DLP Lab1 Backpropagation

0711529 陳冠儒

1. Introduction (20%)

   建造一個具有兩層 hidden layers 的 model，並計算 forward 和 backward propagation，來預測 Linear 和 XOR dataset。

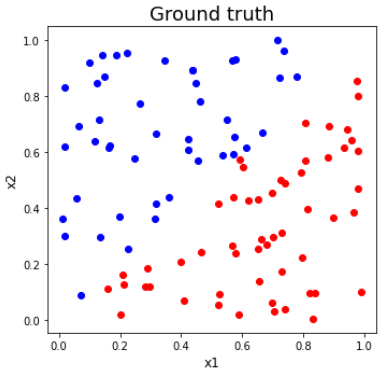
   Dataset：

   x: shape = (number of data, input dimension)

   y: shape = (number of data, label)

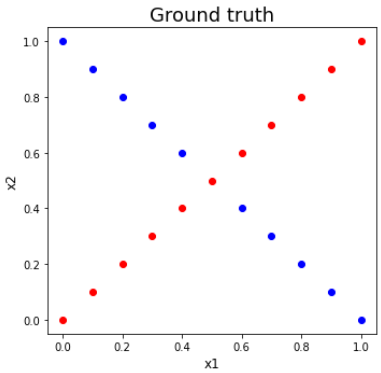
   (1.) Linear

   - label = 0, x1 > x2 which falls on the bottom-right side.
   - label = 1, x1 < x2 which falls on the upper-left side.



   (2.) XOR

   - label = 0, x2 = x1
   - label = 1, x2 = 1 - x1

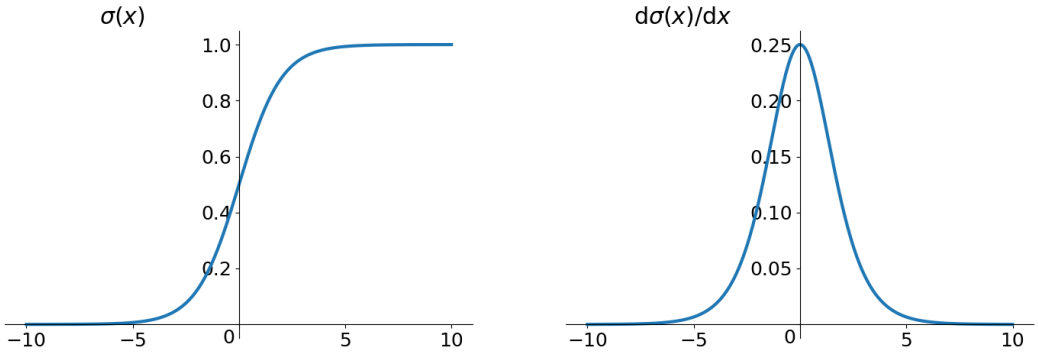


2. Experiment setups(30%):

   A. Sigmoid functions

   - 簡介

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



Sigmoid functions 可以用來當作 hidden layer 的 activation function 來使其成 non-linear，也可以用在 output layer 使其成 Bernoulli output distributions。

- Sigmoid function 在數學上的介紹
  Sigmoid function 是 logistic function 的一個特例。

$$logistic(\text{x}) = \frac{L}{1+e^{-k(x-x_0)}}$$

L 控制最大最小值、k 為最小到最大的變化速率、x_0為 x 的偏移量，而 sigmoid function 就是當 L = 1、k = 1 且x_0=0，故在 output layer 用 sigmoid function 的二分法 model 也被稱為 logistic regression。

- Derivation of sigmoid function

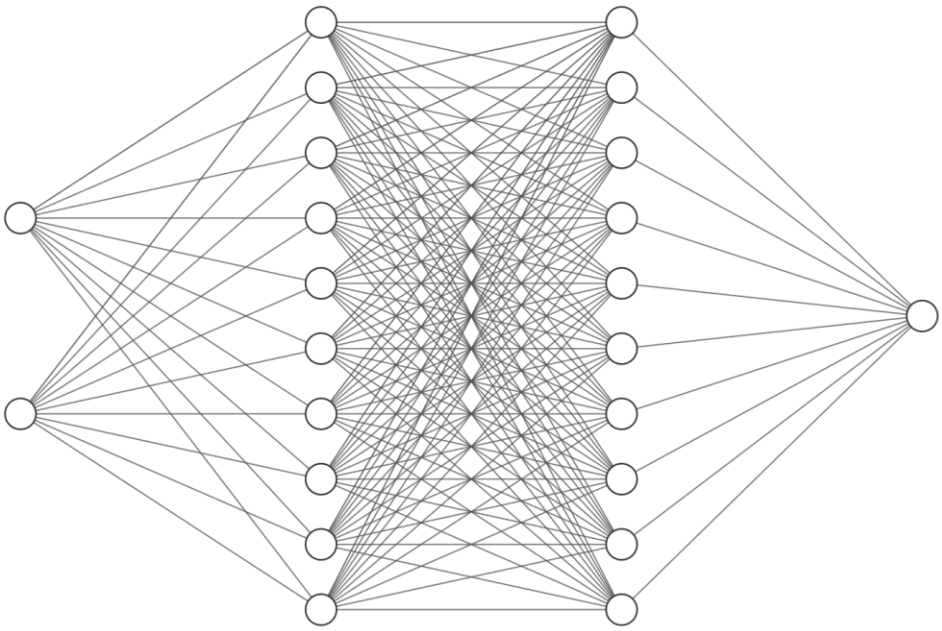
$$\frac{\text{d}\sigma}{\text{dx}} = \frac{d}{dx}\left(\frac{1}{1+e^{-x}}\right)$$
$$= \frac{e^{-x}}{(1+e^{-x})^2}$$
$$= \frac{1}{(1+e^{-x})}\frac{e^{-x}}{(1+e^{-x})}$$
$$= \frac{1}{(1+e^{-x})}\frac{e^{-x}}{(1+e^{-x})}$$
$$= \frac{1}{(1+e^{-x})}\frac{(1+e^{-x})-1}{(1+e^{-x})}$$
$$= \frac{1}{(1+e^{-x})}\left[1-\frac{1}{(1+e^{-x})}\right]$$

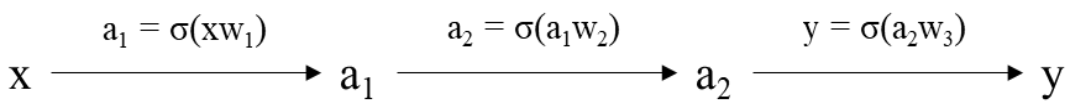
$$= \text{sigmoid}(x)[1 - \text{sigmoid}(x)]$$

- Sigmoid 優點
  中央區的信號增益較大，對兩側區的信號增益小，在信號的特徵空間映射上佳，學習時可以將重點特徵推向中央區，將非重點特徵推向兩側區。

- Sigmoid 缺點
  (1) 均值不為 0，意味著自帶了一個 bias，在計算時是額外的負擔，這會使得收斂變得更慢。
  (2) 在其飽和區會變化的很緩慢，微分趨近於 0，容易造成梯度消失，使得訊息丟失。

B. Neural network





- Neural Architecture & parameters design
  - ✓ hidden units (10, 10)
  - ✓ learning rate 0.01
  - ✓ Epoch 150000
  - ✓ Training data number: (1.) linear: 100, (2.) XOR: 21

✓ Testing data number: (1.) linear: 100, (2.) XOR: 61

- Loss Function

$$\mathrm{L} = -\frac{1}{m}\sum_{m} ylog(y_{pred}) + (1-y)\log(1-y_{pred})$$

我使用 Maximum Likelihood 來當作 loss function，上式所代表的是兩個 Bernoulli distribution 的 cross entropy，即算出 y distribution 和 y_{pred} distribution 的距離，minimizing cross entropy 就是希望這兩個越接近越好。

C. Backpropagation

- $\frac{\partial \mathrm{L}}{\partial \mathrm{w}_3} = \frac{\partial y_{pred}}{\partial w_3}\frac{\partial L}{\partial y_{pred}}$, where

$$\frac{\partial L}{\partial y_{pred}} = -\frac{y}{y_{pred}} + \frac{1-y}{1-y_{pred}}$$

$$\frac{\partial y_{pred}}{\partial w_3} = a_2\sigma(a_2 w_3)[1-\sigma(\mathrm{a}_2\mathrm{w}_3)]$$

- $\frac{\partial \mathrm{L}}{\partial \mathrm{w}_2} = \frac{\partial a_2}{\partial w_2}\frac{\partial y_{pred}}{\partial a_2}\frac{\partial L}{\partial y_{pred}}$, where

$$\frac{\partial y_{pred}}{\partial a_2} = w_3\sigma(a_2 w_3)[1-\sigma(a_2 w_3)]$$

$$\frac{\partial a_2}{\partial w_2} = a_1\sigma(a_1 w_2)[1-\sigma(a_1 w_2)]$$

- $\frac{\partial \mathrm{L}}{\partial \mathrm{w}_1} = \frac{\partial a_1}{\partial w_1}\frac{\partial a_2}{\partial a_1}\frac{\partial y_{pred}}{\partial a_2}\frac{\partial L}{\partial y_{pred}}$, where

$$\frac{\partial a_2}{\partial a_1} = w_2\sigma(a_1 w_2)[1-\sigma(a_1 w_2)]$$

$$\frac{\partial a_1}{\partial w_1} = x\sigma(x w_1)[1-\sigma(x w_1)]$$

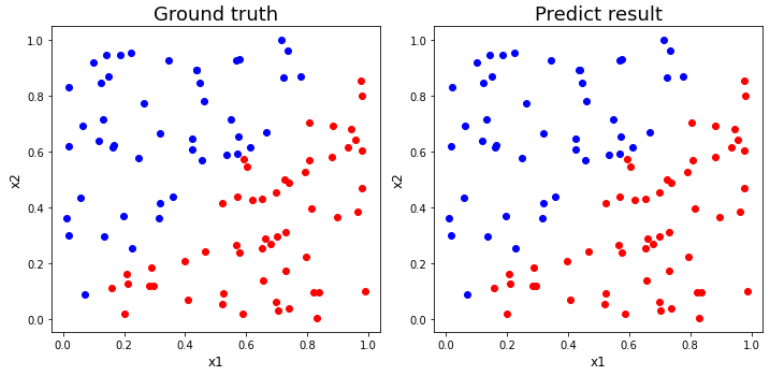
3. Results of your testing (20%)

A. Screenshot and comparison figure

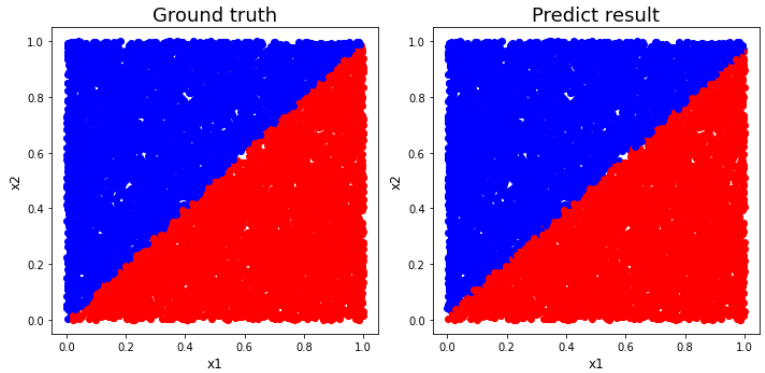
✓ Linear

Testing data generating by changing the random seed

- n_data = 100 → accuracy = 1.0



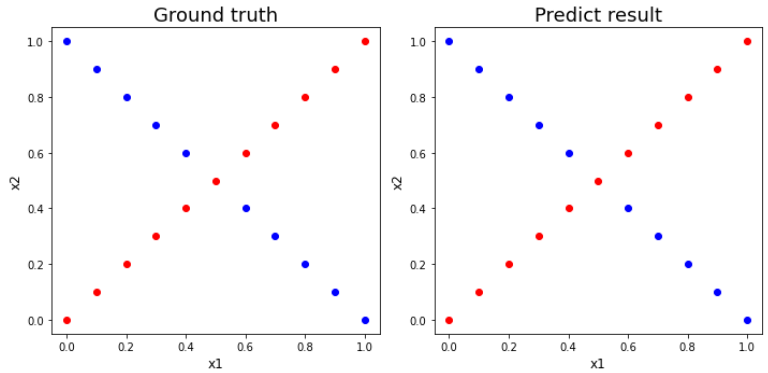
- n_data = 10000 → accuracy = 0.984



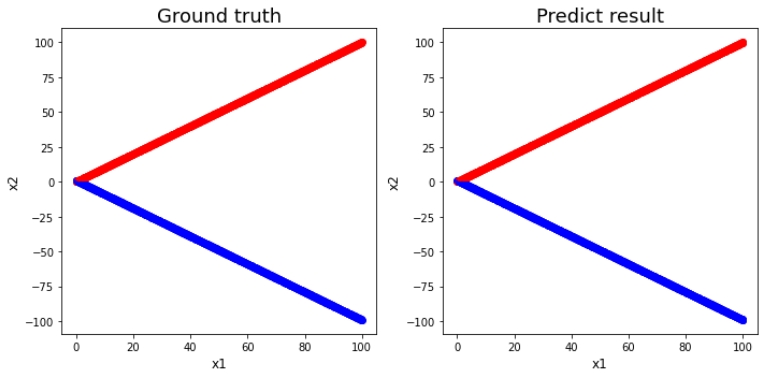
✓ XOR

Testing data generating by changing the number of points the function generated.

- n_data = 21 → accuracy = 1.0

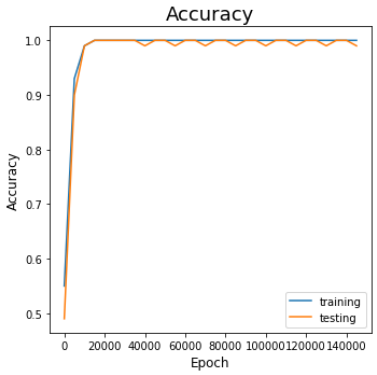


- n_data = 2001 → accuracy = 1.0

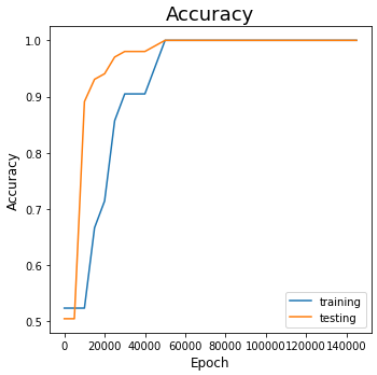


B. Show the accuracy of your prediction

✓ Linear

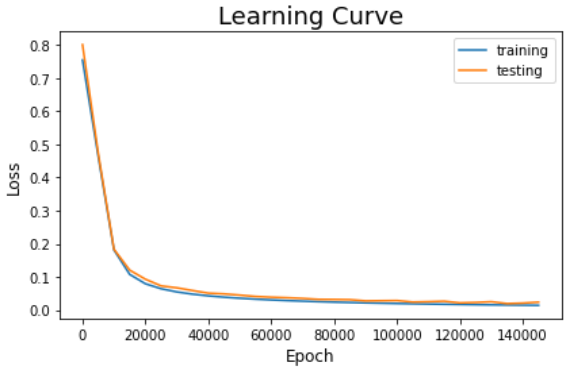


✓ XOR

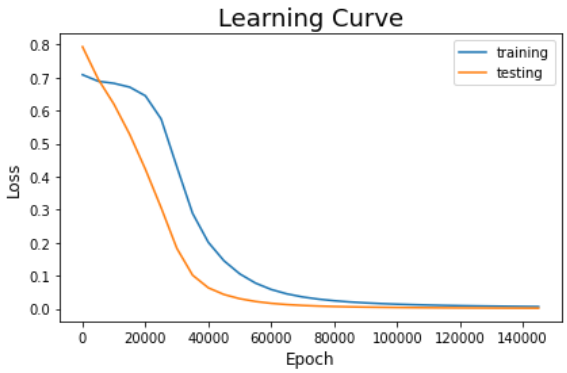


C. Learning curve (loss, epoch curve)

✓ Linear



✓ XOR



D. Anything you want to present

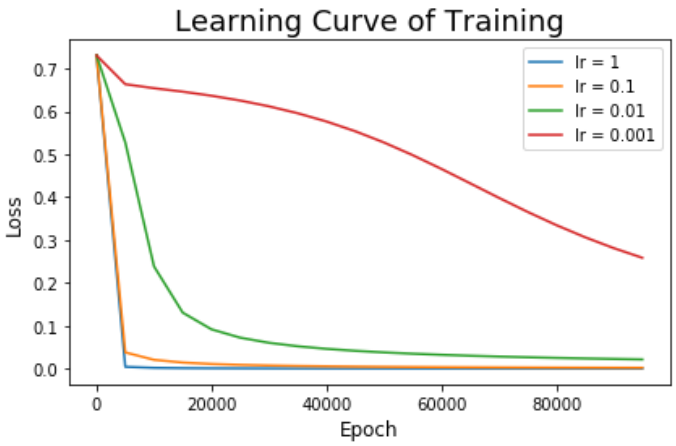
- Discuss about the learning curve and accuracy of XOR
  XOR function accuracy 跟 learning curve testing 都優於 training 的原因，是因為其 function 較為簡單學習，且 dataset 是依照同一個線性 function 所產生的，所以在 training 時，因為只有 21 個資料點，會由於樣本少，而使得每一個錯誤對 cost 和 accuracy 的影響被放大，但在 testing 中，因為資料點多(101 個資料點)，所以錯誤的影響會被稀釋。

4. Discussion (30%)

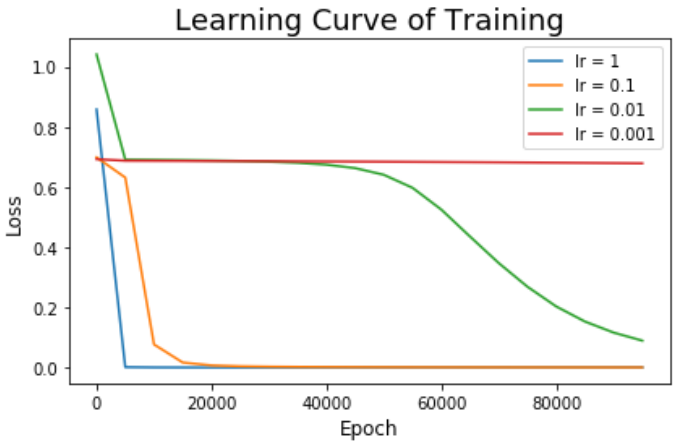
A. Try different learning rates

固定 Epoch 100000、hidden units (5, 5)。

(1.) Linear



(2.) XOR



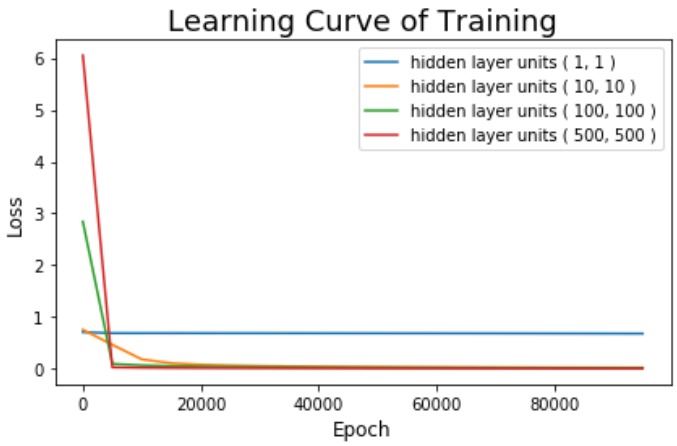
結論：從上面 linear 跟 XOR 不同 learning rate 造成的 learning curve of training 可以證實以下幾點。

a. learning rate 過大的話，他在一個 epoch 改變的很大，因此收斂的速度快，但是過大的 learning rate 可能因為步伐太大而無法收斂到最小點 (因為上面所求的 function 都較為簡單因此不會看到此現象)。
b. learning rate 過小的話，他每一個 epoch 改變的都非常小，因此需要很久很久才會收斂，且收斂時如果進入到了 local minimum 就可能無法走出去了。
c. learning rate 適中的話，收斂速度不慢，且可以進入到 global minimum。

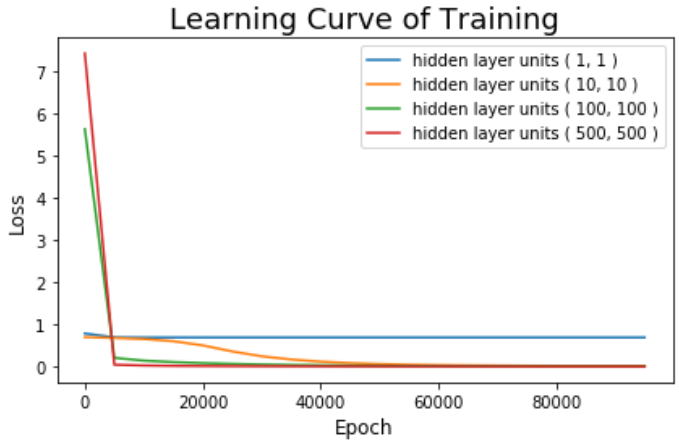
B. Try different numbers of hidden units

固定 Epoch 100000、Learning rate 0.01。

(1.) Linear



(2.) XOR



結論：從上面 linear 跟 XOR 不同 hidden layer units 造成的 learning curve of training 可以證實以下幾點。

a. hidden layer units 越多 neural network 的功能會越強大，能夠擬合越多的 function，故 training 時可以讓 loss 降得很低，但是也有可能會產生 overfitting，因此多的 hidden layer units 可以搭配著 regularization 一起使用，雖然會 train 比較久但是可以得到比較好的結果。

b. hidden layer units 太少的話，雖然因為運算可已很快，但是可能會造成 underfitting，因此最好選擇適中的 hidden layer units，以在運算時間和 fitting 結果中達到平衡。

C. Try without activation functions

$$x \xrightarrow{a_1=xw_1} a_1 \xrightarrow{a_2=a_1w_2} a_2 \xrightarrow{y=\sigma(a_2w_3)} y$$

Backpropagation

- $\frac{\partial \mathrm{L}}{\partial \mathrm{w}_3} = \frac{\partial y_{pred}}{\partial w_3}\frac{\partial L}{\partial y_{pred}}$, where

  $\frac{\partial L}{\partial y_{pred}} = -\frac{y}{y_{pred}} + \frac{1-y}{1-y_{pred}}$

  $\frac{\partial y_{pred}}{\partial w_3} = a_2\sigma(a_2 w_3)[1-\sigma(\mathrm{a}_2\mathrm{w}_3)]$

- $\frac{\partial \mathrm{L}}{\partial \mathrm{w}_2} = \frac{\partial a_2}{\partial w_2}\frac{\partial y_{pred}}{\partial a_2}\frac{\partial L}{\partial y_{pred}}$, where

  $\frac{\partial y_{pred}}{\partial a_2} = w_3\sigma(a_2 w_3)[1-\sigma(a_2 w_3)]$

  $\frac{\partial a_2}{\partial w_2} = a_1$

- $\frac{\partial \mathrm{L}}{\partial \mathrm{w}_1} = \frac{\partial a_1}{\partial w_1}\frac{\partial a_2}{\partial a_1}\frac{\partial y_{pred}}{\partial a_2}\frac{\partial L}{\partial y_{pred}}$, where

  $\frac{\partial a_2}{\partial a_1} = w_2$

  $\frac{\partial a_1}{\partial w_1} = x$

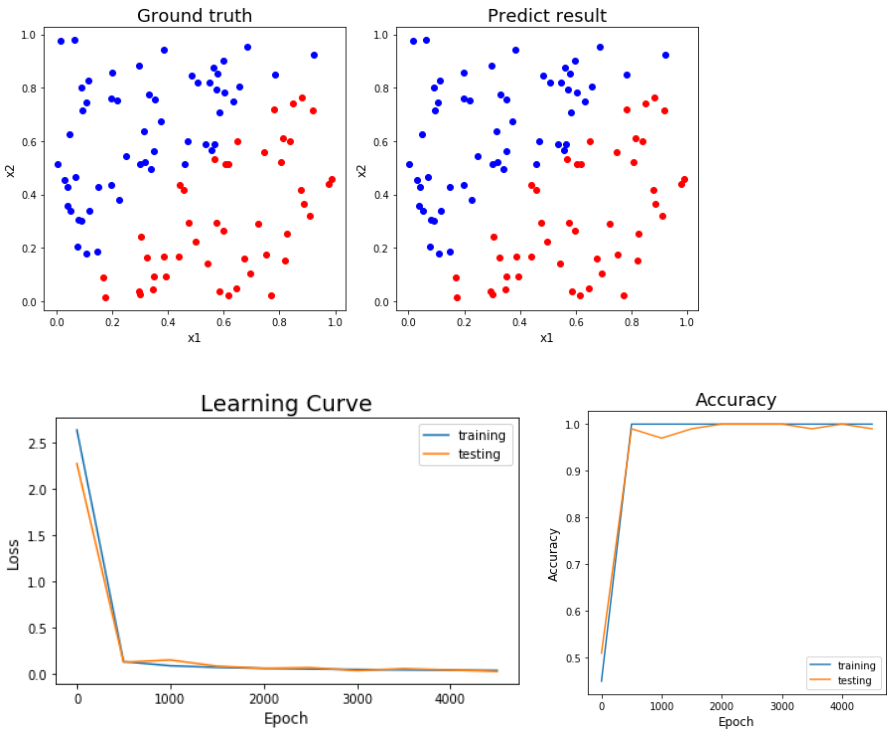
其他的參數使用同最上面的 neural network parameters 設定

- ✓ hidden units (10, 10)
- ✓ learning rate 0.01
- ✓ Training data number: (1.) linear: 100, (2.) XOR: 21
- ✓ Testing data number: (1.) linear: 100, (2.) XOR: 61

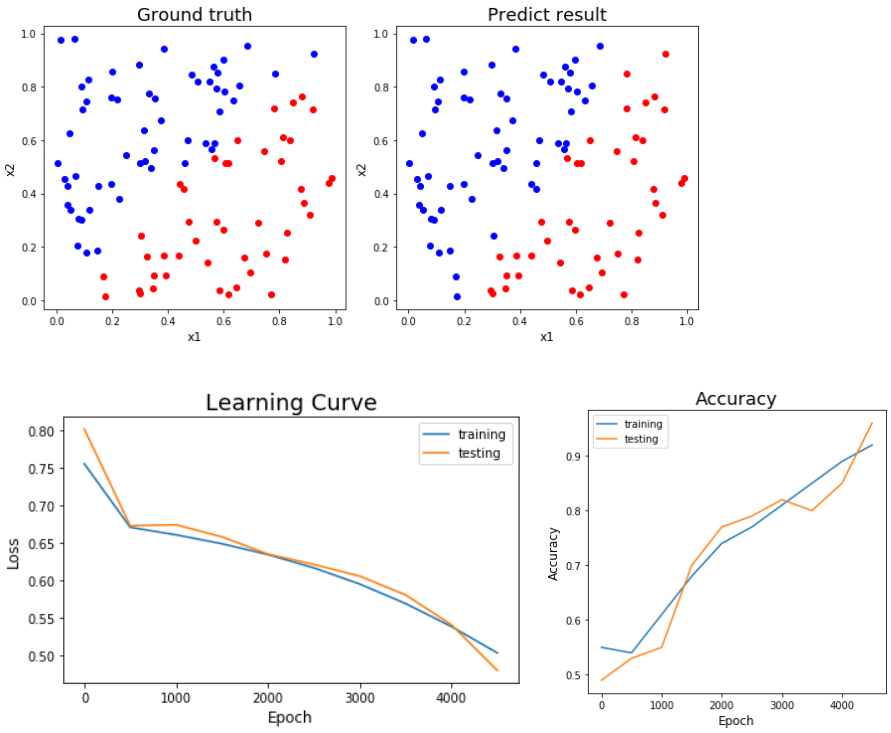
(1.) Linear

在小的 epoch 沒有 activation function 的 network 反而比有用 sigmoid 當 activation function 還有有效，因為這個 dataset 本身就是要訓練出一條 linear 的 function，所以是不用添加 sigmoid function 使其變成 non-linear 的，而添加 non-linear 的 activation function 後，也是可以訓練出來的，只是會需要更多次的 epoch 去 fit 他。

- Epoch 5000 without activation function



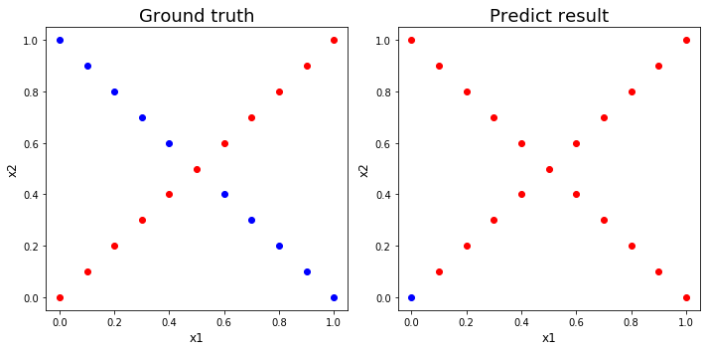
- Epoch 5000 with activation function sigmoid

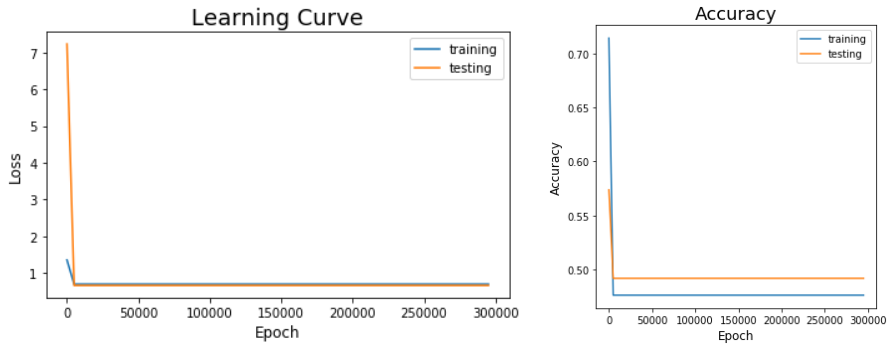


(2.) XOR

正如 AI winter 時，Minsky and Papert 的《Perceptrons》書中所述，XOR 因為不是線性的問題，所以不論用再多的 epoch 或是再小的 learning rate 去 train，都不會得到正確的答案。

- Epoch 30000 without activation function





D. Anything you want to share

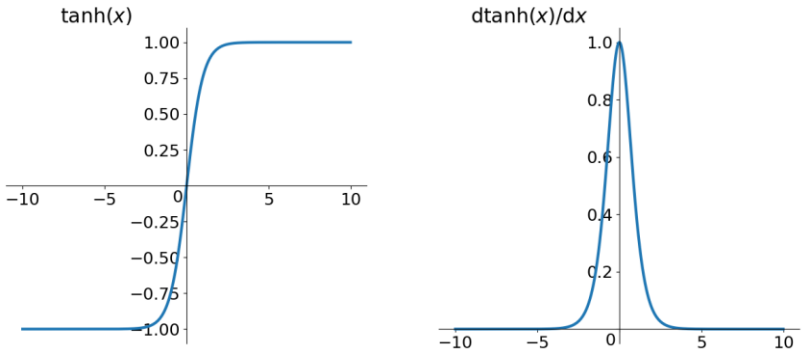
- Use Different Activation Function at hidden layer

  a. tanh

$$\tanh(x) = \frac{\sinh(x)}{\cosh(x)}$$

  - Derivation of sigmoid function

$$\frac{\mathrm{d}}{dx}\tanh(x) = \operatorname{sech}^2(x) = \frac{1}{\cosh^2(x)}$$



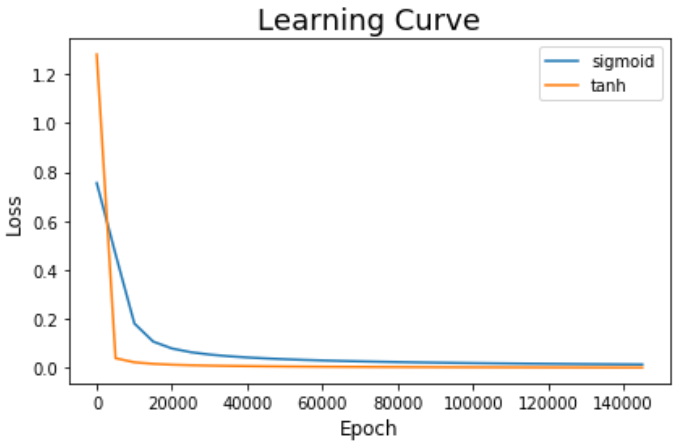
  - tanh 優點

    zero-centered，其均值是 0，沒有自帶 bias，在計算時是額外負擔少，這會使得收斂變得更快。

  - tanh 缺點

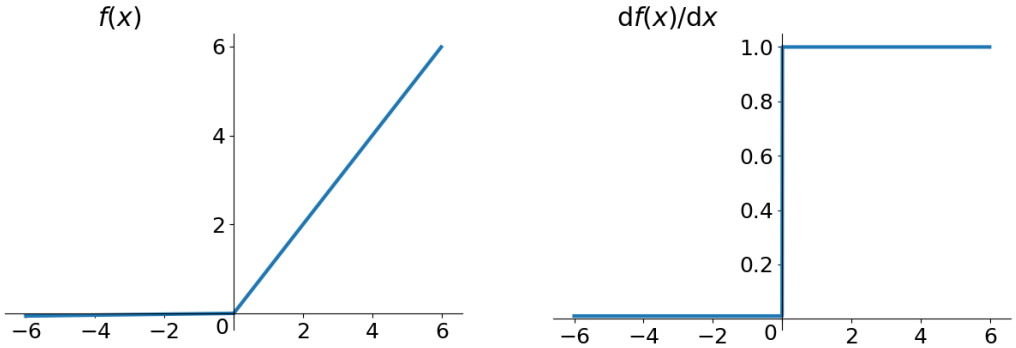
在其飽和區的接近於 0，都容易產生後續梯度消失、計算量大的問題。

- 套用到 model
  我將其套用到 linear neural network model 的 hidden layer 中，其 training loss 與 sigmoid 比較(其他參數固定下)，可以發現確實收斂的比 sigmoid 更快。



b. Leaky ReLU

$$leaky\ relu(x) = \text{np.where}(x > 0, x, \text{alpha} * x)$$

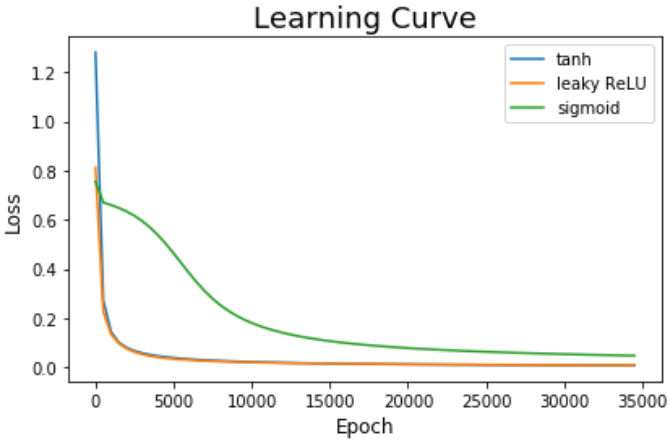


- Leaky ReLU 優點
  基本上繼承了 ReLU 的所有優點
  (1) 解決了 gradient vanishing 問題(在正區間)
  (2) 計算速度非常快，收斂速度遠快於 sigmoid 和 tanh
  (3) 解決 Dead ReLU Problem

- Leaky ReLU 缺點
  在實際操作當中，並沒有完全證明 Leaky ReLU 總是好於 ReLU。

- 套用到 model

  我將其套用到 linear neural network model 的 hidden layer 中，其 training loss 與 sigmoid 比較(其他參數固定下)，可以發現確實收斂的比 sigmoid 更快，但可能因為所求 function 較為簡單，因此跟 tanh 差不多。



5. Reference

- 也談激活函數 Sigmoid, Tanh, ReLu, softplus, softmax：https://zhuanlan.zhihu.com/p/48776056
- 聊一聊深度學習的 activation function：https://zhuanlan.zhihu.com/p/25110450