnn_practice.ipynb
target	目前是Sigmoid, 請嘗試不同的activation function <input type="checkbox"/> ReLU、Leaky ReLU <input type="checkbox"/> Softplus <input type="checkbox"/> Tanh

Try Activation Function

- + learning_rate: 0.01
- + loss function: softmax cross entropy
- + optimizer: 基本款 => GradientDescentOptimizer
- + activation function: sigmoid or softplus or relu

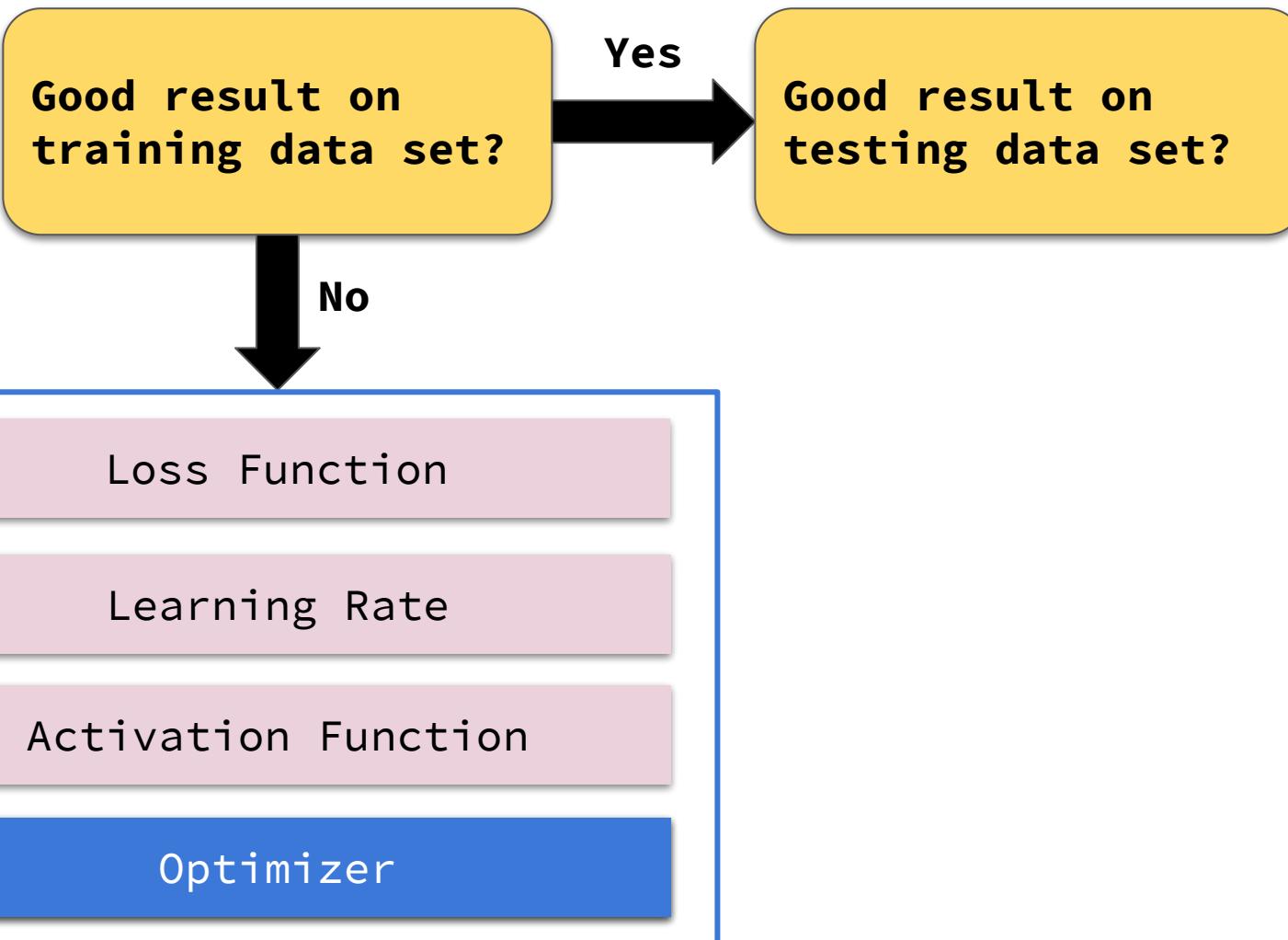
Deep Neural Network(DNN) Tips

ReLU vs Sigmoid



Deep Neural Network(DNN) Tips

Choose Optimizer



Deep Neural Network(DNN) Tips



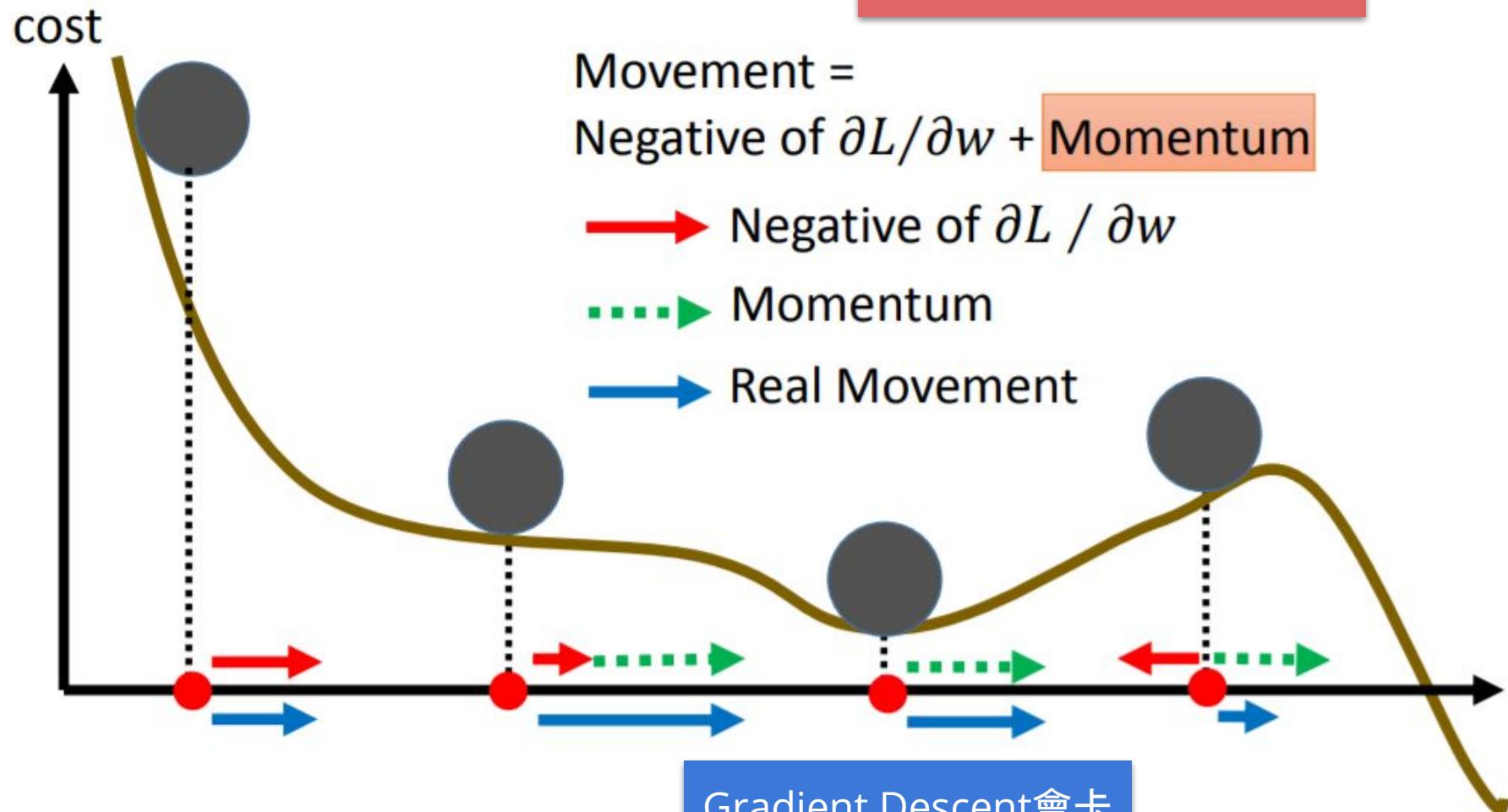
Optimizer

- ❑ Gradient Descent
- ❑ Momentum
- ❑ Adagrad – Adaptive Learning Rate
- ❑ RMSprop – Similar with Adagrad
- ❑ Adam – Similar with RMSprop + Momentum

Deep Neural Network(DNN) Tips

Optimizer - Momentum

當然沒辦法保證會跑到
global minimum

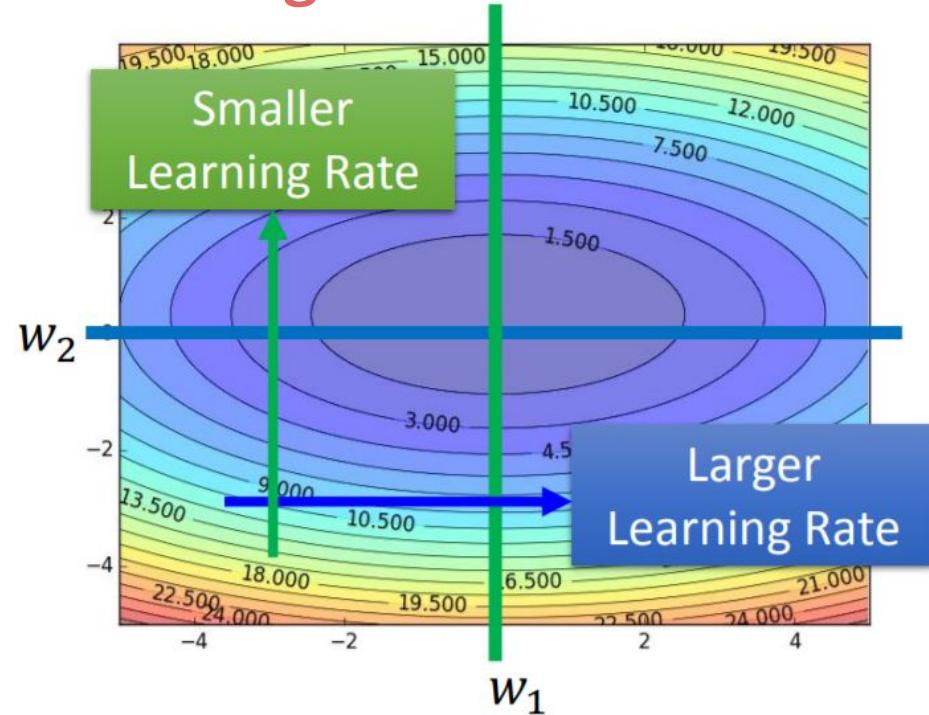


Gradient Descent會卡
在這裡(gradient = 0)

Optimizer - Adaptive Learning Rate



- ❑ 認為一開始離目的很遠, 所以用比較大的learning rate
- ❑ 在train好幾個epoch之後, 離目的越來越近, 需要較小的learning rate
- ❑ learning rate需要在training runtime調整
- ❑ 因材施教:每個參數都有不同的learning rate



Deep Neural Network(DNN) Tips

Optimizer - Adagrad



$$\eta^t = \frac{\eta}{\sqrt{t + 1}} \quad g^t = \frac{\partial L(\theta^t)}{\partial w}$$

Vanilla Gradient descent

$$w^{t+1} \leftarrow w^t - \eta^t g^t$$

w is one parameters

Adagrad

$$w^{t+1} \leftarrow w^t - \frac{\eta^t}{\sigma^t} g^t$$

σ^t : **root mean square** of the previous derivatives of parameter w

Deep Neural Network(DNN) Tips

Optimizer - Adagrad

□ 直覺的理解



特別大

g^0	g^1	g^2	g^3	g^4	...
0.001	0.001	0.0002	0.0087	0.1	...

g^0	g^1	g^2	g^3	g^4	...
8.7	10.87	18.7	87.7	0.1	...

特別小

$$w^{t+1} \leftarrow w^t - \frac{\eta}{\sqrt{\sum_{i=0}^t (g^i)^2}} g^t$$

- 老馬識途: 參考之前的經驗修正現在的步伐
- 不完全相信當下的gradient
- 缺點: learning rate會一直遞減, 越train越慢...

Deep Neural Network(DNN) Tips



Optimizer - RMSProp

Adagrad

$$w^{t+1} = w^t - \frac{\eta}{\sigma^t} g^t$$

$$\sigma^t = \sqrt{(g^0)^2 + \dots + (g^t)^2}$$

RMSProp

$$w^{t+1} = w^t - \frac{\eta}{\sigma^t} g^t$$

$$\sigma^t = \sqrt{\alpha(\sigma^{t-1})^2 + (1 - \alpha)(g^t)^2}$$

α 小: 代表相信當前gradient

α 大: 代表相信之前的gradient

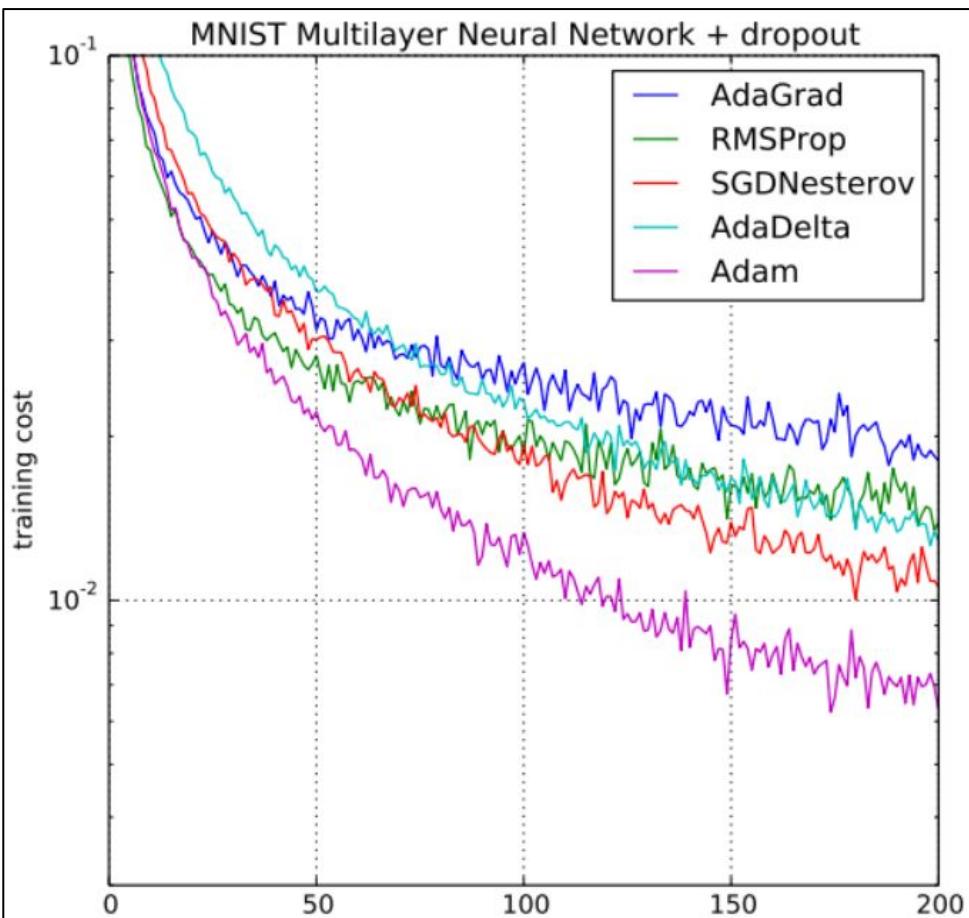
- 另一種參考過去gradient的方式

Deep Neural Network(DNN) Tips

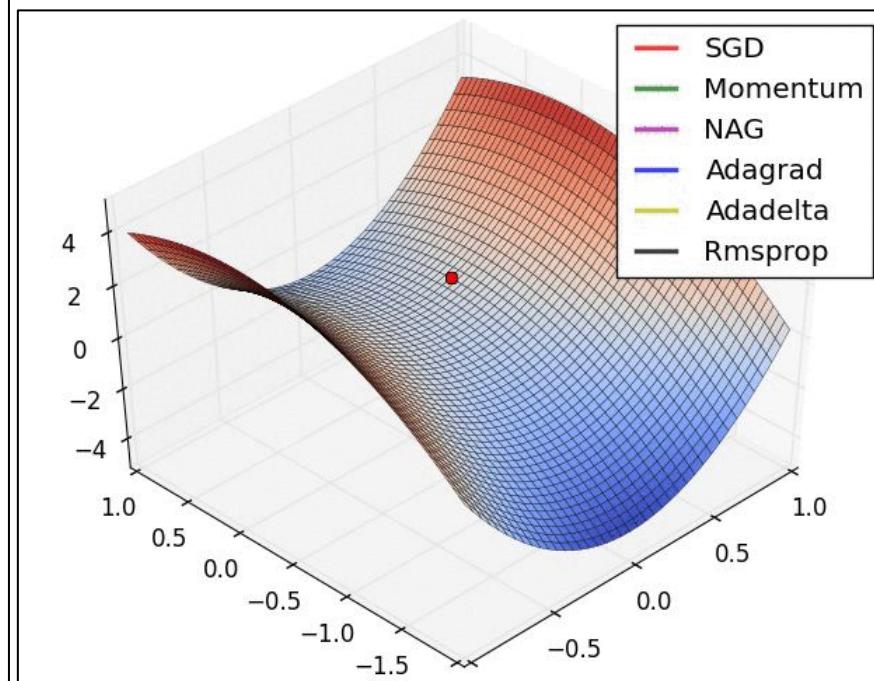


Optimizer - Adam

- 綜合 RMSProp + Momentum 的優點
- 個人建議起手式, 先從Adam開始嘗試



實際上Optimizer的效果還是會跟資料有關，沒有絕對要用什麼工具的規則！



Deep Neural Network(DNN) Tips



Lab: lab_dnn_tutorial_practice.ipynb (10 ~ 15 minutes)

搜尋 "Do:" 就可以找到可能要修改的位置

lab filename	lab_tutorial_dnn_practice.ipynb
target	使用不同的Optimizer: <input type="checkbox"/> tf.train.AdamOptimizer <input type="checkbox"/> tf.train.AdagradOptimizer <input type="checkbox"/> tf.train.MomentumOptimizer(momentum=0.9) <input type="checkbox"/> tf.train.RMSPropOptimizer

Try Different Optimizer

- + learning_rate: 0.01
- + loss function: softmax cross entropy
- + optimizer: GradientDescentOptimizer or AdamOptimizer
- + activation function: softplus

Deep Neural Network(DNN) Tips

Result: Gradient Descent vs Adam



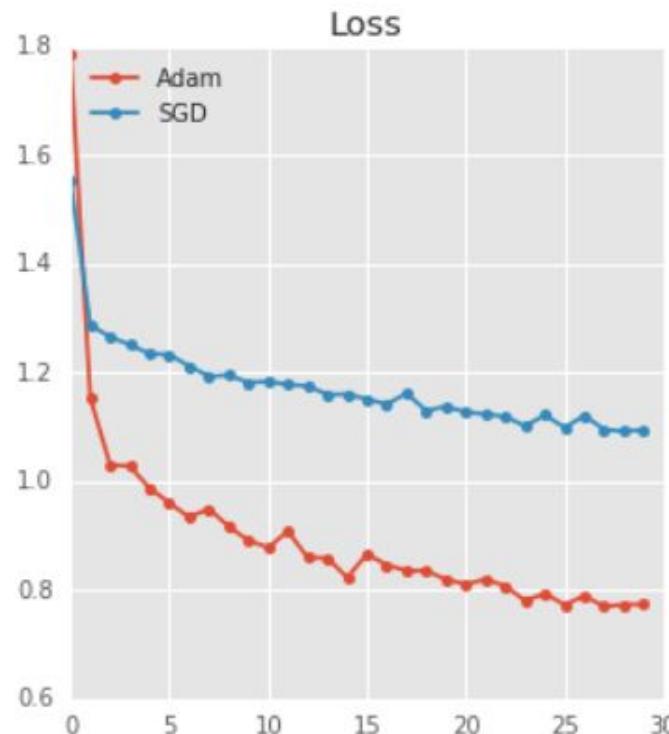
model with Adam optimizer!

30/30. train loss: 0.773, valid loss: 0.975, train acc: 0.732, valid acc: 0.696

model with SGD optimizer!

30/30. train loss: 1.095, valid loss: 1.434, train acc: 0.605, valid acc: 0.484

<matplotlib.text.Text at 0x7f2ca857d7f0>



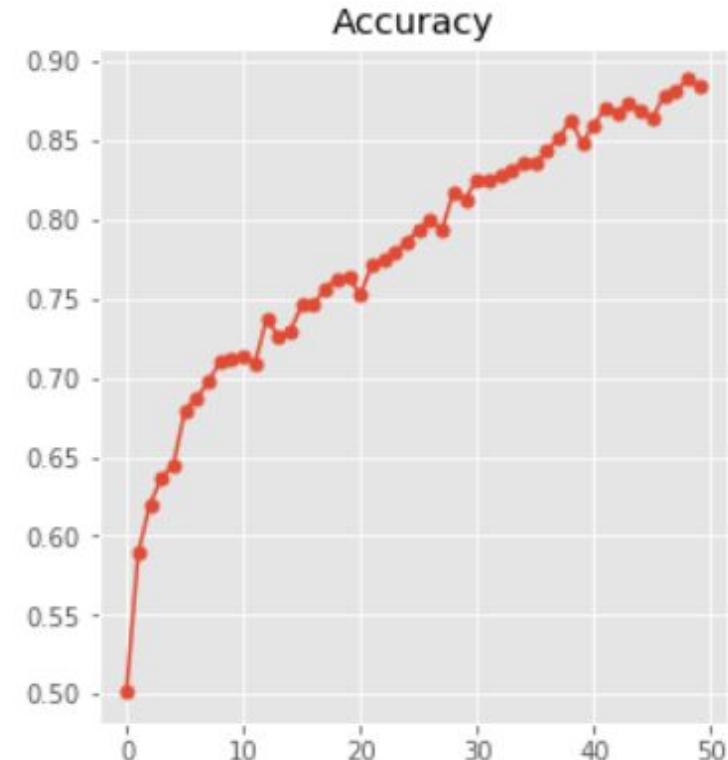
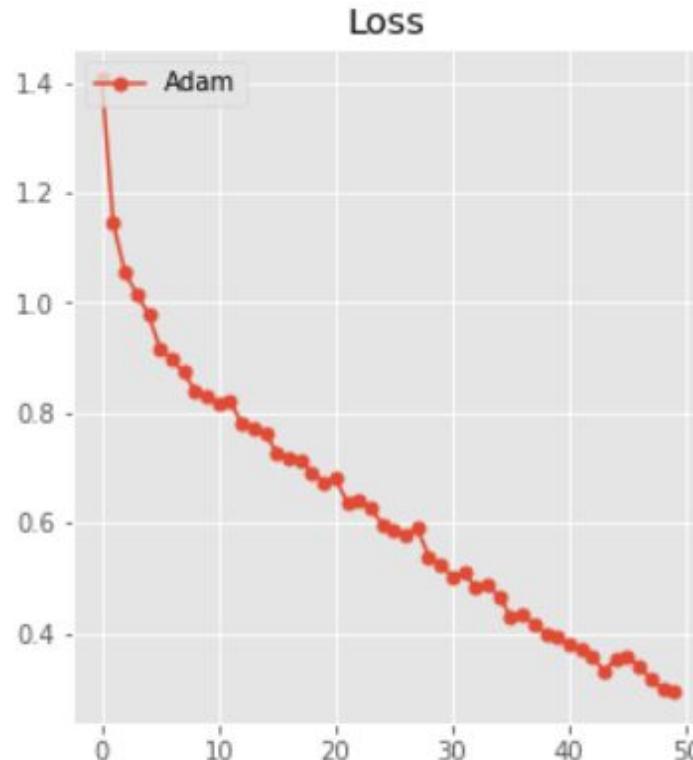
Deep Neural Network(DNN) Tips

Current Best Model Configuration



Loss function	cross entropy
Activation function	relu + softmax
Optimizer	Adam

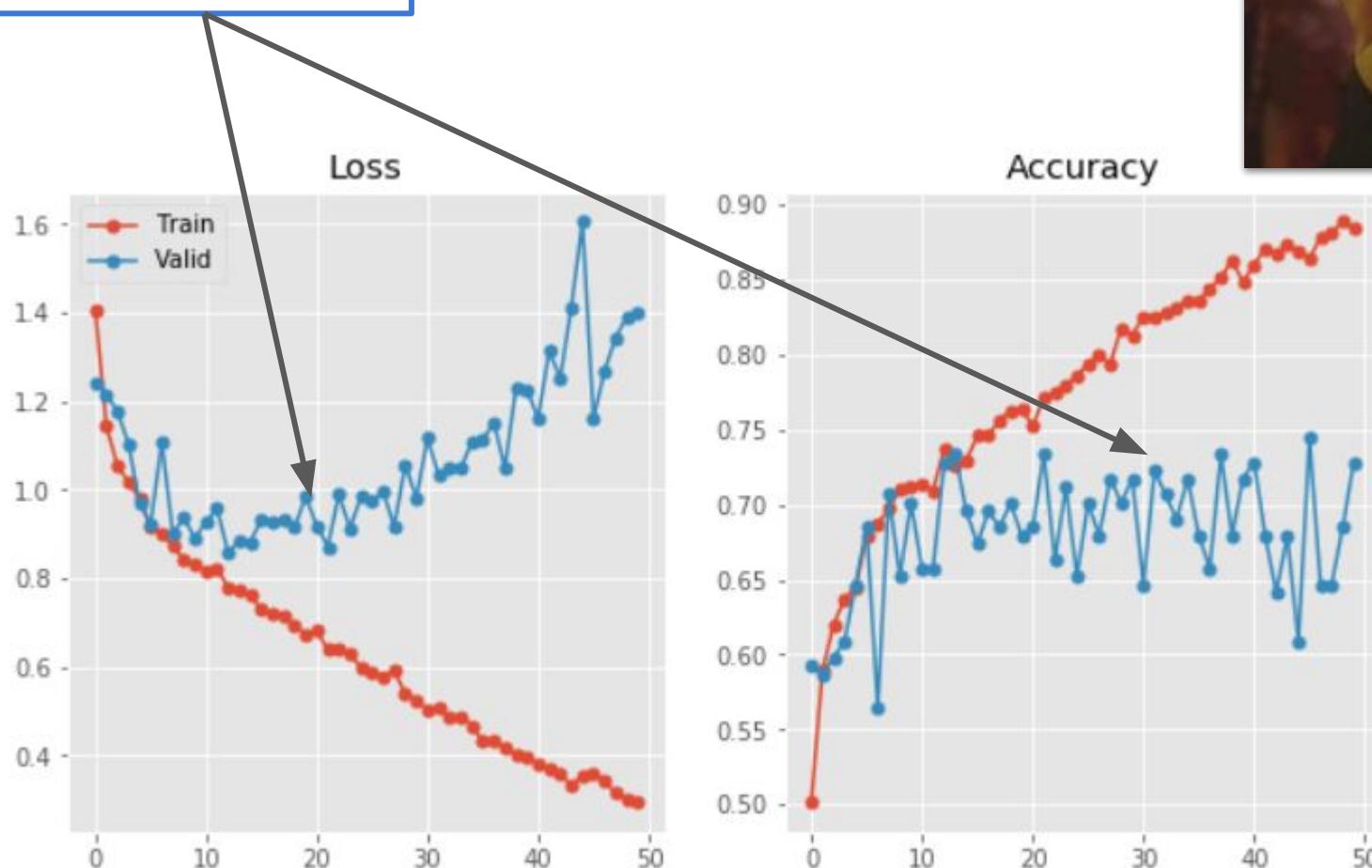
50 epochs後準確率達90%，可以舉杯慶祝了!!!



Deep Neural Network(DNN) Tips

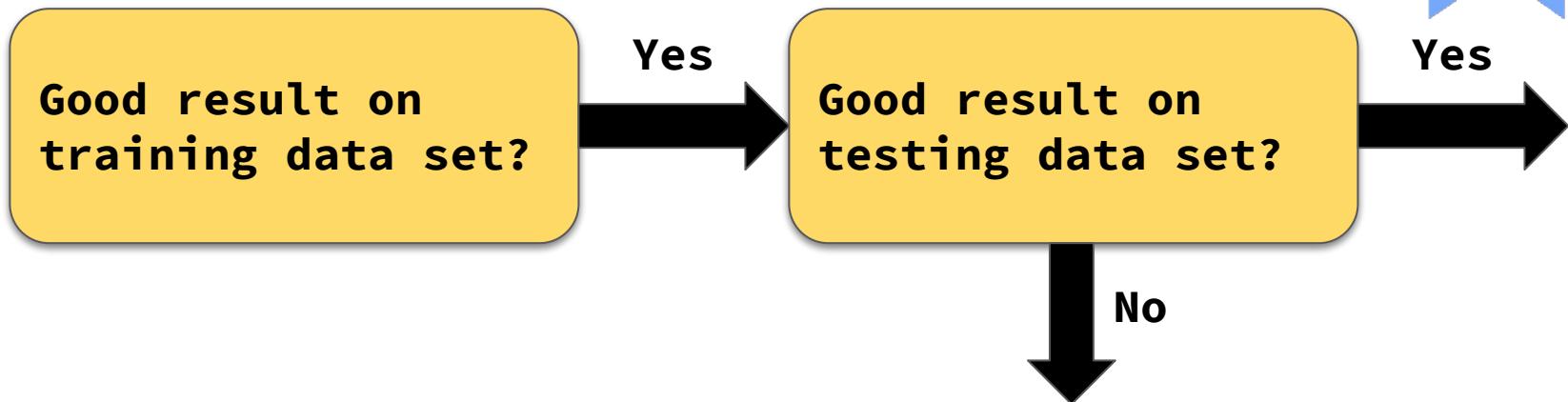
果然...世事難預料!

Overfitting!



Deep Neural Network(DNN) Tips

Overfitting

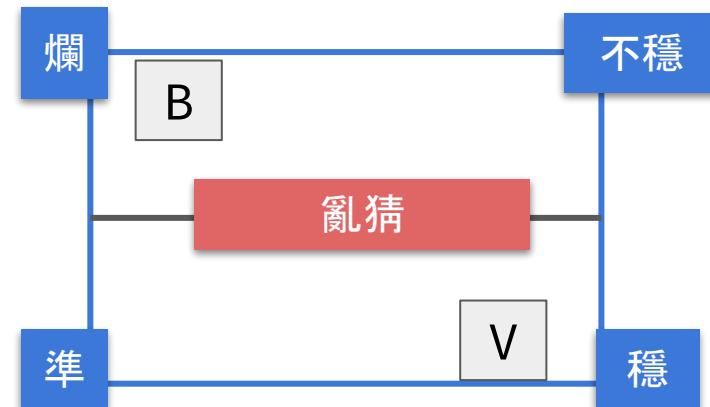
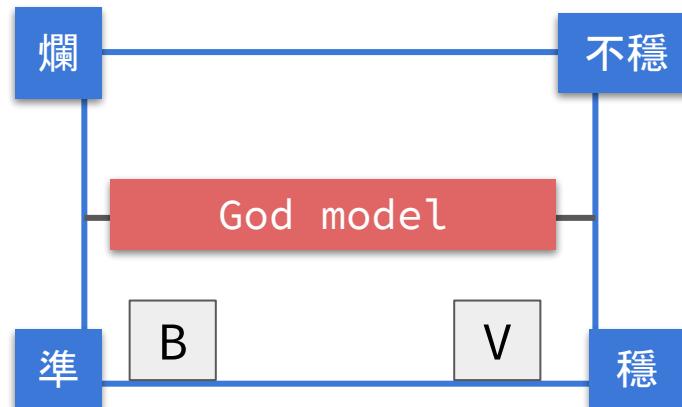
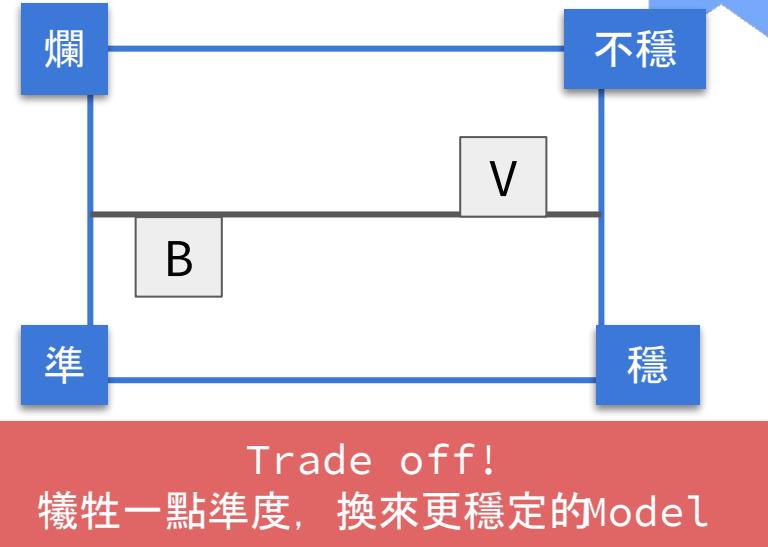
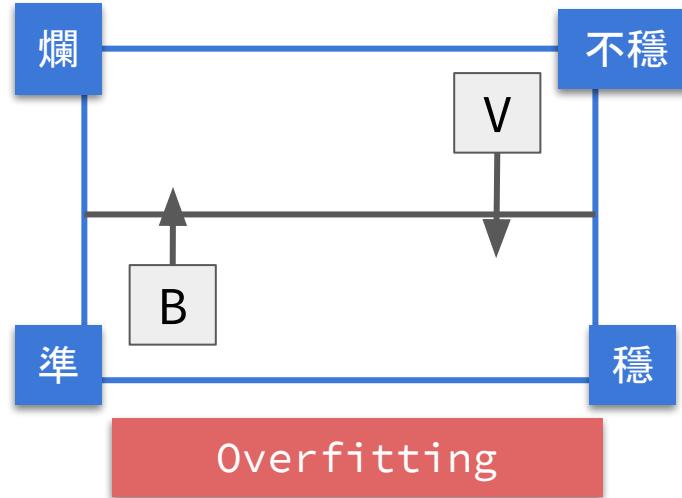


Overfitting的意義:

代表model已經太過於擬合data,
必須限制model的能力, **達到**
generalization的目的

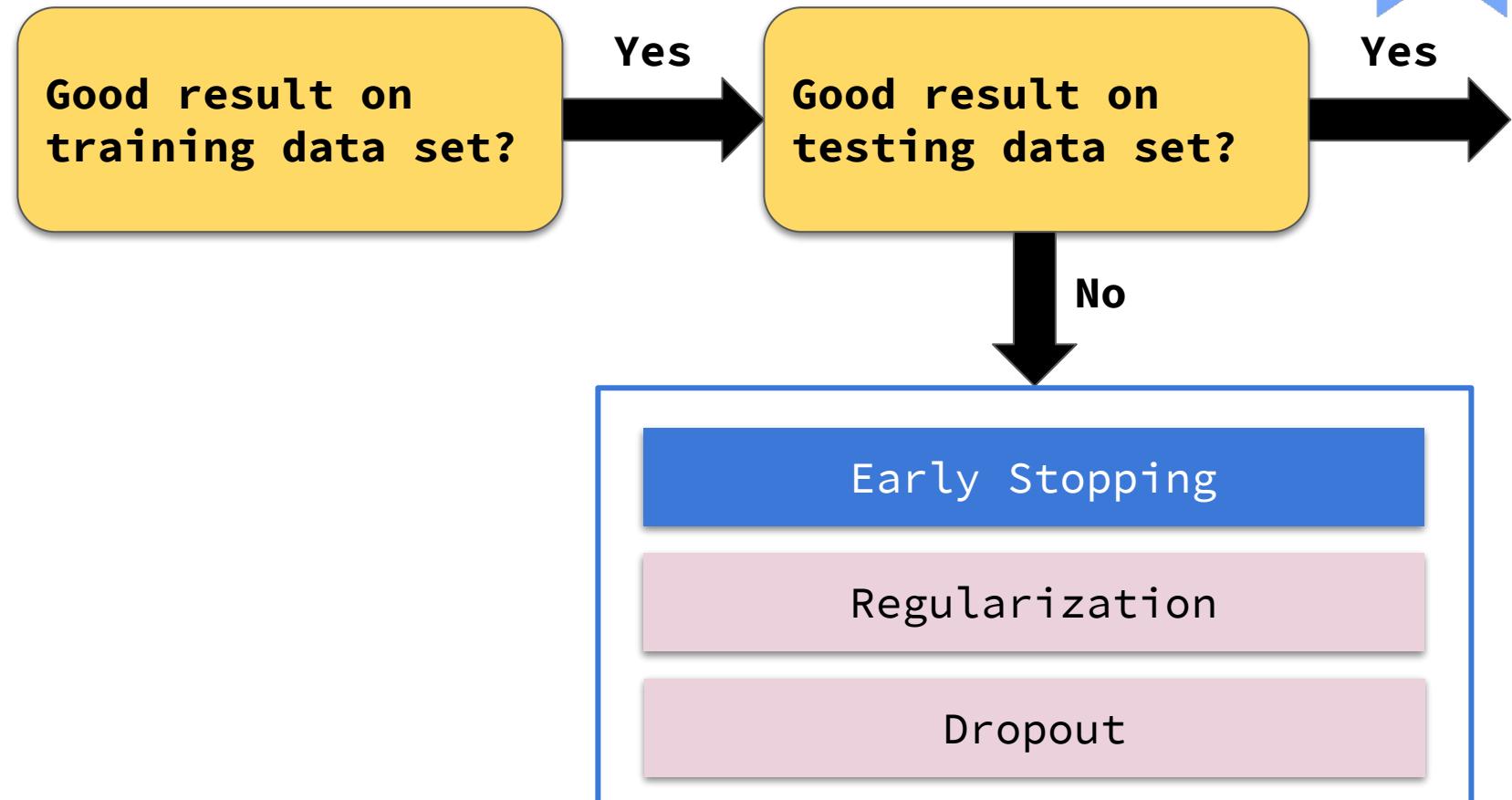
Deep Neural Network(DNN) Tips

Model Bias and Variance



Deep Neural Network(DNN) Tips

Overfitting

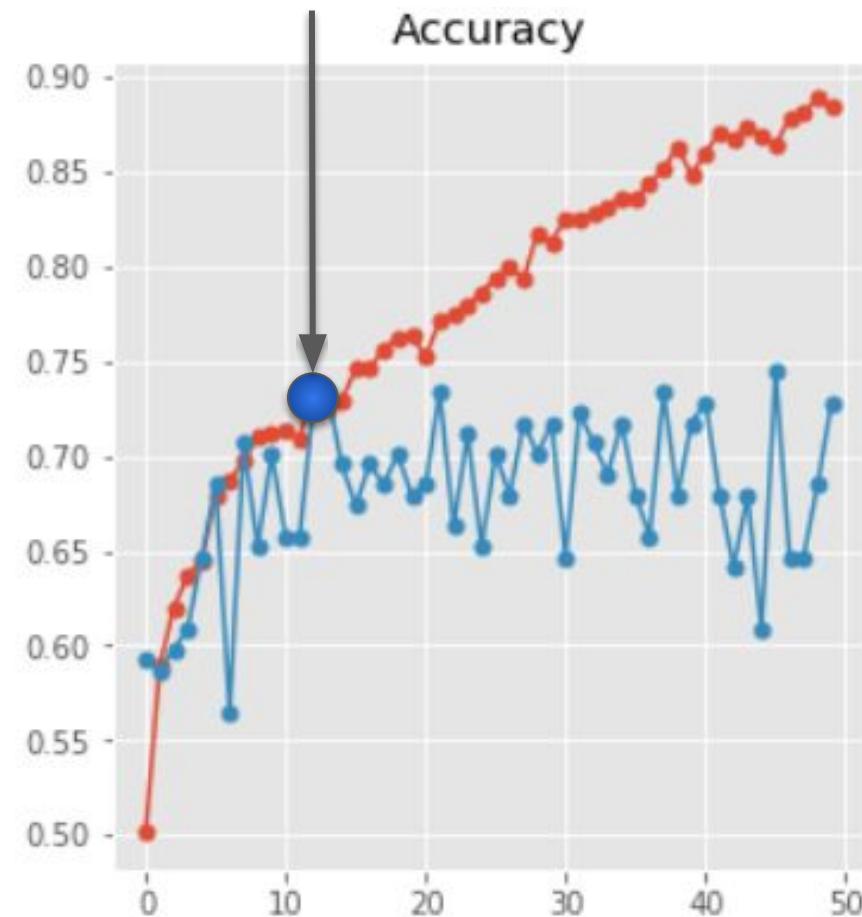
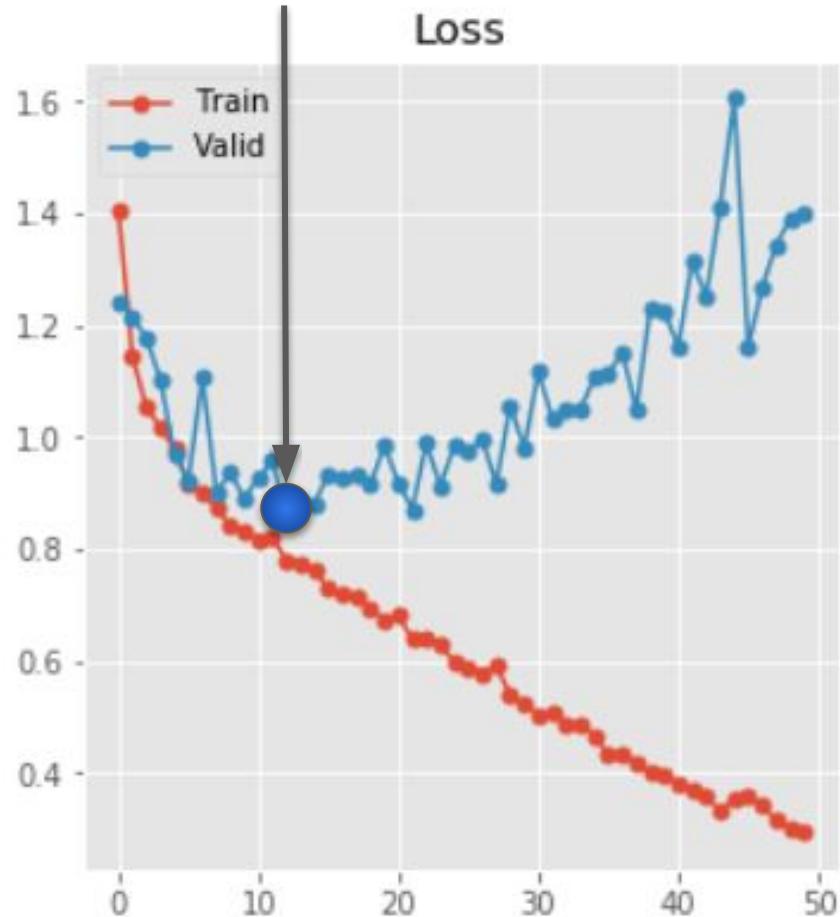


Deep Neural Network(DNN) Tips

Early Stopping: 懸崖勒馬!



早一點停下來就沒事了！



Deep Neural Network(DNN) Tips

Early Stopping: 懸崖勒馬!



```
tf.reset_default_graph()
model_adam = ModelAdam()
model_adam.fit(data_fn(X_train, Y_train, n_batch, shuffle=True),
                data_fn(X_valid, Y_valid, n_batch, shuffle=False),
                n_epoch,
                lr=learning_rate,
                callback=EarlyStopping(thres=5))
```



Model fit加上這個callback object即可!

Deep Neural Network(DNN) Tips

Early Stopping: 懸崖勒馬!



fit 有這麼一段, check callback function
on epoch end

```
for ep in range(1, n_epoch + 1):
    training ...
    if callback is not None:
        callback(self)
```

```
def __call__(self, model):
    last_vl_loss = model.hist["vl_loss"][-1]
    if self.min_loss > last_vl_loss:
        self.patient = 0
        self.min_loss = last_vl_loss
    else:
        self.patient += 1
    if self.patient >= self.thres:
        raise StopIteration("\nEarly Stopping, valid loss keep increasing for {} times !
                            lowest loss is {:.3f} !"
                            .format(self.patient, self.min_loss))
```

EarlyStopping物件

validation loss是否持續增加?超過一定threshold則停止

Deep Neural Network(DNN) Tips



Lab: lab_dnn_tutorial_practice.ipynb (5 ~ 10 minutes)

搜尋 "Do:" 就可以找到可能要修改的位置

lab filename	lab_tutorial_dnn_practice.ipynb
target	加入callback object: EarlyStopping check if overfitting

Solve Overfitting: EarlyStopping

檢查Valid Loss變化, 若發現Loss持續增加超過一定Threshold, 則停止Training

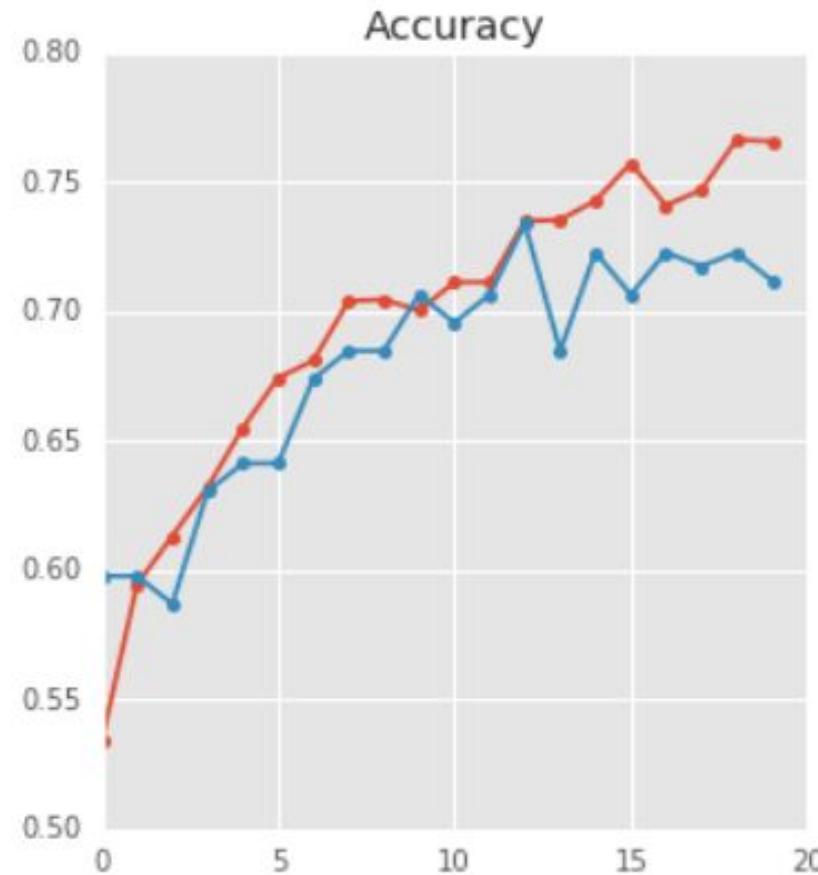
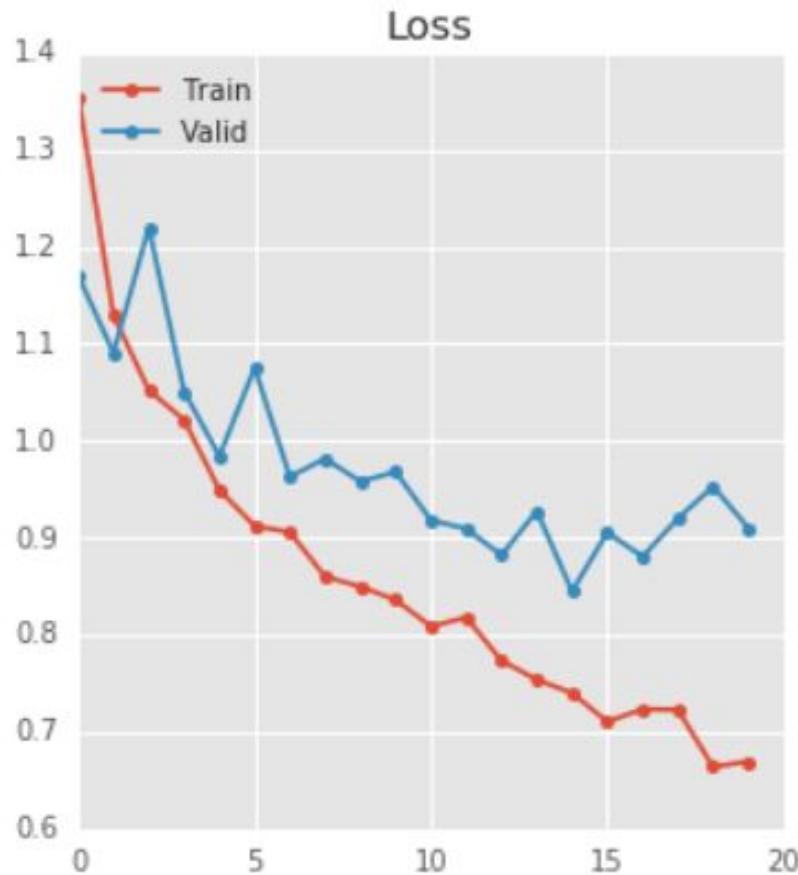
```
+ learning_rate: 0.001
+ loss function: softmax cross entropy
+ optimizer: AdamOptimizer
+ activation function: softplus
+ number of train epoch: 50
```

Deep Neural Network(DNN) Tips

Early Stopping - Threshold=5

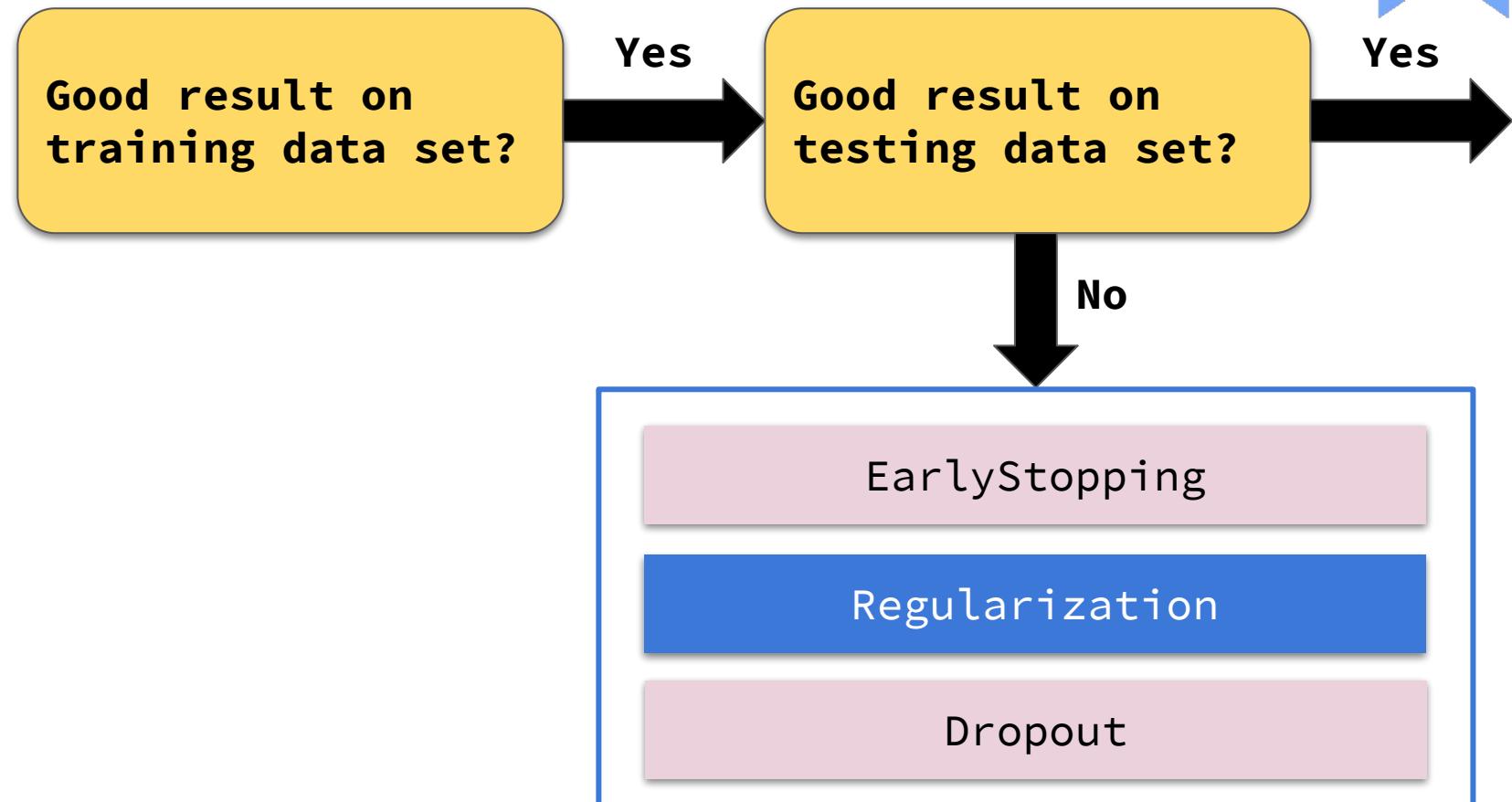


19/50. train loss: 0.664, valid loss: 0.953, train acc: 0.766, valid acc: 0.723
Early Stopping, valid loss keep increasing for 5 times ! lowest loss is 0.845 !



Deep Neural Network(DNN) Tips

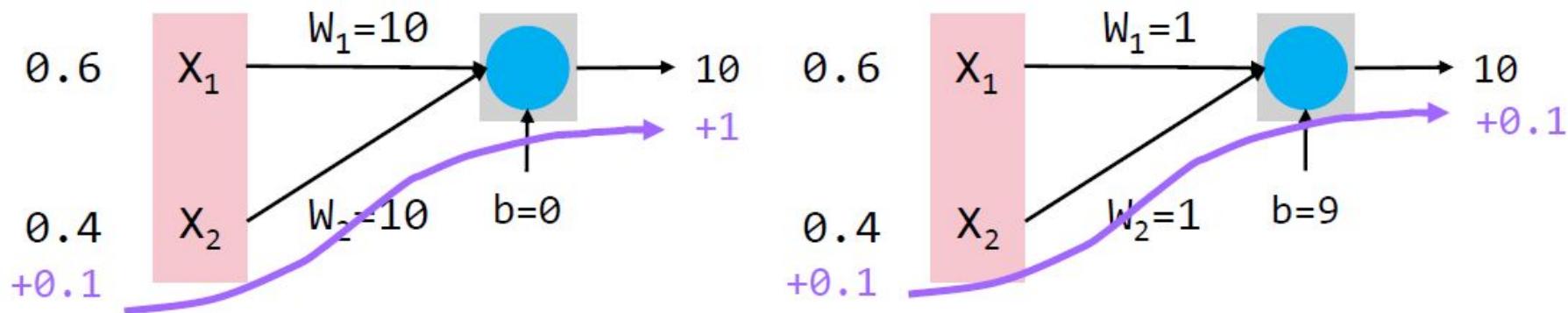
Overfitting



Deep Neural Network(DNN) Tips

Regularization

- Loss限制weights 的大小讓output曲線比較平滑



w_i 較小 $\rightarrow \Delta x_i$ 對 \hat{y} 造成的影響($\Delta \hat{y}$)較小
 \rightarrow 對 input 變化比較不敏感 \rightarrow generalization 好

Deep Neural Network(DNN) Tips

Regularization



- 作法: 加入目標(Loss)函數中, 一起優化

$$L'(\theta) = L(\theta) + \lambda(\text{regularizer})$$

- λ 是用來調整regularization的比重(通常在0.01以下)
- Regularizer兩種形式: L1 and L2

$$L_1 = \sum_i |w_i|$$

L1 norm: sum of absolute values

$$L_2 = \sum_i |w_i|^2$$

L2 norm: square sum of absolute values

一般狀況是不會加入Bias的參數 (Optional)

Deep Neural Network(DNN) Tips

Regularization: L2 Norm



L2 regularization:

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

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

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

$$= \underline{(1 - \eta \lambda)w^t} - \eta \frac{\partial L}{\partial w}$$

接近0

Weight Decay

Deep Neural Network(DNN) Tips



Lab: lab_dnn_tutorial_practice.ipynb (5 ~ 10 minutes)

lab filename	lab_tutorial_dnn_practice.ipynb
target	加入l2 regularizer: <code>tf.contrib.layers.l2_regularizer</code> <code>tf.contrib.layers.l1_regularizer</code>

Solve Overfitting: Add Regularizer Term to Loss Function

```
+ learning_rate: 0.001
+ loss function: softmax cross entropy
+ optimizer: AdamOptimizer
+ activation function: softplus
+ number of train epoch: 50

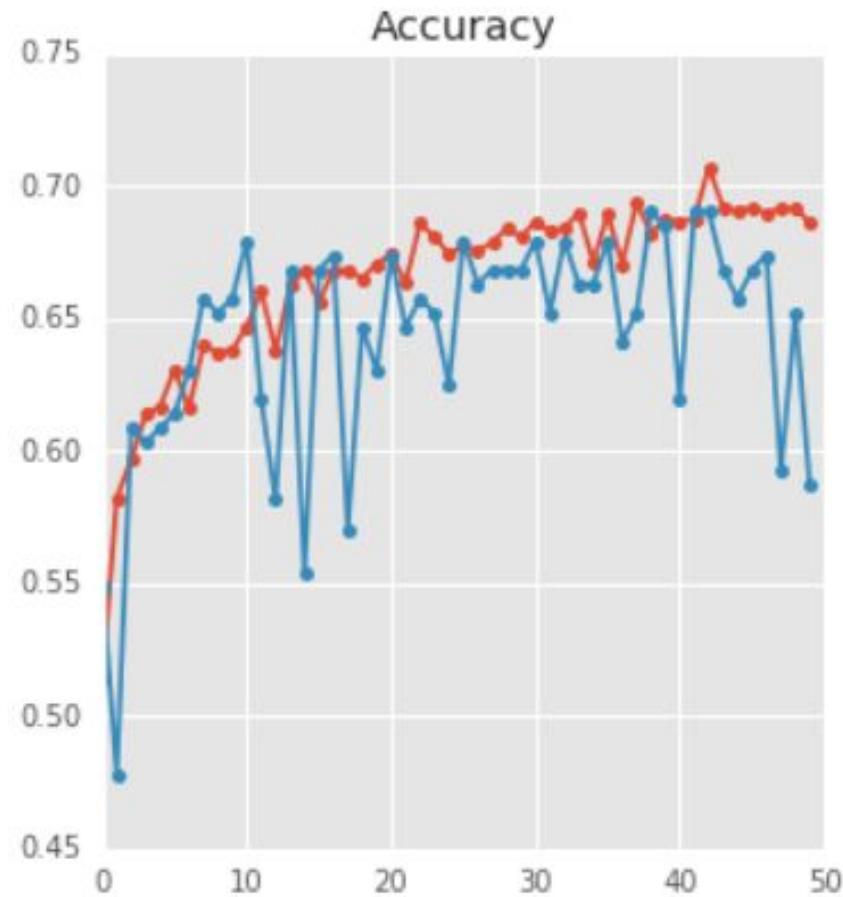
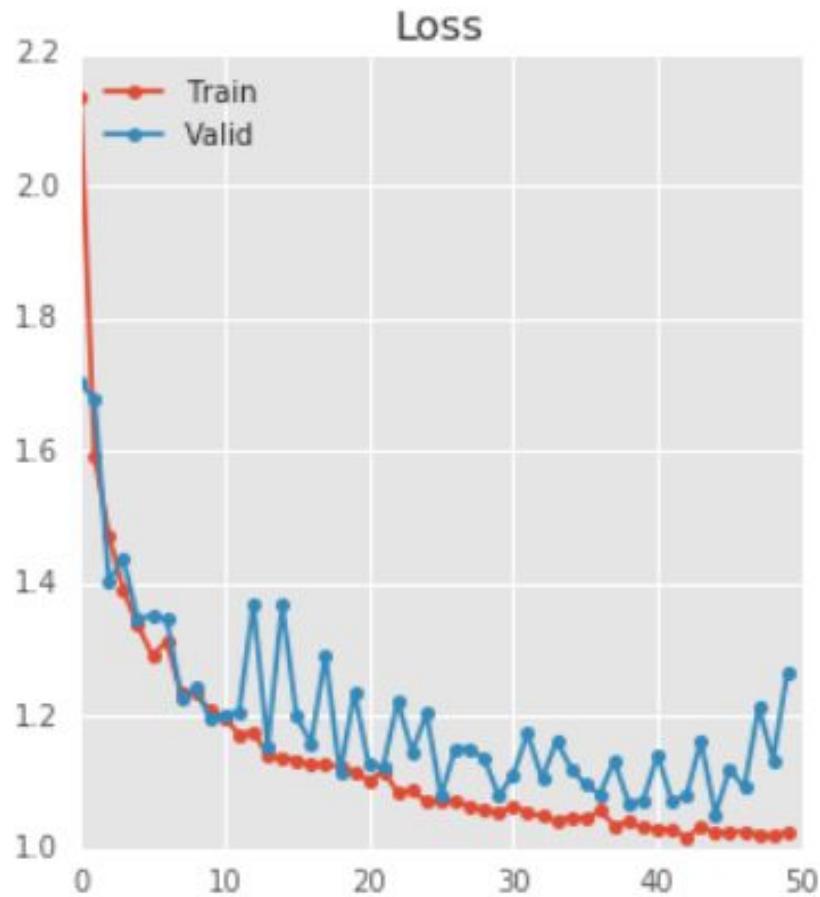
+ >>> add l2 regularizer term to each hidden layers with scale 0.01
```

Deep Neural Network(DNN) Tips

Result - Regularization

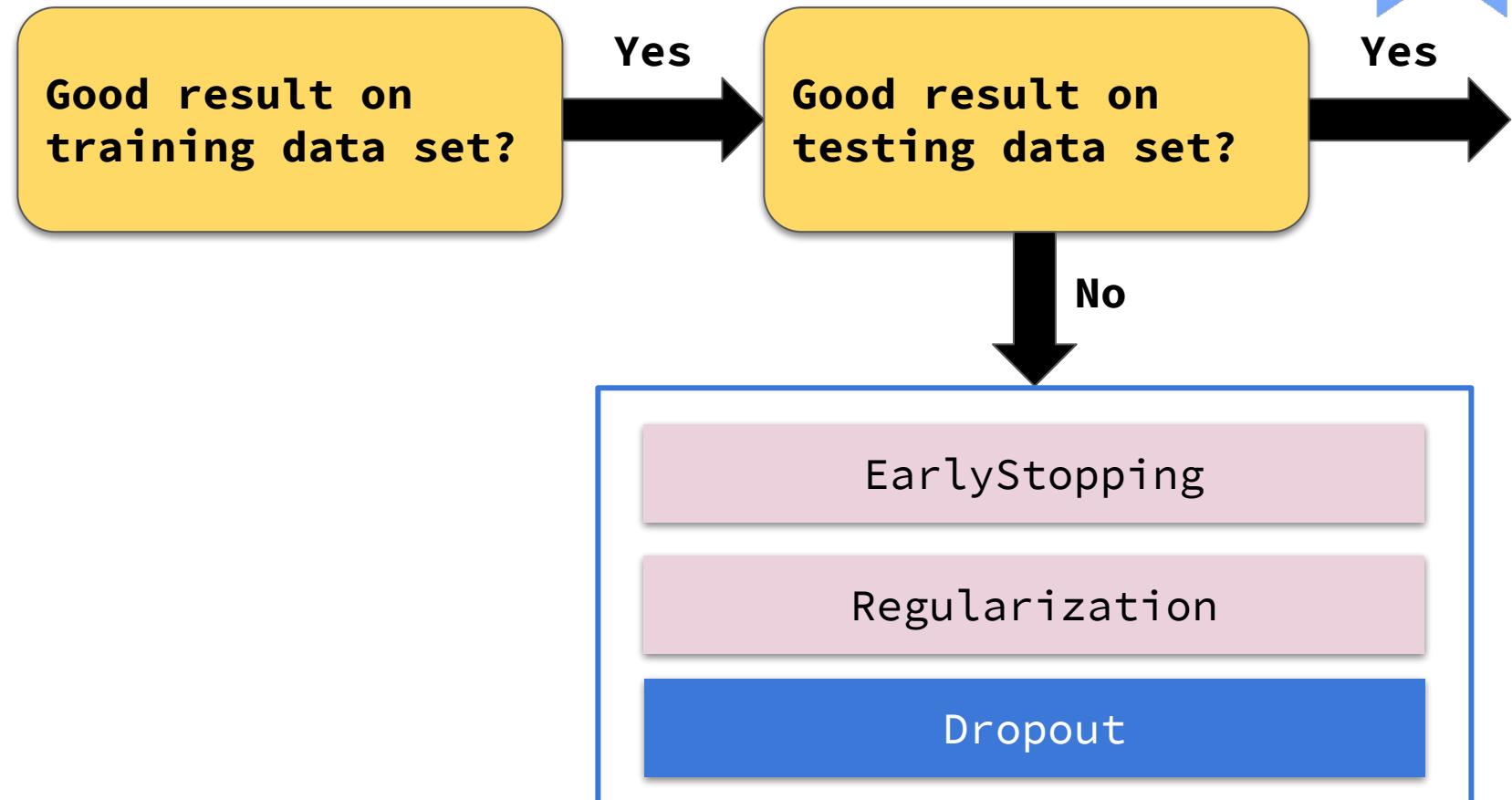


50/50. train loss: 1.025, valid loss: 1.268, train acc: 0.686, valid acc: 0.587



Deep Neural Network(DNN) Tips

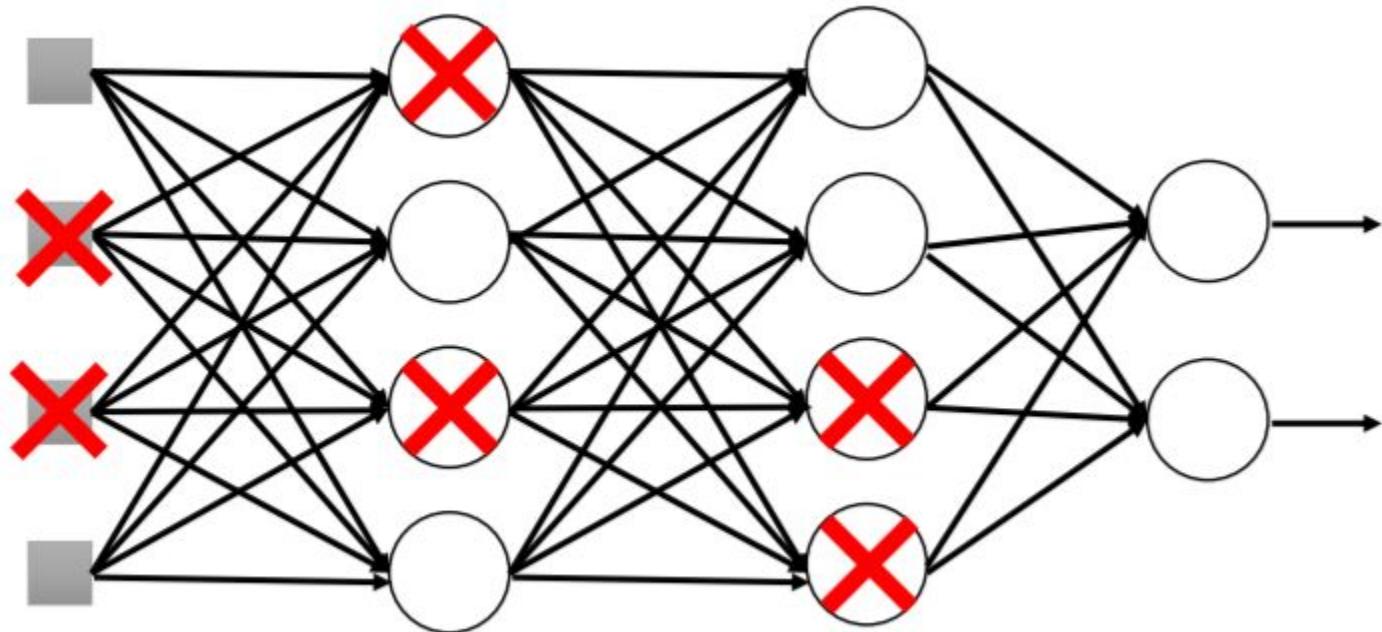
Overfitting



Dropout



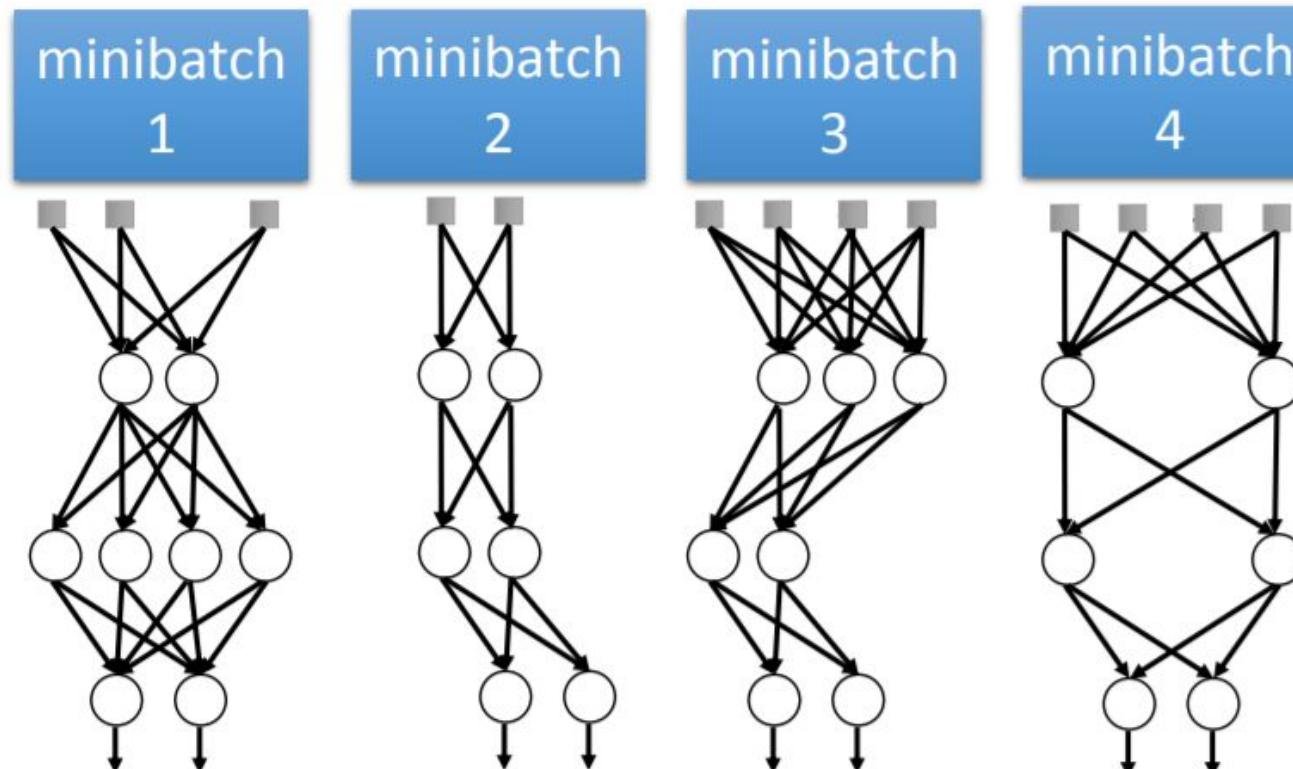
Training:



- 每個batch training時都會隨機丟掉 $p\%$ 的neuron
(weight設為0 or 乘0)
- Dropout always work on training time, not on evaluate or prediction**

Deep Neural Network(DNN) Tips

Dropout - 終極的ensemble方法



Training of Dropout

M neurons

⋮

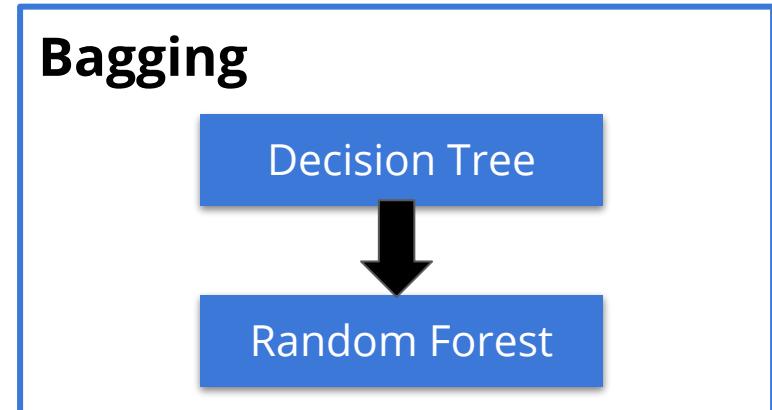
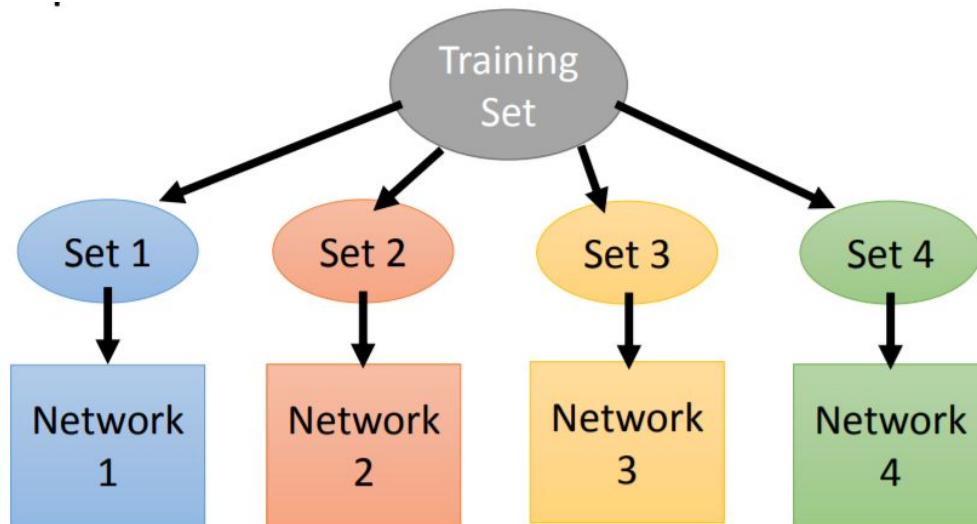


2^M possible networks

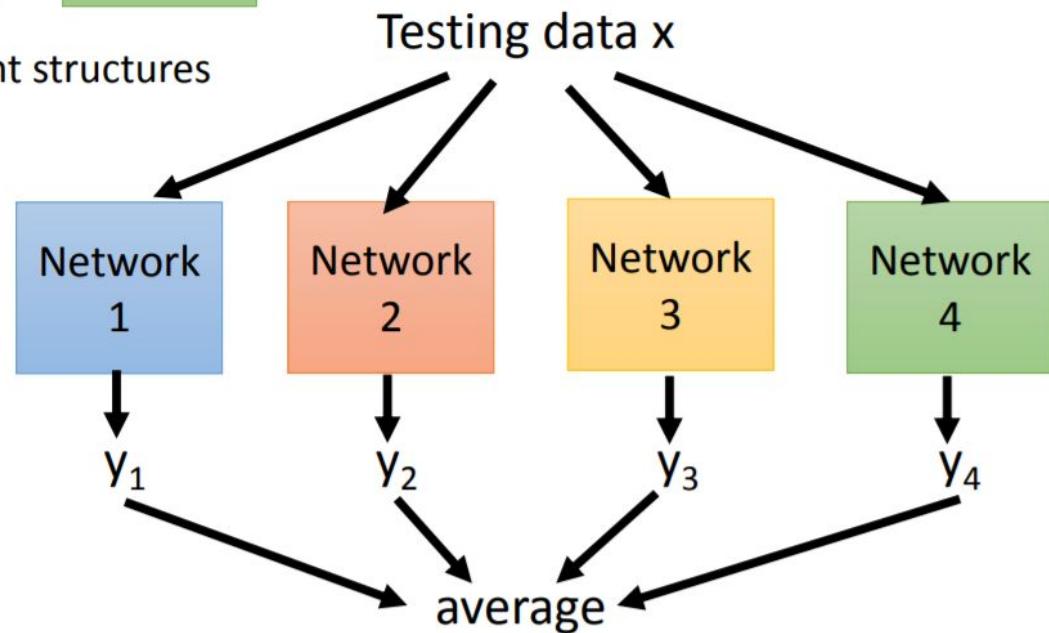
- ❑ Using one mini-batch to train one network
- ❑ Weights in the network are shared

Deep Neural Network(DNN) Tips

Dropout - 終極的ensemble方法



Train a bunch of networks with different structures



Deep Neural Network(DNN) Tips



直覺的理解

增加訓練的難度



在真正的考驗時爆發



Dropout的結果

- ❑ 會讓training performance變差, 包含training速度
- ❑ 不要一開始就使用dropout!

Deep Neural Network(DNN) Tips



Lab: lab_dnn_tutorial_practice.ipynb (5 ~ 10 minutes)

搜尋 "Do:" 就可以找到可能要修改的位置

lab filename	lab_tutorial_dnn_practice.ipynb
target	Network加入dropout: drop_rate可嘗試 0.3, 0.4, 0.5, 0.6

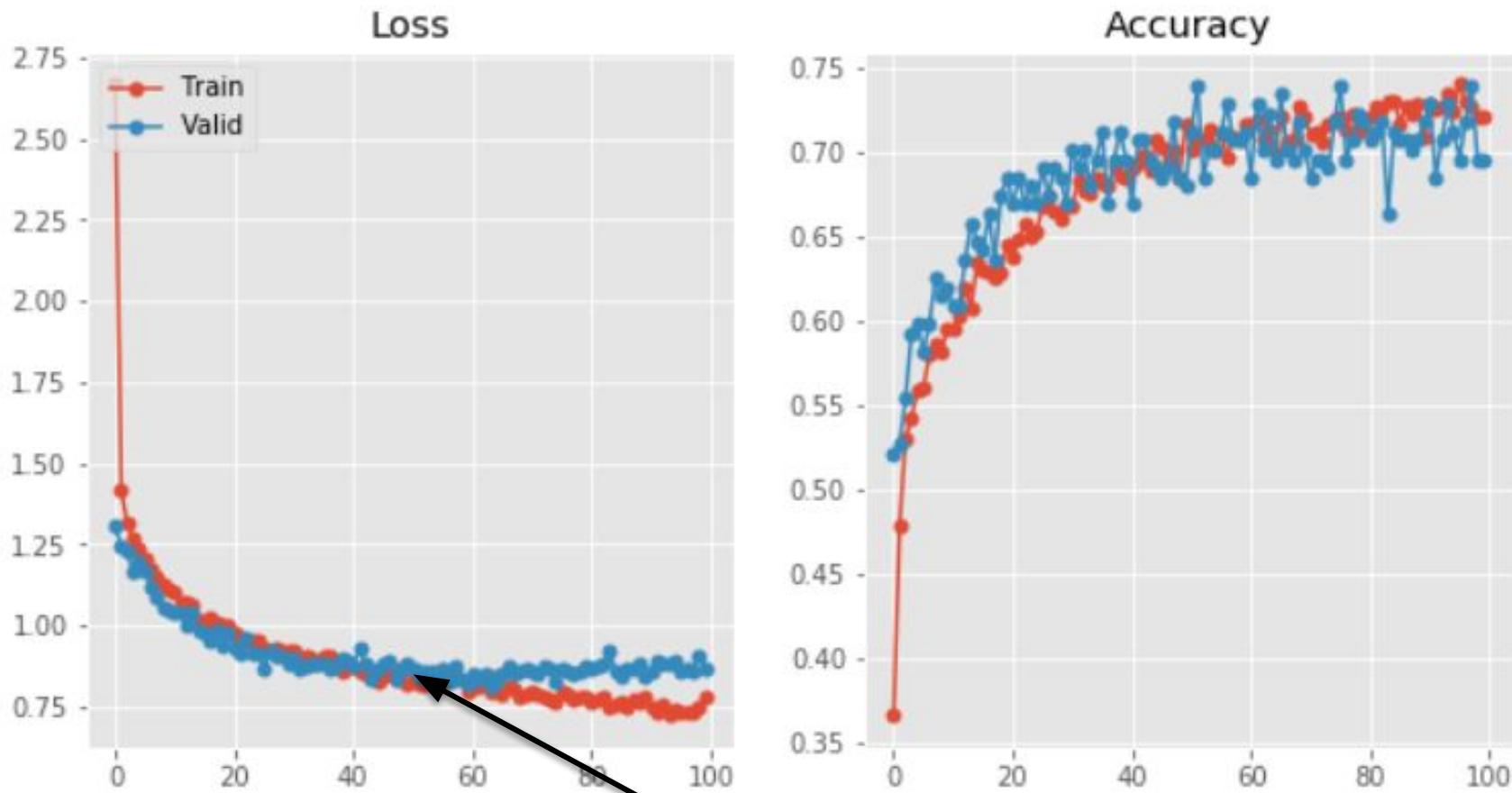
Solve Overfitting: Dropout

```
+ learning_rate: 0.001
+ loss function: softmax cross entropy
+ optimizer: AdamOptimizer
+ activation function: relu

+ number of train epoch: 100
>>> Dropout會讓training速度變慢，增加epoch次數讓分數更好！
```

Deep Neural Network(DNN) Tips

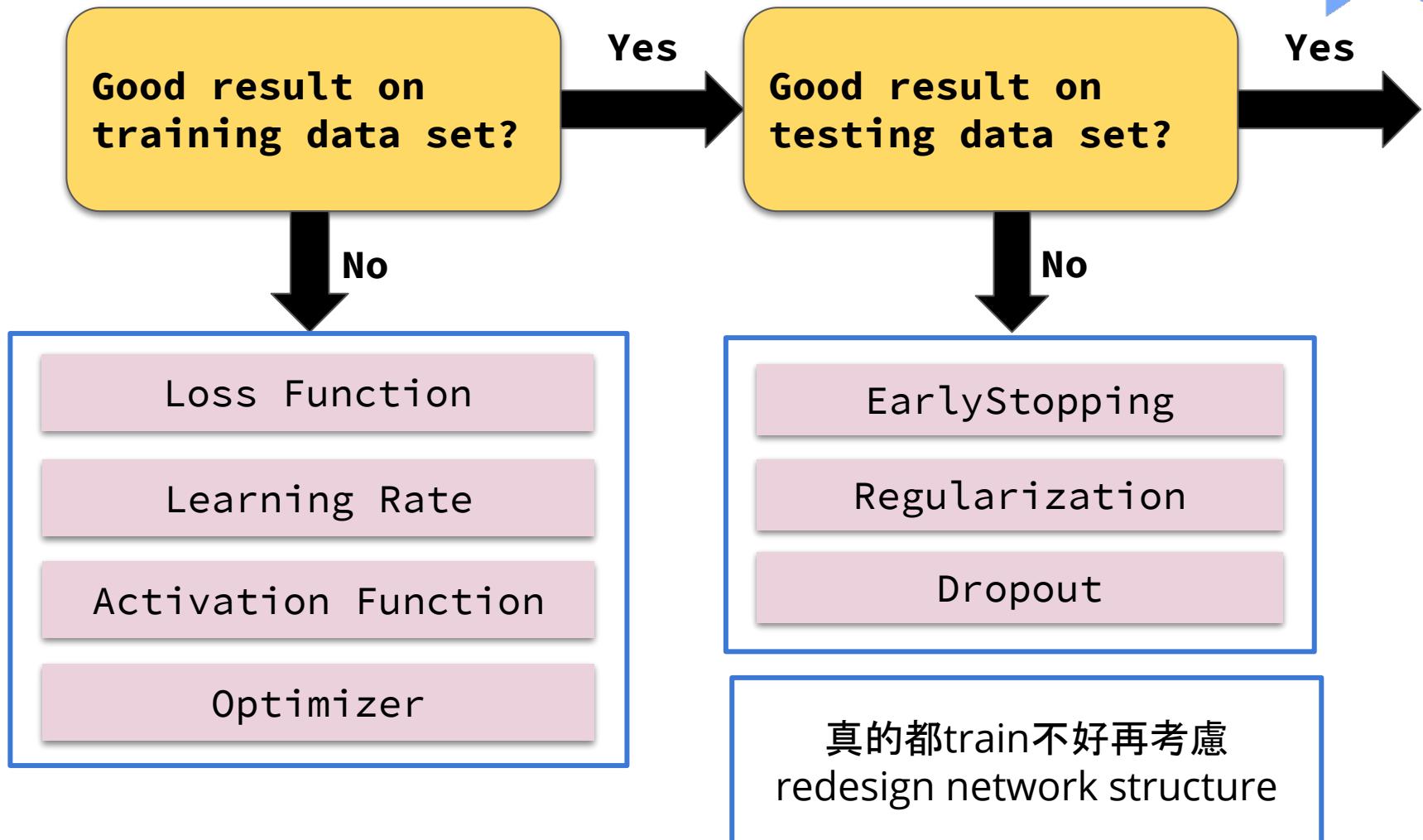
Result - Dropout(After 100 Epochs)



Valid Loss曲線在50 epochs之後已經下降變慢，但並沒有遽增

Deep Neural Network(DNN) Tips

Recall



Recommendation Engine in Matrix Factorization

Machine Learning Framework

Deep Neural Network(DNN) Tips

Recommnedation Engine in Matrix Factorization

Matrix Factorization with DNN



Recommendation Engine in Matrix Factorization

Movielens Data



User Movie Rating

	userId	movieId	rating	timestamp
0	0	30	2.5	1260759144
1	0	833	3.0	1260759179
2	0	859	3.0	1260759182
3	0	906	2.0	1260759185
4	0	931	4.0	1260759205

User Movie Tag

	userId	movieId	tag	timestamp
0	14	304	sandra 'boring' bullock	1138537770
1	14	1517	dentist	1193435061
2	14	5166	Cambodia	1170560997
3	14	6118	Russian	1170626366
4	14	6178	forgettable	1141391765

Movie Metadata

	movieId	title	genres
0	0	Toy Story (1995)	Adventure Animation Children Comedy Fantasy
1	1	Jumanji (1995)	Adventure Children Fantasy
2	2	Grumpier Old Men (1995)	Comedy Romance
3	3	Waiting to Exhale (1995)	Comedy Drama Romance
4	4	Father of the Bride Part II (1995)	Comedy

Recommendation Engine in Matrix Factorization



Concept of Latent Factor

U: User, M: Movie, R: Rating(1 ~ 5)

	M1	M2	M3	M4	M5
U1	3	5			1
U2			2	4	
U3	5		1		
U4		3		3	3

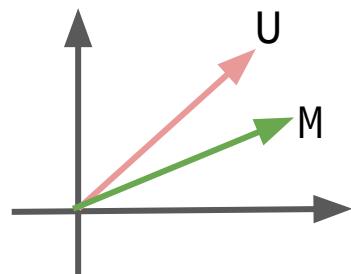
≈

U1			
U2			
U3			
U4			

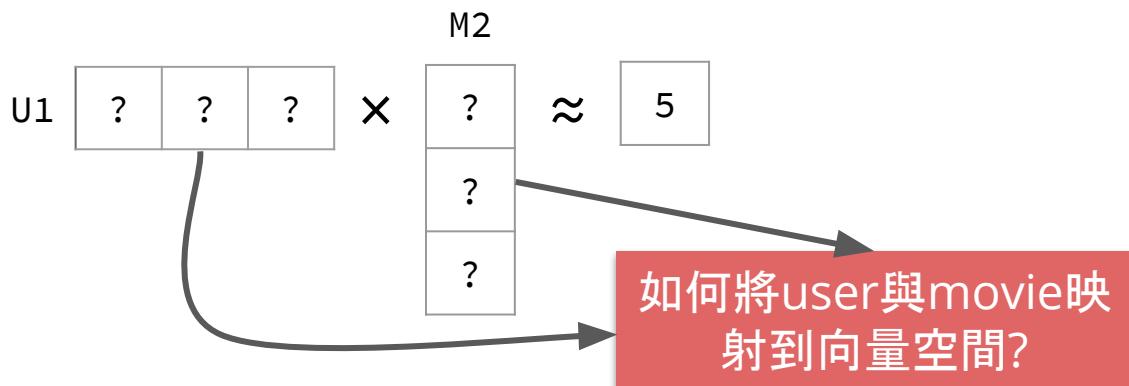
X

M1				
M2				
M3				
M4				

latent factor



User與Movie的距離



Recommendation Engine in Matrix Factorization

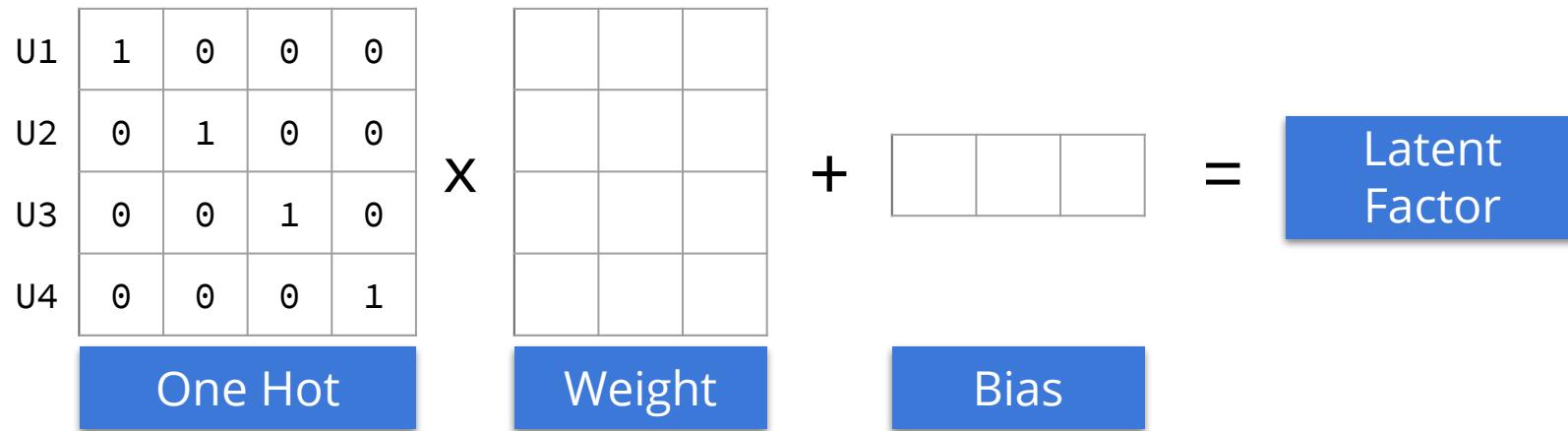
Relections(Transformation)



user id \Rightarrow 1263, 1080, 87...

movie id \Rightarrow 103, 554, 187...

- 把user和movie看成是categorical variable \Rightarrow One Hot Encoding
- **Recall feature transformation(cascading functions)**



Recommendation Engine in Matrix Factorization

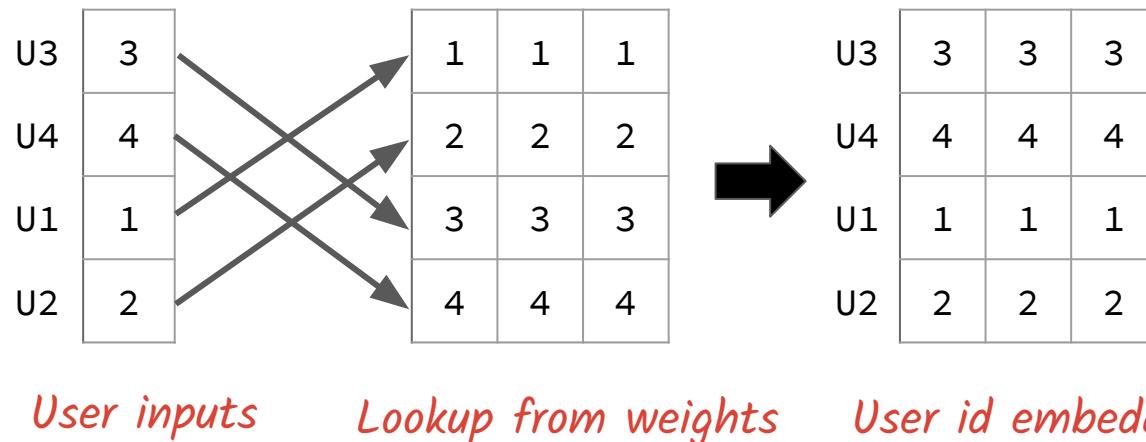
User ID To Embedding



$$\begin{array}{c} \text{U1} \\ \text{U2} \\ \text{U3} \\ \text{U4} \end{array} \begin{array}{|c|c|c|c|} \hline & 1 & 0 & 0 & 0 \\ \hline & 0 & 1 & 0 & 0 \\ \hline & 0 & 0 & 1 & 0 \\ \hline & 0 & 0 & 0 & 1 \\ \hline \end{array} \times \begin{array}{|c|c|c|} \hline & 1 & 1 & 1 \\ \hline & 2 & 2 & 2 \\ \hline & 3 & 3 & 3 \\ \hline & 4 & 4 & 4 \\ \hline \end{array} = \begin{array}{|c|c|c|} \hline & 1 & 1 & 1 \\ \hline & 2 & 2 & 2 \\ \hline & 3 & 3 & 3 \\ \hline & 4 & 4 & 4 \\ \hline \end{array}$$

One Hot *Weights*

第i號user的latent factor等價於取Weight的第i列 \Rightarrow **Embedding**



- movie id embedding 也是如法炮製
- 實際都會省略 bias

Recommendation Engine in Matrix Factorization

Model Formulation and Loss Function

$U, M \Rightarrow$ 透過embedding取得的latent factor



	M1	M2	M3	M4	M5
U1	3	5			1
U2			2	4	
U3	5		1		
U4		3		3	3

$$r_{12} \Rightarrow u_1 \cdot m_2 \approx 5$$

$$r_{33} \Rightarrow u_3 \cdot m_3 \approx 1$$

$$r_{23} \Rightarrow u_2 \cdot m_3 \approx 2$$

:

Regression

$$L = \sum_{(i,j)} (u_i \cdot m_j - r_{ij})^2$$

model function

- User rating behavior and movie being rated are highly **biased**
- Recall linear combination $\Rightarrow \mathbf{wx + b}$

$$\rightarrow u_i \cdot m_j + b_u + b_m + b_{global}$$

- 不是 U 和 M 的外在影響因素
- Model bias

Minimizing $L = \sum_{(i,j)} (u_i \cdot m_j + b_u + b_m + b_{global} - r_{ij})^2$

model function with bias

Recommendation Engine in Matrix Factorization

Prediction

以向量運算表示 $u_i \cdot m_j + b_u + b_m + b_{global}$



Predict =

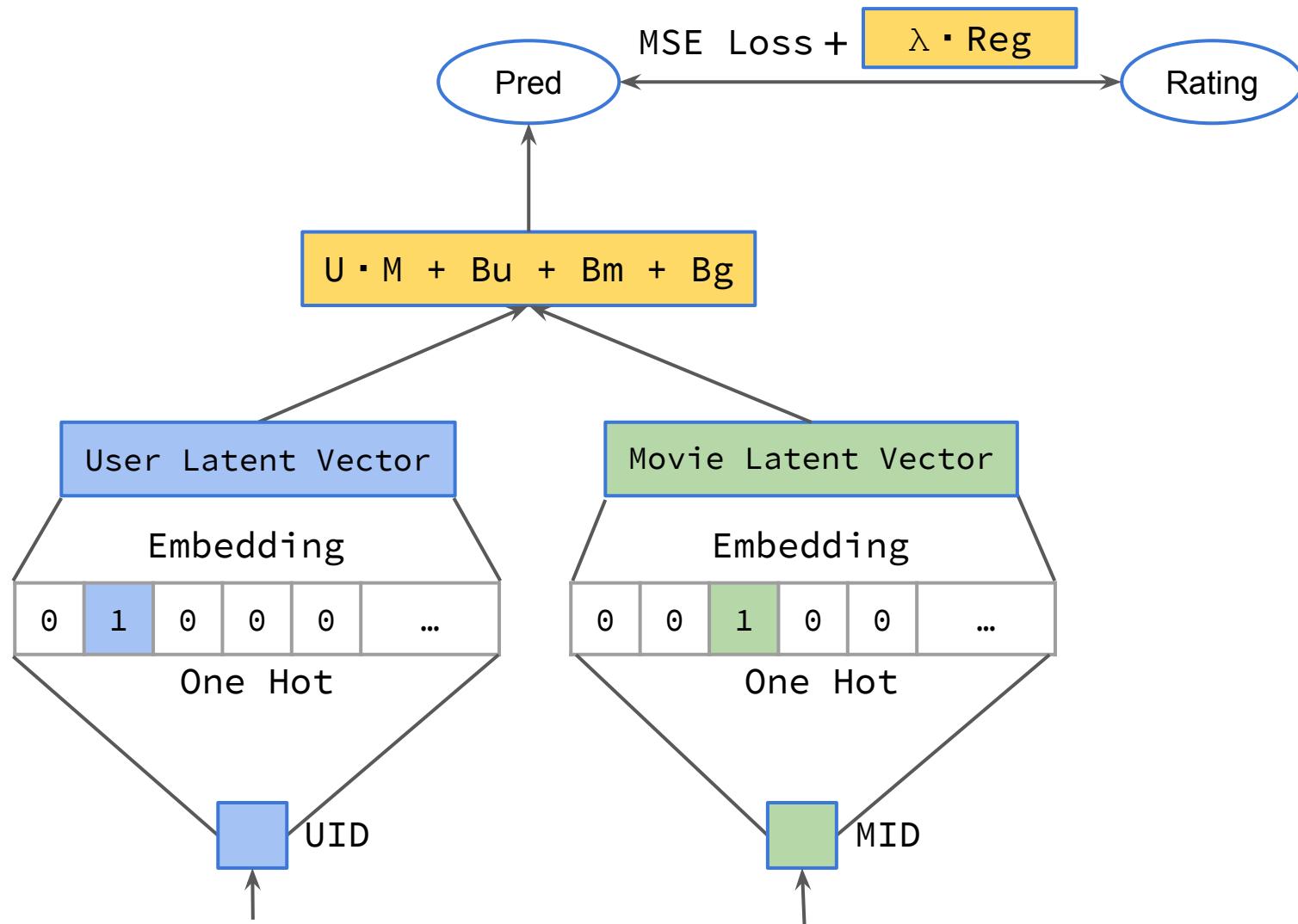
$$\begin{array}{c} u_i \cdot m_j \\ \text{M1 M2 M3 M4 M5} \\ \hline \text{U1} & 3.5 & 4.7 & 0.87 & 1.87 & 0.8 \\ \text{U2} & 4.87 & 5.6 & 2.1 & 3.87 & 4.57 \\ \text{U3} & 5.87 & 4.78 & 1.87 & 1.2 & 3.6 \\ \text{U4} & 1.7 & 3.2 & 2.9 & 2.87 & 2.6 \end{array} + \begin{array}{c} \text{Row Wise} \\ b_u \\ \hline 0.1 \\ 0.2 \\ 0.3 \\ 0.4 \end{array} + \boxed{0.5} b_{global} + \begin{array}{c} \text{Element Wise} \\ \hline \begin{array}{c} + \\ \begin{array}{c} 0.8 & 0.7 & 0.6 & 0.5 & 0.4 \end{array} \\ b_m \end{array} \end{array}$$

```
[[ 4.9   6.    2.07  2.97  1.8 ]
 [ 6.37  7.    3.4   5.07  5.67]
 [ 7.47  6.28  3.27  2.5   4.8 ]
 [ 3.4   4.8   4.4   4.27  3.9 ]]
```

這個例子實際跑出來的

Recommendation Engine in Matrix Factorization

First Model - Basic MF



First Model - Basic MF



- ❑ Function set formulation

$$f_{u,m}(x) = u_i \cdot m_j + b_u + b_m + b_{global}$$

- ❑ Loss function

$$L = \sum (f_{u,m}(x) - r_{ij})^2 + \lambda \cdot \frac{1}{2} (\sum u^2 + \sum m^2 + \sum b_u^2 + \sum b_m^2)$$

Recommendation Engine in Matrix Factorization

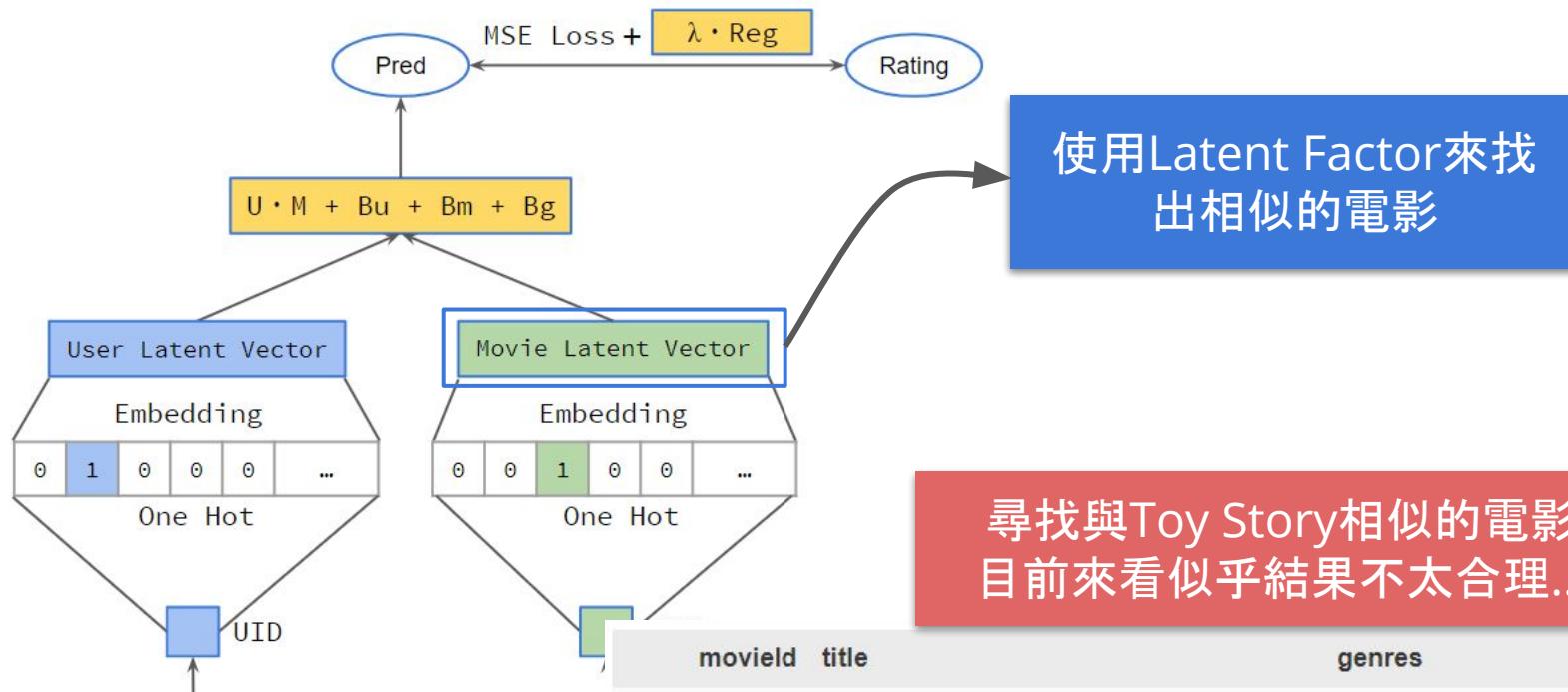
Basic MF



filename	reco_model_mf.ipynb
lab filename	lab_reco_model_mf.ipynb
number of users	671
number of movies	9125
rating range	0.5 ~ 5分, <input type="checkbox"/> 4分以上算正評 <input type="checkbox"/> 未滿4分認為是負評或是不列入推薦名單
neural network	matrix factorization

Recommendation Engine in Matrix Factorization

Basic MF - 活用Latent Factor(Embedding)!



	movielid	title	genres
0	0	Toy Story (1995)	Adventure Animation Children Comedy Fantasy
7551	7551	Letters to Juliet (2010)	Drama Romance
5158	5158	Laws of Attraction (2004)	Comedy Romance
8473	8473	Lone Survivor (2013)	Action Drama Thriller War
900	900	Perfect Candidate, A (1996)	Documentary
2786	2786	My Chauffeur (1986)	Comedy
6867	6867	To the Left of the Father (Lavoura Arcaica) (2010)	Drama
3794	3794	From Hell (2001)	Crime Horror Mystery Thriller
921	921	T-Men (1947)	Film-Noir
4279	4279	The Pumaman (1980)	Action Adventure Fantasy Sci-Fi

Recommendation Engine in Matrix Factorization



Model All Metrics

Model	Dummy Model(亂猜)	Basic MF
RMSE Loss	3.23	0.92
ROC AUC	0.50	0.74
NDCG	strict: 0.45 normal: 0.63	strict: 0.63 normal: 0.76
Precision at 10	strict: 0.37 normal: 0.54	strict: 0.50 normal: 0.63

Recommendation Engine in Matrix Factorization



Lab: lab_reco_model_mf.ipynb (20 minutes)

搜尋 ”Do:” 就可以找到可能要修改的位置

lab filename	lab_reco_model_mf.ipynb
target	您可能要修改的參數: <input type="checkbox"/> learning rate <input type="checkbox"/> dim: dimension of latent factor(user and movie) <input type="checkbox"/> reg: 正規項的強度 <input type="checkbox"/> dropout <input type="checkbox"/> loss function <input type="checkbox"/> optimizer
ideal score	valid loss: 將低到0.92左右 roc auc: 0.74左右 ndcg: strict condition 0.63左右

Model - MF with History



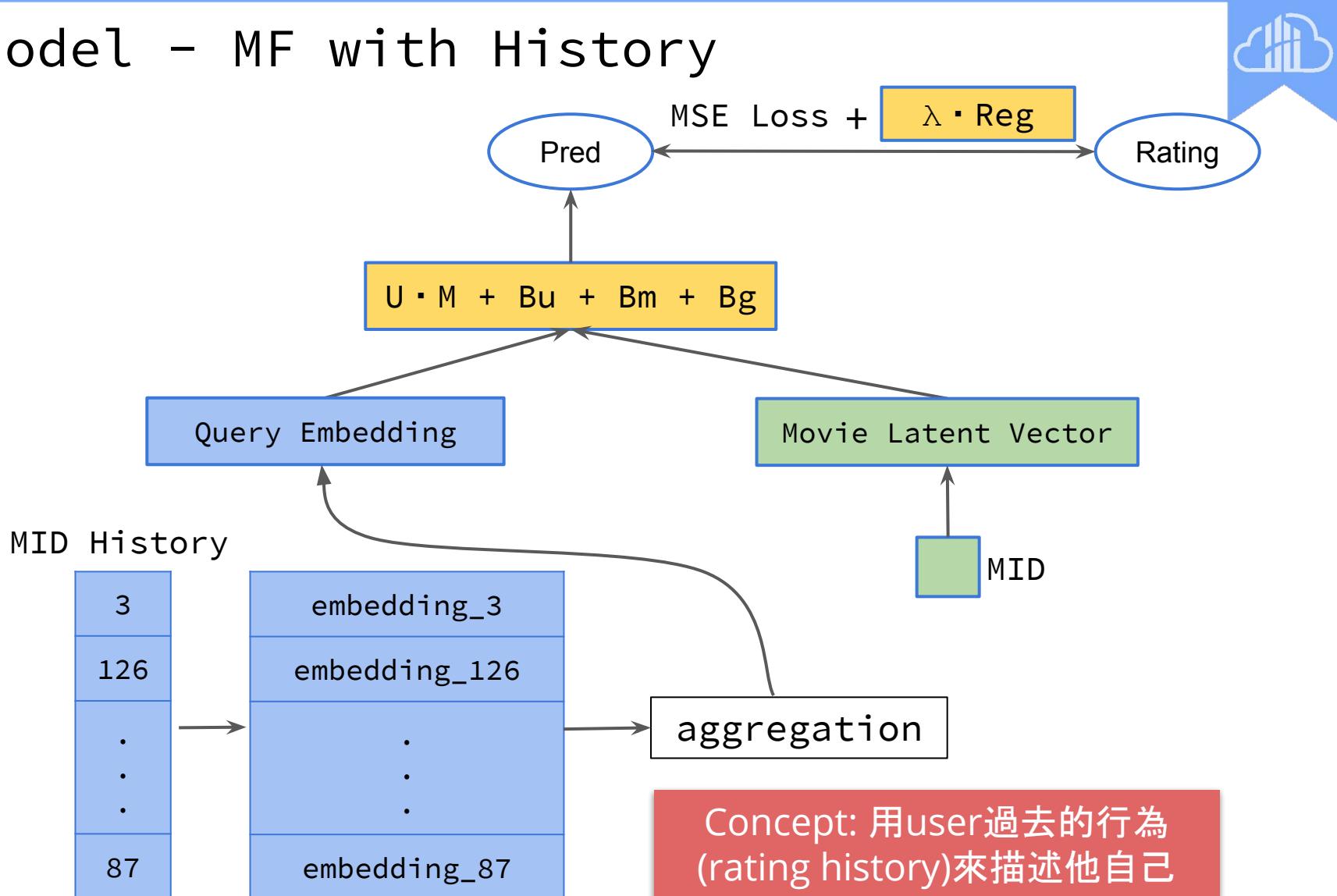
- 如果新的User進來...
 - 假設Profile資訊不足, 不知道是男是女...
 - 推薦最歡迎的電影(完全沒User rating)
 - 就算累積了Rating history, Model還是要重train
 - Fall back to content base recommendation



在不重train的狀況下, 希望能夠對新來的
user(有rating history)做推薦

Recommendation Engine in Matrix Factorization

Model – MF with History



Recommendation Engine in Matrix Factorization

Model - MF with History



filename	reco_model_mf_with_history.ipynb
lab filename	lab_reco_model_mf_with_history.ipynb
number of users	671
number of movies	9125
rating range	0.5 ~ 5分, <input type="checkbox"/> 4分以上算正評 <input type="checkbox"/> 未滿4分認為是負評或是不列入推薦名單
neural network	matrix factorization

Recommendation Engine in Matrix Factorization

Model - MF with History

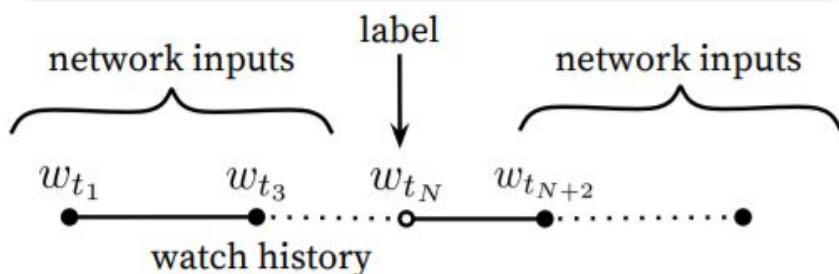


	user_id	query_movie_ids	query_movie_ids_len	candidate_movie_id	rating
0	47	[263, 2860, 3856, 6034, 2409, 2374, 7128, 5339...]	357	1393	4.5
1	175	[966, 5026, 4395, 6042, 3871, 6521, 5020, 953,...]	177	4002	3.0
2	261	[45, 1104, 154, 4514, 4171, 3801, 3727, 3644, ...]	471	3089	2.0

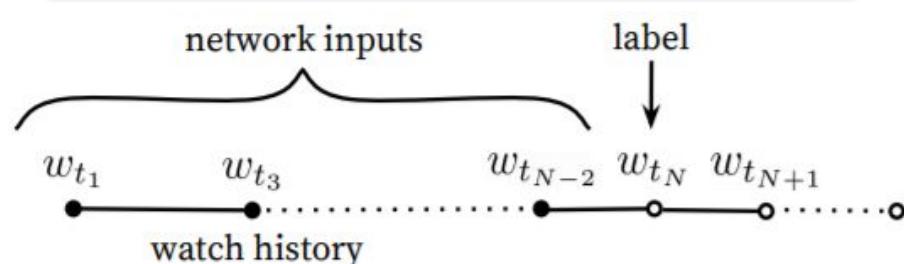
□ 觀察47號user

- query_movie_ids: train data中47號的有rating的movie id, 排除1393號
⇒ leave one out (這裡的設計也可以只用47號喜歡的電影)

leave one out for predict



leave one out(考慮時間)



Recommendation Engine in Matrix Factorization

Model - MF with History



user_id	query_movie_ids	query_movie_ids_len	candidate_movie_id	rating
0 47	[263, 2860, 3856, 6034, 2409, 2374, 7128, 5339...]	357	1393	4.5
1 175	[966, 5026, 4395, 6042, 3871, 6521, 5020, 953,...]	177	4002	3.0
2 261	[45, 1104, 154, 4514, 4171, 3801, 3727, 3644, ...]	471	3089	2.0

□ 觀察47號user

- query_movie_ids: train data中47號的有rating的movie id, 排除1393號
⇒ leave one out (這裡的設計也可以只用47號喜歡的電影)
- query_movie_ids_len: 描述query_movie_ids movie id的個數(tensorflow 限制, mini batch裡面的變數長度必須要一致)

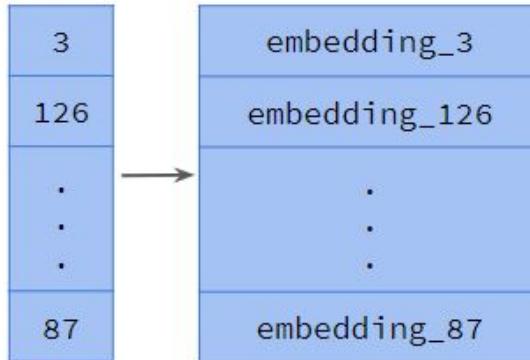
所以47號User在train data裡只有一筆資料了?

Recommendation Engine in Matrix Factorization

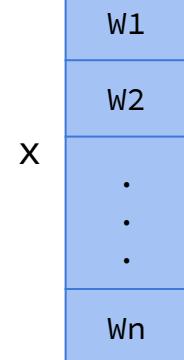
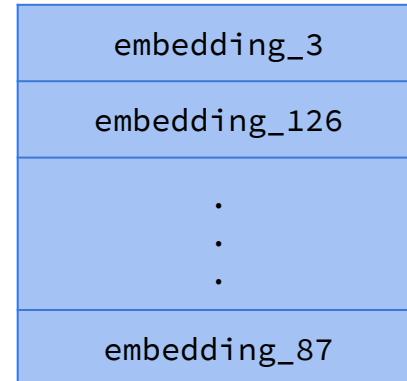
Movie ID Aggregation(Pooling) ?



MID History



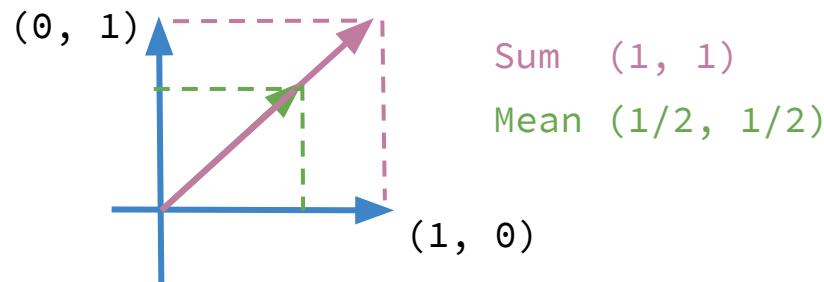
aggregation



Tensorflow documentation上找到處理multivalent embedding的方式

Sum	embedding向量相加
Mean	embedding向量相加後平均
SqrtN	將weight normalize後對embedding做weighted sum

在這批資料中效果較好

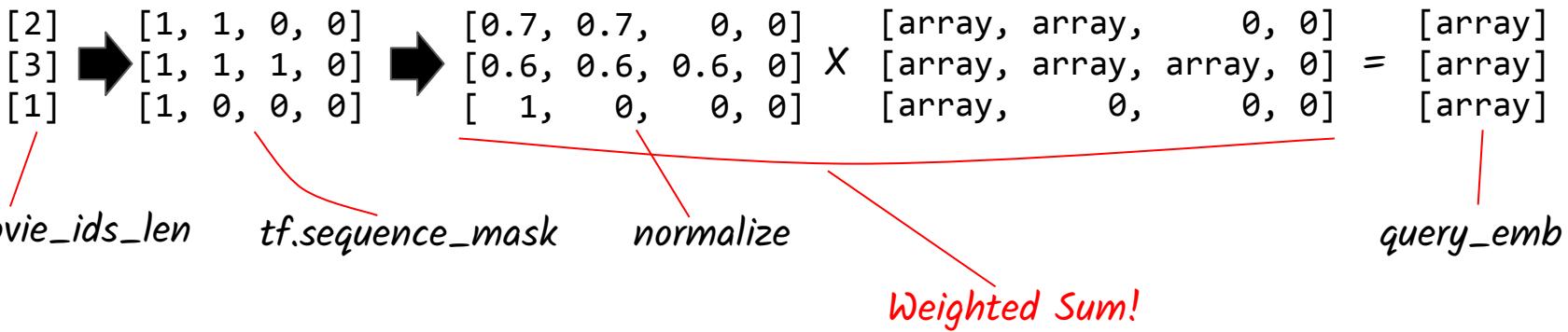


Recommendation Engine in Matrix Factorization

MF with History - Aggregation(Pooling)



```
with tf.variable_scope("computation"):  
    # weighted avg of query embedding  
    '''sqrtn aggregation(pooling), X: data, W: weight  
        X_1*W_1 + X_2*W_2 + ... + X_n*W_n / sqrt(W_1**2 + W_2**2 + ... W_n**2)  
        = weighted sum of X and normalized W  
        here data = self.query_emb, weight = weighted'''  
  
    query_emb_mask = tf.sequence_mask(self.query_movie_ids_len)  
    weighted = tf.nn.l2_normalize(tf.to_float(query_emb_mask), 1)[:, :, tf.newaxis]  
    self.query_emb = tf.reduce_sum(self.query_emb * weighted, 1)  
    self.query_bias = tf.matmul(self.query_emb, self.b_query_movie_ids[:, tf.newaxis])
```



Recommendation Engine in Matrix Factorization

MF with History - Prediction (graph_computation)



```
infer = tf.reduce_sum(self.query_emb * self.candidate_emb, 1, keep_dims=True)
infer = tf.add(infer, self.b_global)
infer = tf.add(infer, self.query_bias)
self.infer = tf.add(infer, self.candidate_bias, name="infer")
```

```
# one query for all items
self.pred = tf.matmul(self.query_emb, tf.transpose(self.w_candidate_movie_id)) + \
            tf.reshape(tf.matmul(self.w_candidate_movie_id,
                                self.b_candidate_movie_id[:, tf.newaxis]), (1, -1)) + \
            self.query_bias + \
            self.b_global
```

- ❑ infer節點與pred節點做的事情是一樣的，都是帶下列的公式

$$f_{u,m}(x) = u_i \cdot m_j + b_u + b_m + b_{global}$$

- ❑ pred節點純粹是為了執行效能而已

Recommendation Engine in Matrix Factorization

MF with History - Metrics and Comparison



Model	MF	MF with history
RMSE Loss	0.92	0.92
ROC AUC	0.74	0.74
NDCG	strict: 0.63 normal: 0.76	strict: 0.63 normal: 0.76
Precision at 10	strict: 0.50 normal: 0.63	strict: 0.51 normal: 0.63

- MF with history的Model效果沒有進步
- 但後者的Model較有彈性!

Recommendation Engine in Matrix Factorization



Lab: lab_reco_model_mf_with_history.ipynb (20 minutes)

搜尋 ”Do:” 就可以找到可能要修改的位置

lab filename	lab_reco_model_mf_with_history.ipynb
target	您可能要修改的參數: <ul style="list-style-type: none"><input type="checkbox"/> learning rate<input type="checkbox"/> dim: dimension of latent factor(user and movie)<input type="checkbox"/> reg: 正規項的強度<input type="checkbox"/> dropout<input type="checkbox"/> loss function<input type="checkbox"/> optimizer
ideal score	valid loss: 降低到0.92左右 roc auc: 0.74左右 ndcg: strict condition 0.63左右

Recommendation Engine with DNN

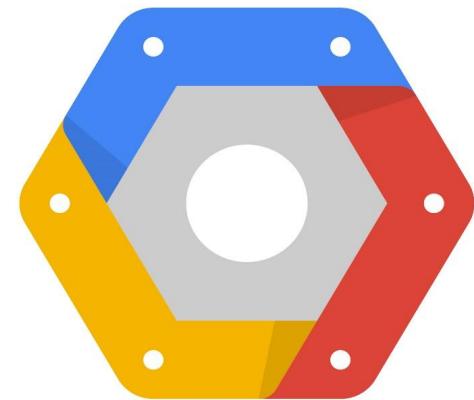


Machine Learning Framework

Deep Neural Network(DNN) Tips

Recommender Engine in Matrix Factorization

Matrix Factorization with DNN



Recommendation Engine with DNN



Model - MF with DNN

- 同樣的想法，也來改進一下**Movie Latent Factor**

userId	movieId	title	genres	rating
0 0	30	Dangerous Minds (1995)	Drama	2.5
1 0	833	Dumbo (1941)	Animation Children Drama Musical	3.0
2 0	859	Sleepers (1996)	Thriller	3.0
3 0	906	Escape from New York (1981)	Action Adventure Sci-Fi Thriller	2.0
4 0	931	Cinema Paradiso (Nuovo cinema Paradiso) (1989)	Drama	4.0

- Movie Meta好像太少了點
 - title欄位有**年份資料**
 - genres告訴我們一部電影可以有多種類型
 - rating是user對電影的評分，一部電影被多個user評分
⇒ 可以算出**平均評分**

Recommendation Engine with DNN

Model – MF with DNN



- 於是電影可以用下列Meta來描述

genres	genres_len	avg_rating	year	candidate_movie_id	rating
[1, 0, 3, 0]	3	0.716049	0.833333	1393	4.5
[5, 13, 10, 1]	4	0.643739	0.877193	4002	3.0
[0, 7, 2, 0]	3	0.545752	0.859649	3089	2.0
[7, 11, 2, 0]	3	0.448148	0.833333	1308	3.0
[1, 0, 0, 0]	1	0.779449	0.736842	2340	1.5

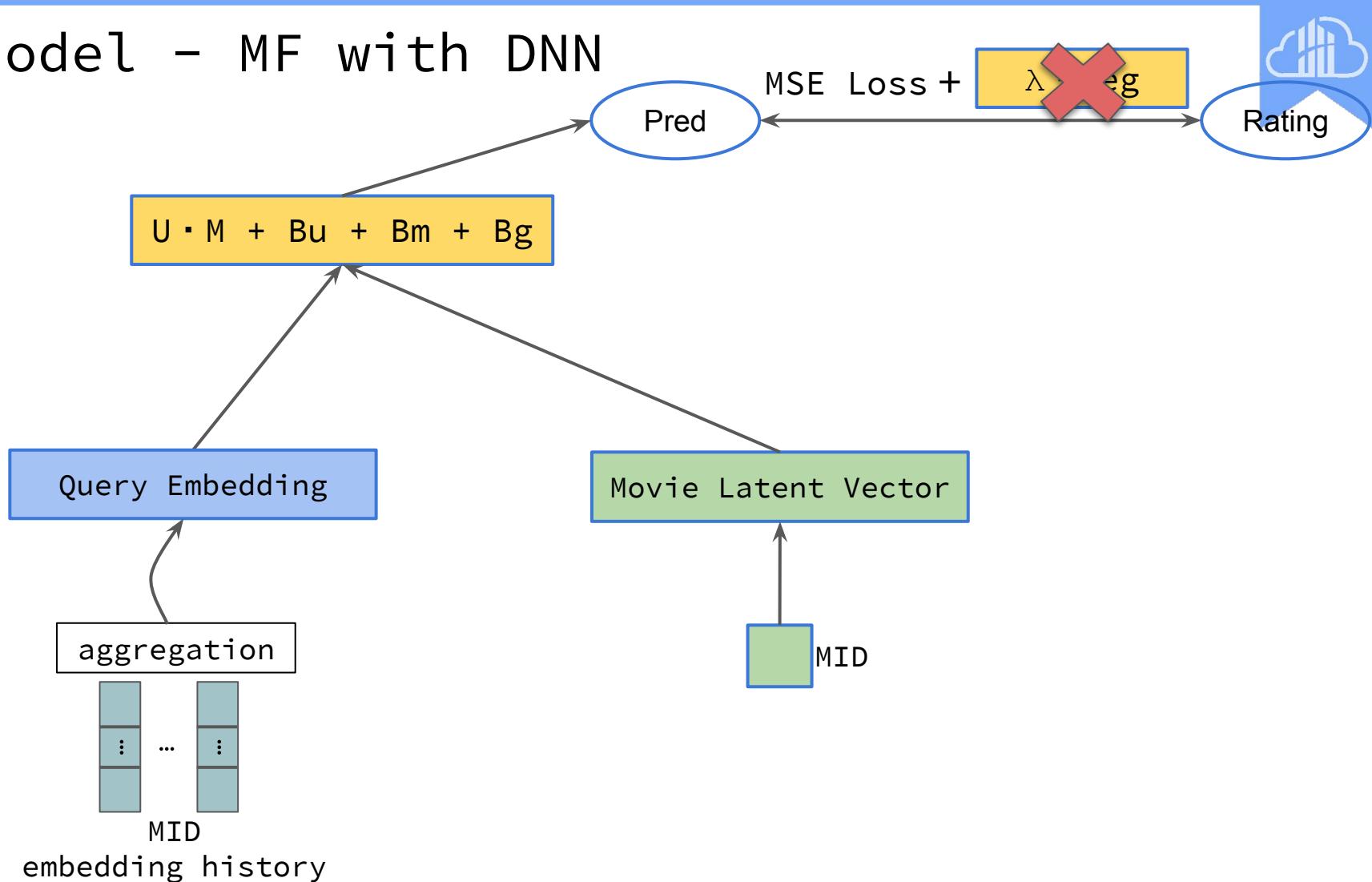
- genres: multivalent variable \Rightarrow embedding \Rightarrow average
- genres_len: 描述genres長度
- avg_rating: 電影的平均評分(已normalize to 0 ~ 1之間)
- year: 從title extract出來(已normalize to 0 ~ 1之間)

Enrich movie metadata

- 還可以Crawler搜尋電影導演、演員、票房等等...

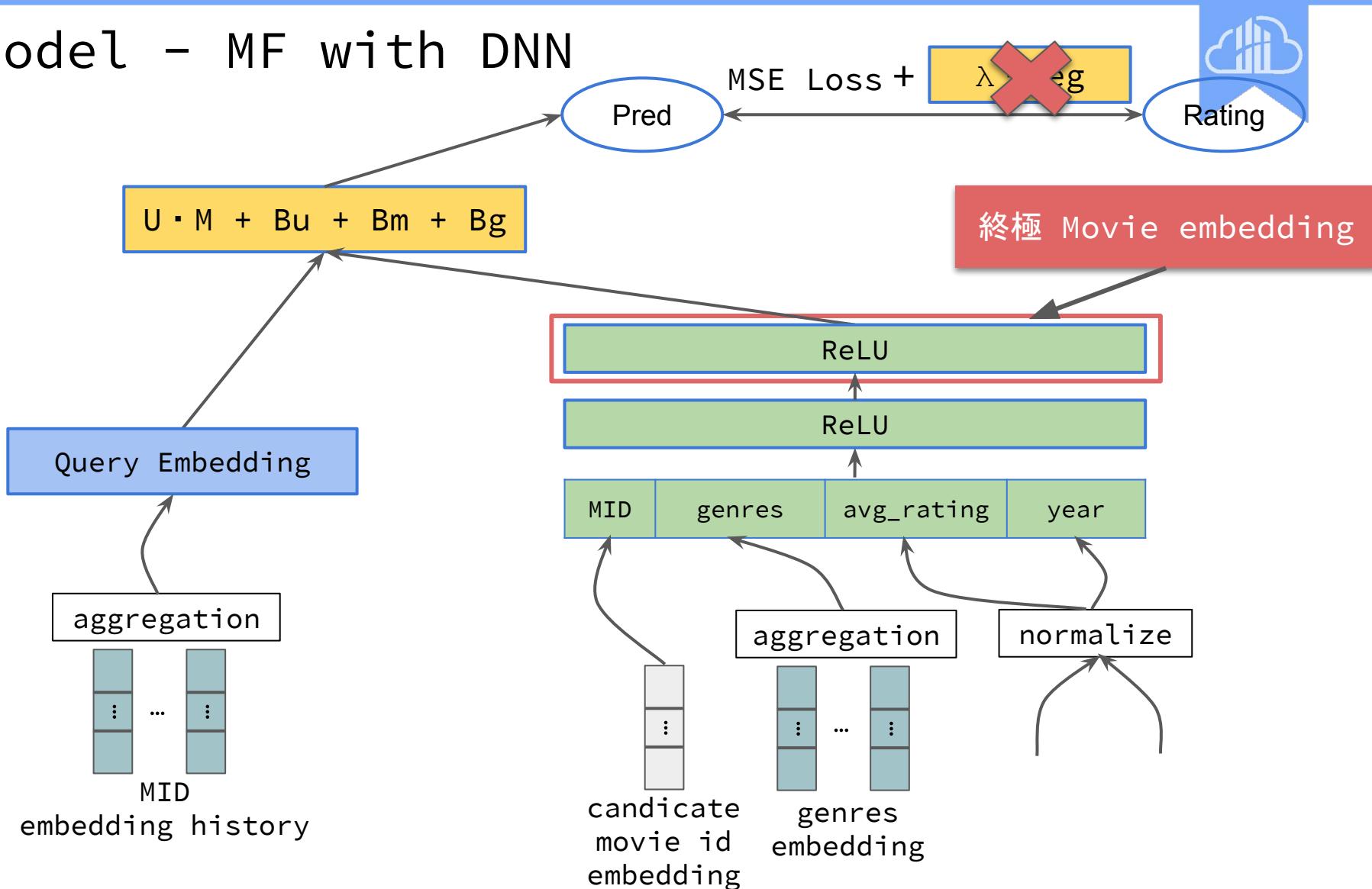
Recommendation Engine with DNN

Model – MF with DNN



Recommendation Engine with DNN

Model – MF with DNN



Recommendation Engine with DNN

Model - MF with DNN



filename	reco_model_mf_dnn.ipynb
lab filename	lab_reco_model_mf_dnn.ipynb
number of users	671
number of movies	9125
rating range	0.5 ~ 5分, <input type="checkbox"/> 4分以上算正評 <input type="checkbox"/> 未滿4分認為是負評或是不列入推薦名單
neural network	matrix factorization with dnn

Recommendation Engine with DNN

Model - MF with DNN(graph_dnn)



```
with tf.variable_scope("dnn"):  
    # encode [item embedding + item metadata]  
    self.item_repr = tf.concat([self.candidate_emb, self.genres_emb, self.avg_rating[:, tf.newaxis], self.year[:, tf.newaxis]], 1)  
    self.candidate_bias = tf.matmul(self.item_repr, self.b_candidate_movie_id[:, tf.newaxis])  
  
    # dp_scale = 0.5  
    self.item_repr = tf.layers.dense(self.item_repr, dim, kernel_initializer=init_fn, activation=tf.nn.relu)  
    # self.item_repr = tf.layers.dropout(self.item_repr, dp_scale, training=self.isTrain)  
    self.item_repr = tf.layers.dense(self.item_repr, dim, kernel_initializer=init_fn, activation=tf.nn.relu)  
    # self.item_repr = tf.layers.dropout(self.item_repr, dp_scale, training=self.isTrain)
```

- item_repr: concat metadata
 - candidate movie id embedding (16) +
 - genres embedding (8) +
 - 平均分數 (1) +
 - 年份 (1)
$$\Rightarrow \text{shape} = 16 + 8 + 1 + 1 = 26$$
- 兩層DNN(fully connected layer) + ReLU \Rightarrow encoding
- **這裡的Network structure可以自行調整(增加層數 or 調整neuron數)**

Recommendation Engine with DNN

Model - MF with DNN

Trainging



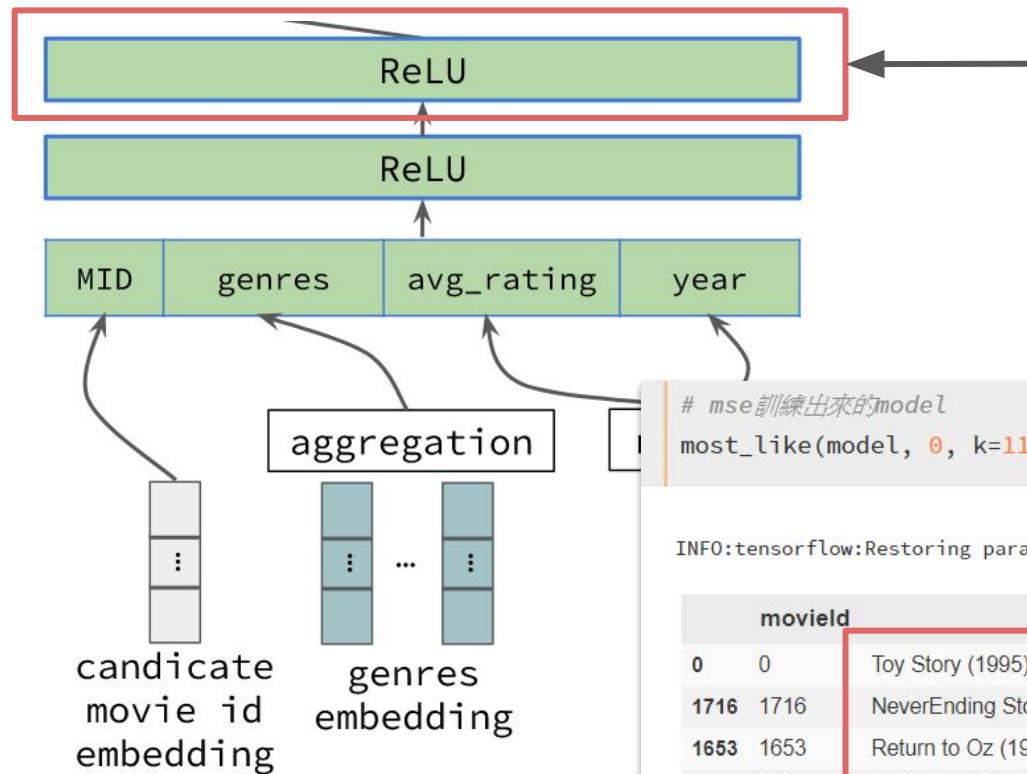
```
# !cp -a ./model/model_mf_with_dnn_bak/* ./model/model  
# 每個batch的數量  
n_batch = 128  
with tf.Session(graph=model.graph) as sess:  
    model.fit(sess, dataFn(trProcessed, n_batch))
```

Epoch	Train Error	Val Error	Elapsed Time
01	1.096	0.934	14.236 secs, saving ...
02	0.961	0.894	15.311 secs, saving ...
03	0.931	0.870	14.697 secs, saving ...
04	0.910	0.854	13.682 secs, saving ...
05	0.894	0.844	
06	0.881	0.835	
07	0.870	0.827	
08	0.862	0.823	15.012 secs, saving ...
09	0.856	0.818	13.786 secs, saving ...
10	0.851	0.820	13.758 secs

Valid RMSE error已經可以低達0.81

Recommendation Engine with DNN

MF with DNN - 活用Latent Factor(Embedding)!



找出跟Toy Story相似的電影

- Indide Out
- Toy Story2
- Antz(螞蟻雄兵)

...

INFO:tensorflow:Restoring parameters from ./model/model_mf_with_dnn/model-9

movielid		title	genres
0	0	Toy Story (1995)	Adventure Animation Children Comedy Fantasy
1716	1716	NeverEnding Story, The (1984)	Adventure Children Fantasy
1653	1653	Return to Oz (1985)	Adventure Children Fantasy
8911	8911	Inside Out (2015)	Adventure Animation Children Comedy Drama Fantasy
1815	1815	Antz (1998)	Adventure Animation Children Comedy Fantasy
3217	3217	Emperor's New Groove, The (2000)	Adventure Animation Children Comedy Fantasy
7945	7945	Asterix and the Vikings (Astérix et les Viking...)	Adventure Animation Children Comedy Fantasy
2506	2506	Toy Story 2 (1999)	Adventure Animation Children Comedy Fantasy
3079	3079	The Spiral Staircase (1945)	Horror Mystery Thriller
813	813	Escape to Witch Mountain (1975)	Adventure Children Fantasy
6246	6246	MirrorMask (2005)	Adventure Children Drama Fantasy

Recommendation Engine with DNN

Model - 實際上使用Embedding的例子



使用User embedding找出該user偏好的衣服類型



甚至可以找不同類型的搭配(馬克思禮服找到一堆寬腿褲)



<https://making.dia.com/embedding-everything-for-anything2anything-recommendations-fca7f58f53ff>

Recommendation Engine with DNN

MF with DNN – Metrics and Comparison



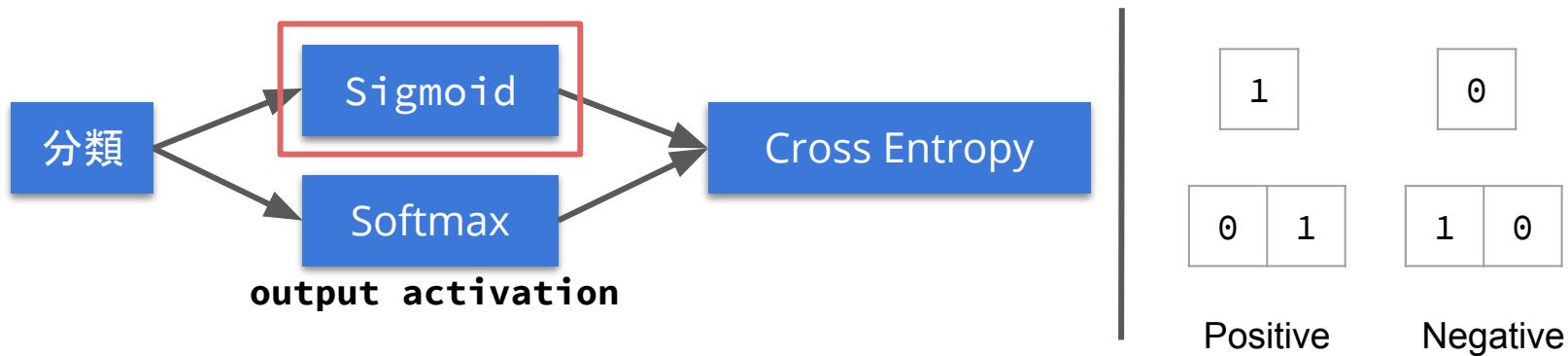
Model	MF	MF with history	MF with DNN
RMSE Loss	0.92	0.92	0.82
ROC AUC	0.74	0.74	0.80
NDCG	strict: 0.63 normal: 0.76	strict: 0.63 normal: 0.76	strict: 0.73 normal: 0.86
Precision at 10	strict: 0.50 normal: 0.63	strict: 0.51 normal: 0.63	strict: 0.59 normal: 0.68

Recommendation Engine with DNN



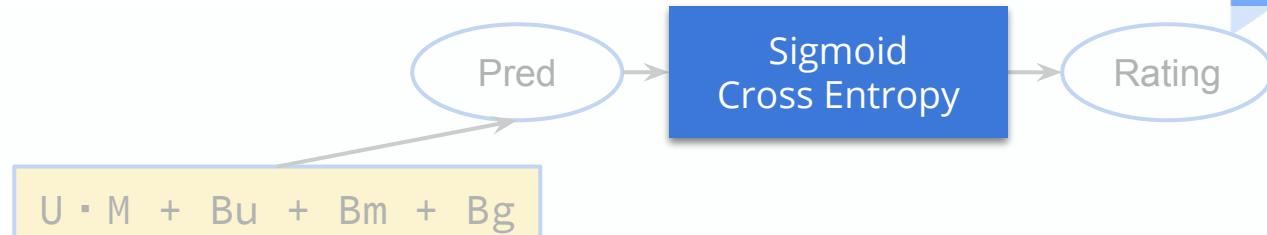
Model - MF with DNN(Classification)

- Movie rating: explicit feedback
- **通常User不會這麼熱心的對你的商品評價**
 - implicit feedback
 - 商品點閱次數
 - Youtube影片觀看時間
 - ...
 - ⇒ 通常只能得到1(Positive), 0(Negative)
- **Label 0.5 ~ 5 ⇒ 1 or 0 (這裡4分以上 = 1 others 0)**
 - Regression ⇒ Binary classification
 - $P(\text{使用者A 喜歡 電影B}) = ??$



Recommendation Engine with DNN

Model – MF with DNN(Classification)

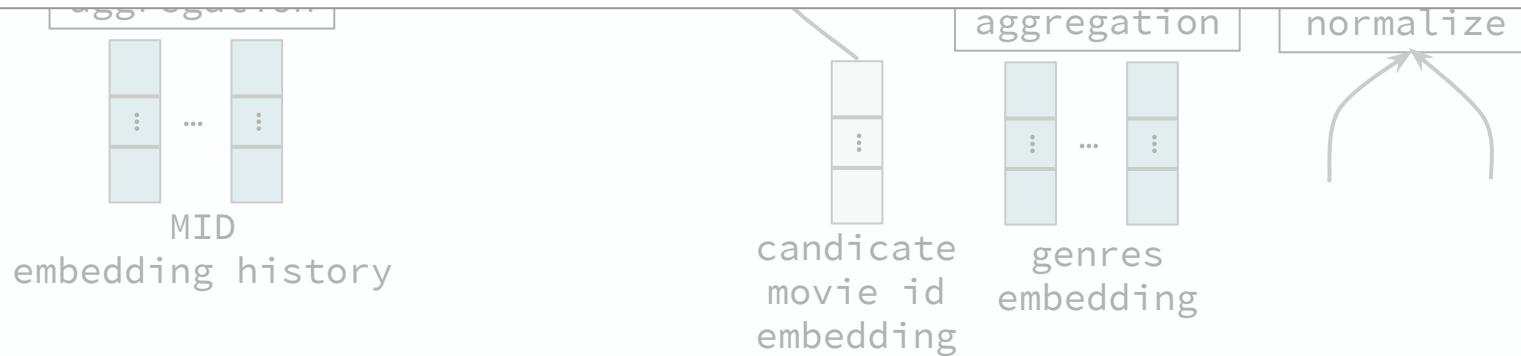


❑ Function set

$$f_{u,m}(x) = \sigma(u_i \cdot m_j + b_u + b_m + b_{global})$$

❑ Loss function: cross entropy

$$L = - \sum (\hat{y} \cdot \ln f_{u,m}(x) + (1 - \hat{y})(1 - \ln f_{u,m}(x)))$$



Recommendation Engine with DNN

Model - MF with DNN



```
with tf.variable_scope("loss"):  
    # if rating >= 4 then 1 else 0  
    self.alter_rating = tf.to_float(self.rating >= 4)  
    self.loss = tf.reduce_mean(tf.nn.sigmoid_cross_entropy_with_logits(labels=self.alter_rating[:, np.newaxis], logits=self.infer))  
  
    # for eval  
    self.rmse_loss = tf.sqrt(tf.losses.mean_squared_error(labels=self.alter_rating[:, tf.newaxis], predictions=self.infer))  
    self.mae_loss = tf.reduce_mean(tf.abs(self.infer - self.alter_rating[:, tf.newaxis]))  
pass
```

- ❑ alter_rating: 在**tensorflow**裡轉換label
 - ❑ if rating >= 4 ⇒ 1, else 0

Recommendation Engine with DNN

Model - MF with DNN

Trainging



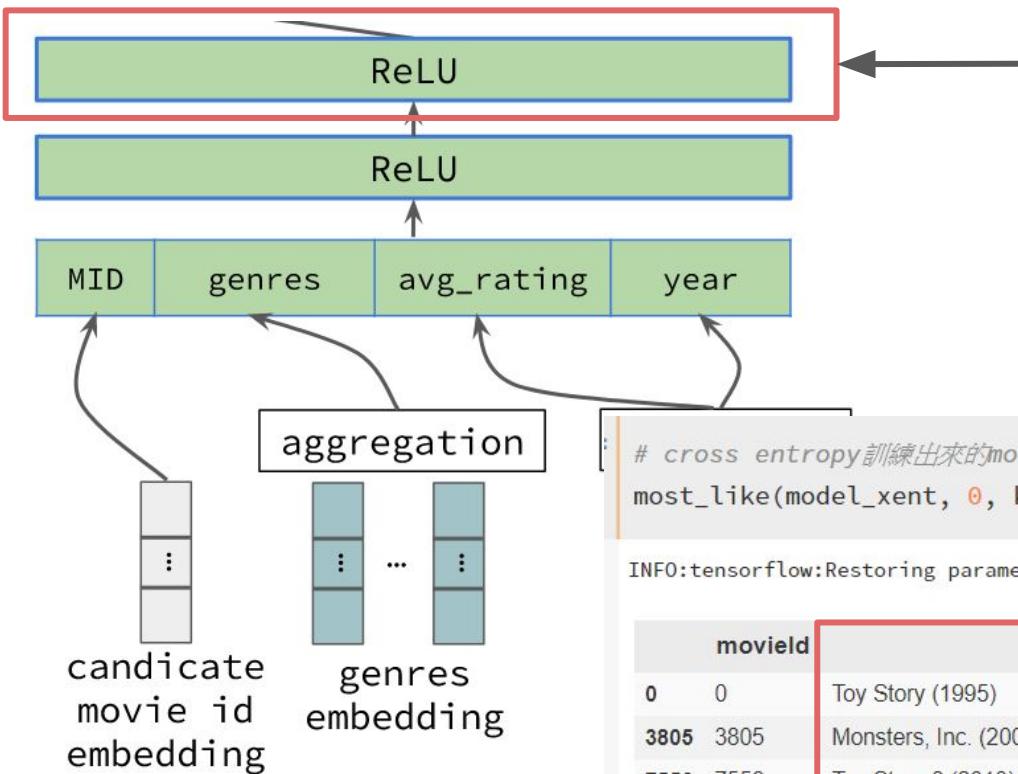
```
n_batch = 128
with tf.Session(graph=model_xent.graph) as sess:
    model_xent.fit(sess, dataFn(trProcessed, n_batch=n_batch, sh
```

Epoch	XEntropy Error	Val XENT Error	Val RMSE Error	Elapsed Time
01	0.653	0.622	0.745	15.133 secs, saving ...
02	0.607	0.590	0.888	15.839 secs, saving ...
03	0.580	0.571	0.930	15.542 secs, saving ...
04	0.562	0.559	0.986	14.085 secs, saving ...
05	0.551	0.566	1.271	14.290 secs
06	0.542	0.552	1.190	17.491 secs, saving ...
07	0.536	0.547	1.241	14.598 secs, saving ...
08	0.530	0.539	1.200	15.505 secs, saving ...
09	0.524	0.534	1.271	14.136 secs, saving ...
10	0.520	0.535	1.447	

- ❑ 最佳化cross entropy
- ❑ RMSE的分數與之前的Model相比是無意義的 ⇒ Label值域已經轉變!

Recommendation Engine with DNN

MF with DNN(Cross Entropy), 活用Embedding



找出跟Toy Story相似的電影

- Monsters, Inc
- Toy Story3
- Toy Story2

...

```
# cross entropy訓練出來的model  
most_like(model_xent, 0, k=11)
```

```
INFO:tensorflow:Restoring parameters from ./model/model_mf_with_dnn_xent\model-11
```

movielid	title	genres
0 0	Toy Story (1995)	Adventure Animation Children Comedy Fantasy
3805 3805	Monsters, Inc. (2001)	Adventure Animation Children Comedy Fantasy
7556 7556	Toy Story 3 (2010)	Adventure Animation Children Comedy Fantasy IMAX
7373 7373	Black Dynamite (2009)	Action Comedy
7343 7343	Haunted World of El Superbeasto, The (2009)	Action Animation Comedy Horror Thriller
3419 3419	Shrek (2001)	Adventure Animation Children Comedy Fantasy Ro...
215 215	Gordy (1995)	Children Comedy Fantasy
7653 7653	Red (2010)	Action Comedy
7525 7525	Exit Through the Gift Shop (2010)	Comedy Documentary
2506 2506	Toy Story 2 (1999)	Adventure Animation Children Comedy Fantasy
1656 1656	Sleeping Beauty (1959)	Animation Children Musical

Recommendation Engine with DNN

MF with DNN - Metrics and Comparison



Model	MF	MF with history	MF with DNN(MSE)	MF with DNN (Cross Entropy)
RMSE Loss	0.92	0.92	0.82	1.44 cross entropy loss: 0.53(無法與RMSE相比)
ROC AUC	0.74	0.74	0.80	0.81
NDCG	strict: 0.63 normal: 0.76	strict: 0.63 normal: 0.76	strict: 0.73 normal: 0.83	strict: 0.72 normal: 0.82
Precision at 10	strict: 0.50 normal: 0.63	strict: 0.51 normal: 0.63	strict: 0.59 normal: 0.68	strict: 0.59 normal: 0.68

Recommendation Engine with DNN



Lab: lab_reco_model_mf_dnn.ipynb (20 minutes)

搜尋 ”Do:” 就可以找到可能要修改的位置

lab filename	lab_reco_model_mf_dnn.ipynb
target	您可能要修改的參數: <ul style="list-style-type: none"><input type="checkbox"/> learning rate<input type="checkbox"/> dim: dimension of latent factor(user and movie)<input type="checkbox"/> dropout<input type="checkbox"/> loss function<input type="checkbox"/> optimizer<input type="checkbox"/> DNN structure
ideal score	valid rmse loss: 降低到0.82左右 roc auc: 0.80左右 ndcg: strict condition 0.60左右



Model MF with DNN 彈性的應用

- ❑ 對於新的user
 - ❑ 無使用紀錄
 - ❑ 推薦最歡迎商品
 - ❑ 一點點使用紀錄
 - ❑ Content base recommendation(trained item embedding)
 - ❑ 大量使用紀錄
 - ❑ Model recommendation
- ❑ 對於新進來的電影
 - ❑ 將metadata帶入model取得embedding就可以使用
 - ❑ ex: genres、avg_rating、year、director、actors(actresses)...
 - ❑ 不使用指向性的Feature ⇒ Ex: Movie ID



- ❑ 若有課程上的建議，歡迎來信
 - ❑ gary.chen@mile.cloud
 - ❑ johann.chu@mile.cloud
 - ❑ jeff.liu@mile.cloud



填寫問卷
便可索取課堂參考資料連結



CloudMile 萬里雲

