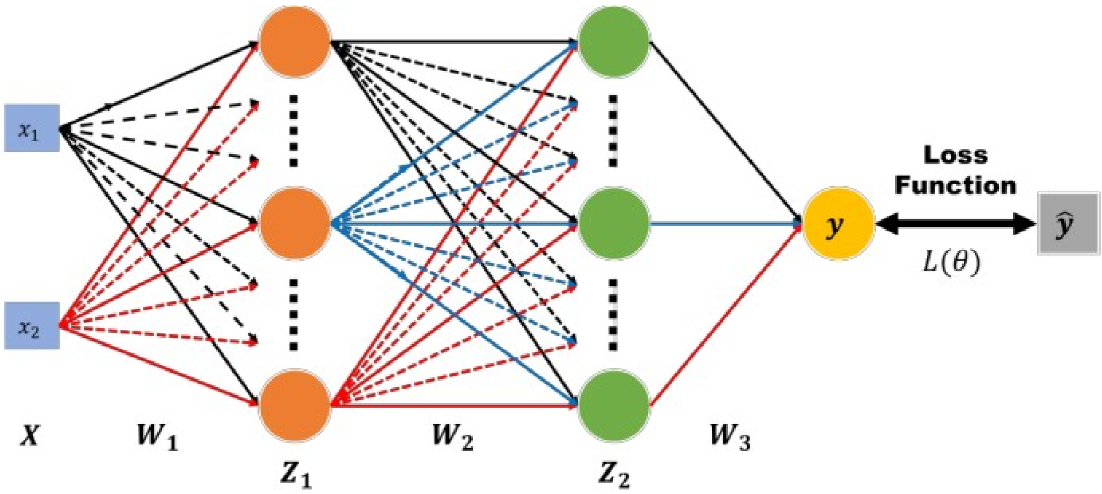
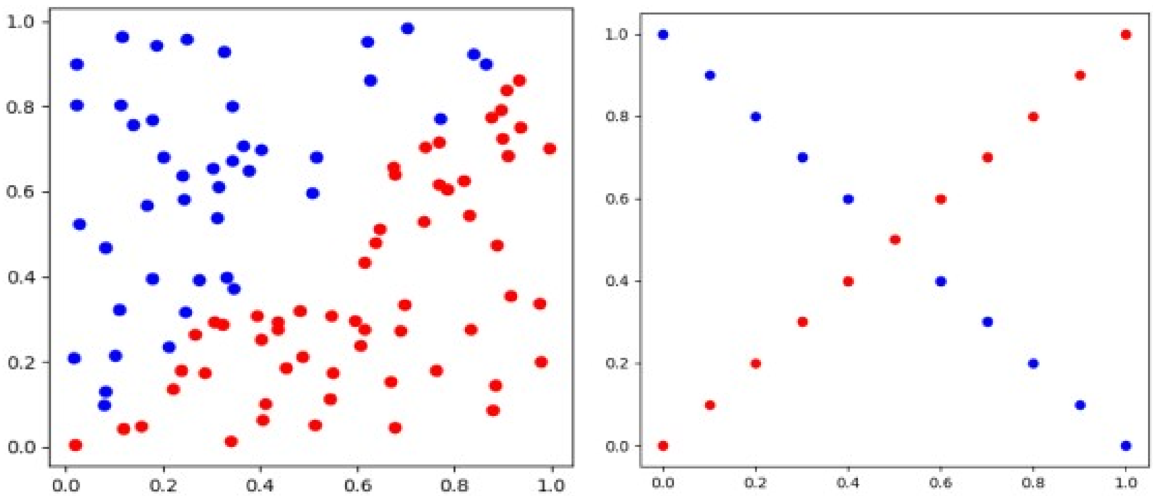
DLP-Lab 1 Backpropagation

1. Introduction:

本次 lab 使用的 network 為 fully connected neural network，具有兩個 input features、兩層 hidden layers 以及一個 output（如下圖），並使用 MSE 來計算 loss。



Input dataset 有兩種，一種為 linear，另一種為 XOR。



本次的目的是利用 python 中的 numpy 推導 backpropagation 並計算 gradient 來更新 weight，以達到 training 的效果。

2. Experiment setups:

A. Sigmoid functions

我們使用 sigmoid function 作為 activation function，公式及其導數的推導如下:

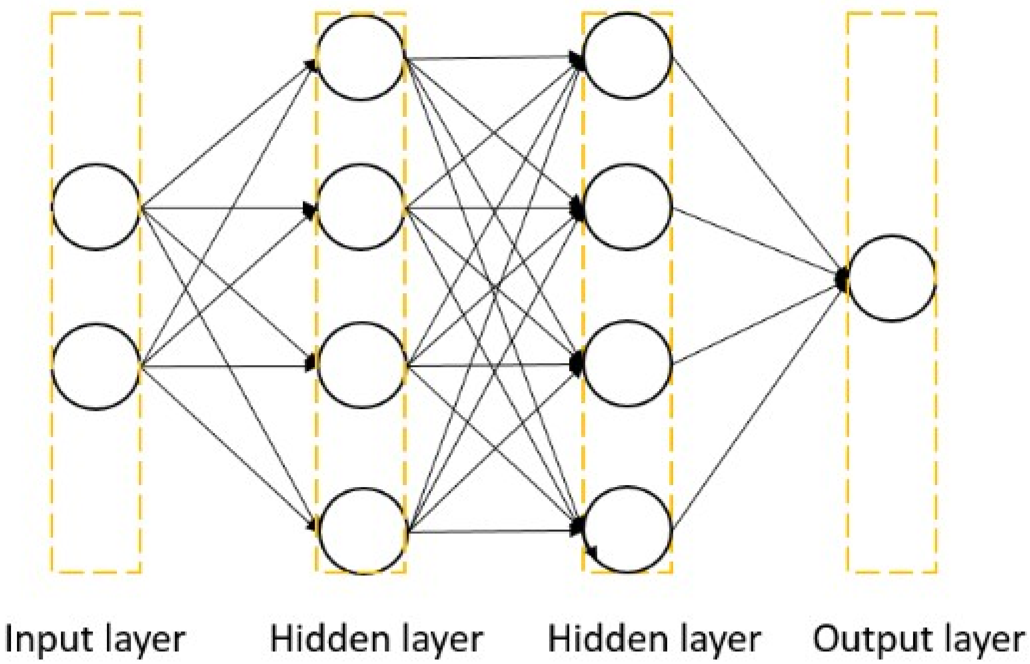
$$\sigma(x) = \frac{1}{1+e^{-x}}$$

$$
\begin{aligned}
\frac{d}{dx}\sigma(x) &= \frac{d}{dx}\left[\frac{1}{1+e^{-x}}\right] \\
&= \frac{d}{dx}\left(1+\mathrm{e}^{-x}\right)^{-1} \\
&= -(1+e^{-x})^{-2}(-e^{-x}) \\
&= \frac{e^{-x}}{\left(1+e^{-x}\right)^2} \\
&= \frac{1}{1+e^{-x}} \cdot \frac{e^{-x}}{1+e^{-x}} \\
&= \frac{1}{1+e^{-x}} \cdot \frac{(1+e^{-x})-1}{1+e^{-x}} \\
&= \frac{1}{1+e^{-x}} \cdot \left(\frac{1+e^{-x}}{1+e^{-x}} - \frac{1}{1+e^{-x}}\right) \\
&= \frac{1}{1+e^{-x}} \cdot \left(1 - \frac{1}{1+e^{-x}}\right) \\
&= \sigma(x) \cdot (1-\sigma(x))
\end{aligned}
$$

B. Neural network

Input neuron 數為 2 個，第一層 hidden layer 的 neuron 數為 4 個，第二層 hidden layer 的 neuron 數為 4 個，最後一層的 output neuron 數為 1 個。

```
# Define number of neurons for each layer
input_Neurons = 2
hid1_Neurons = 4
hid2_Neurons = 4
output_Neurons = 1
```



C. Backpropagation

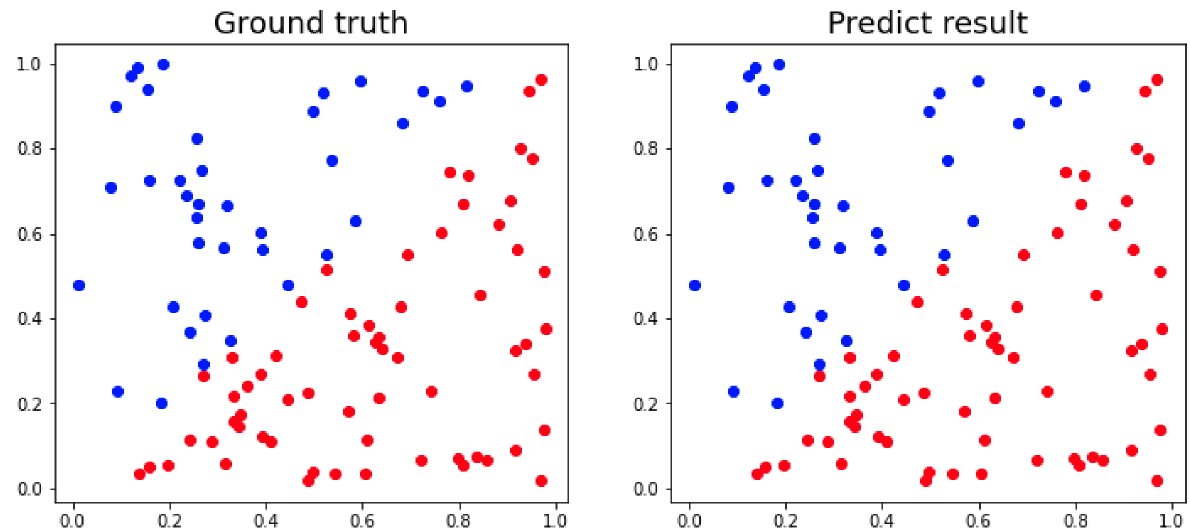
首先將 weights 都給予一隨機初始值，接著做 forward propagation 得到 predict result 並與 ground truth 計算 loss，然後利用 chain rule 將 loss 對每個 weight 做偏微分(back propagation)，得到每一層之間的 gradient，最後再乘上 learning rate 以 gradient descent 的方式更新 weights，目的是使 loss 愈小愈好。

```
# Back propagation
err_grad = t - y_pred #n*1
d_predict_output = err_grad * derivative_sigmoid(y_pred) #n*1
d_hid2_output = d_predict_output.dot(w3.T) * derivative_sigmoid(z2) #n*4
d_hid1_output = d_hid2_output.dot(w2) * derivative_sigmoid(z1) #n*4
```

3. Results of your testing:

A. Screenshot and comparison figure

- Linear dataset

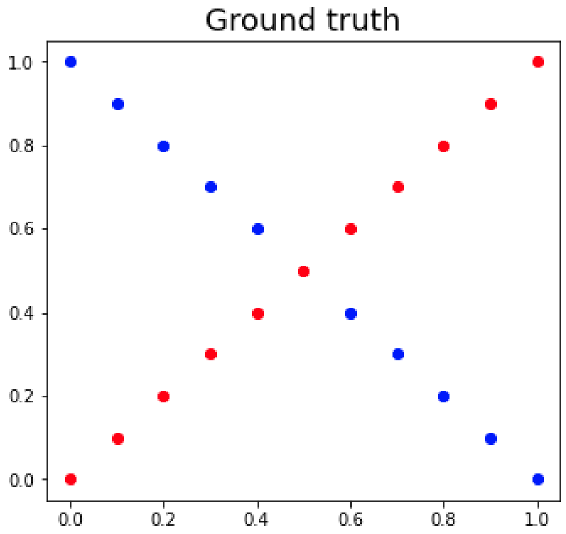


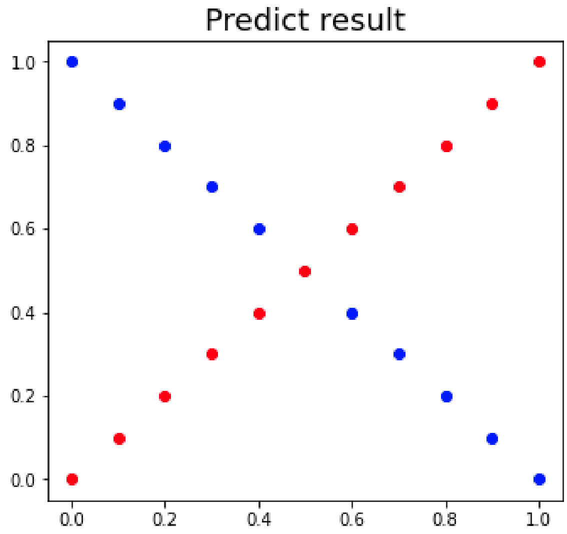

```
epoch     0 loss : 0.396965229707095
epoch  1000 loss : 0.242900579553403
epoch  2000 loss : 0.058870518783482
epoch  3000 loss : 0.021535113496805
epoch  4000 loss : 0.015062057886794
epoch  5000 loss : 0.012045850075054
epoch  6000 loss : 0.010119520113706
epoch  7000 loss : 0.008699688085672
epoch  8000 loss : 0.007576664811112
epoch  9000 loss : 0.006655585534311
epoch 10000 loss : 0.005885199393963
epoch 11000 loss : 0.005233643820714
epoch 12000 loss : 0.004678820395308
epoch 13000 loss : 0.004204095324813
epoch 14000 loss : 0.003796244053030
epoch 15000 loss : 0.003444427767209
epoch 16000 loss : 0.003139646985464
epoch 17000 loss : 0.002874406455117
epoch 18000 loss : 0.002642471234834
epoch 19000 loss : 0.002438665984230
epoch 20000 loss : 0.002258701420099
epoch 21000 loss : 0.002099023650834
epoch 22000 loss : 0.001956685126234
epoch 23000 loss : 0.001829235941526
epoch 24000 loss : 0.001714633616675
epoch 25000 loss : 0.001611169032859
```

```
epoch 26000 loss : 0.001517406084892
epoch 27000 loss : 0.001432132726396
epoch 28000 loss : 0.001354321335900
epoch 29000 loss : 0.001283096633319
epoch 30000 loss : 0.001217709675661
epoch 31000 loss : 0.001157516732152
epoch 32000 loss : 0.001101962071847
epoch 33000 loss : 0.001050563890197
epoch 34000 loss : 0.001002902758242
epoch 35000 loss : 0.000958612104192
epoch 36000 loss : 0.000917370337444
epoch 37000 loss : 0.000878894304489
epoch 38000 loss : 0.000842933828857
epoch 39000 loss : 0.000809267136795
epoch 40000 loss : 0.000777697009466
epoch 41000 loss : 0.000748047533454
epoch 42000 loss : 0.000720161345930
epoch 43000 loss : 0.000693897290376
epoch 44000 loss : 0.000669128414393
epoch 45000 loss : 0.000645740253604
epoch 46000 loss : 0.000623629355716
epoch 47000 loss : 0.000602702006915
epoch 48000 loss : 0.000582873129313
epoch 49000 loss : 0.000564065323514
```

```
Testing prediction:
 [[7.70004580e-06]
 [1.38472164e-06]
 [1.00985356e-07]
 [9.99995744e-01]
 [9.99995961e-01]
 [9.99995829e-01]
 [9.99995020e-01]
 [1.67994042e-07]
 [9.99996033e-01]
 [9.99995706e-01]
 [9.99995857e-01]
 [1.33910637e-07]
 [9.99508098e-01]
 [9.99995610e-01]
 [6.29007159e-02]
 [2.73932347e-06]
 [9.99989837e-01]
 [9.99994150e-01]
 [8.74119727e-07]
 [9.99993297e-01]
 [1.39792802e-07]
 [9.52088935e-01]
 [9.99991538e-01]
 [9.99536579e-01]
 [2.56665124e-07]
 [1.54130134e-07]
 [8.04494301e-06]
 [9.99996004e-01]
 [8.62393484e-08]
 [9.99995208e-01]
 [1.55660319e-07]
 [8.47436956e-08]
 [9.99996017e-01]
 [1.57223490e-07]
 [9.99986149e-01]
 [9.99994882e-01]
 [1.51708668e-07]
 [9.99995767e-01]
 [9.99989121e-01]
 [2.00069955e-07]
 [4.08422407e-07]
 [1.14552815e-07]
 [9.86952417e-01]
 [9.99993391e-01]
 [9.99995923e-01]
 [9.99991869e-01]
 [9.99993252e-01]
 [3.53423830e-03]
 [9.99434845e-01]
 [1.37159528e-06]
 [9.99972953e-01]
 [9.81953149e-01]
 [9.99980281e-01]
 [9.99993036e-01]
 [1.35464506e-05]
 [9.99991160e-01]
 [9.91078095e-08]
 [2.24365531e-07]
 [9.99988482e-01]
 [5.30161956e-07]
 [9.99996005e-01]
 [9.76614804e-08]
 [3.53732200e-07]
 [9.99958699e-01]
 [9.99995473e-01]
 [1.87897232e-06]
 [9.99992327e-01]
 [2.74233426e-02]
 [3.04141355e-07]
 [9.99970828e-01]
 [9.99961174e-01]
 [9.99995874e-01]
 [9.99995836e-01]
 [1.10128845e-07]
 [9.85657551e-08]
 [5.39941479e-07]
 [9.99996078e-01]
 [8.47536299e-08]
 [1.93688697e-07]
 [9.99956933e-01]
 [8.41118640e-01]
 [9.99995971e-01]
 [9.99995536e-01]
 [1.47871707e-01]
 [8.33099623e-08]
 [9.99995971e-01]
 [9.99995536e-01]
 [1.47871707e-01]
 [8.33099623e-08]
 [9.65309870e-08]
 [9.99994874e-01]
 [2.12998887e-07]
 [1.92102578e-05]
 [9.13600806e-08]
 [2.97873082e-07]
 [9.99987432e-01]
 [9.99995565e-01]
 [9.26409227e-08]
 [9.99995695e-01]
 [1.55716833e-07]
 [1.01689893e-07]
 [1.66843439e-07]
 [4.43896876e-07]
 [3.48003512e-05]]
```

- XOR dataset






```
epoch     0 loss : 0.407508640060338
epoch  1000 loss : 0.249439715843531
epoch  2000 loss : 0.249421073376786
epoch  3000 loss : 0.249400921404908
epoch  4000 loss : 0.249378098178887
epoch  5000 loss : 0.249350971913657
epoch  6000 loss : 0.249316669078693
epoch  7000 loss : 0.249269174942631
epoch  8000 loss : 0.249193678868640
epoch  9000 loss : 0.249045205612083
epoch 10000 loss : 0.248635212401745
epoch 11000 loss : 0.246663713767889
epoch 12000 loss : 0.233496974542446
epoch 13000 loss : 0.161588103171700
epoch 14000 loss : 0.069364535446235
epoch 15000 loss : 0.042813329348166
epoch 16000 loss : 0.028582177814472
epoch 17000 loss : 0.018005056502422
epoch 18000 loss : 0.010890002173648
epoch 19000 loss : 0.006778579883588
epoch 20000 loss : 0.004493839989203
epoch 21000 loss : 0.003176106488765
epoch 22000 loss : 0.002368876150185
epoch 23000 loss : 0.001844098483960
epoch 24000 loss : 0.001484850245630
epoch 25000 loss : 0.001228053152481
```

```
epoch 26000 loss : 0.001037787481826
epoch 27000 loss : 0.000892546746993
epoch 28000 loss : 0.000778868486712
epoch 29000 loss : 0.000687991848708
epoch 30000 loss : 0.000614019688977
epoch 31000 loss : 0.000552863882604
epoch 32000 loss : 0.000501616423776
epoch 33000 loss : 0.000458161357636
epoch 34000 loss : 0.000420927893298
epoch 35000 loss : 0.000388728990270
epoch 36000 loss : 0.000360653214330
epoch 37000 loss : 0.000335990671732
epoch 38000 loss : 0.000314181265810
epoch 39000 loss : 0.000294777892106
epoch 40000 loss : 0.000277419827349
epoch 41000 loss : 0.000261813199409
epoch 42000 loss : 0.000247716456652
epoch 43000 loss : 0.000234929420225
epoch 44000 loss : 0.000223284939729
epoch 45000 loss : 0.000212642464761
epoch 46000 loss : 0.000202883043073
epoch 47000 loss : 0.000193905392719
epoch 48000 loss : 0.000185622791016
epoch 49000 loss : 0.000177960590628
```

```
Testing prediction:
 [[9.41727530e-04]
 [9.99944767e-01]
 [3.51158059e-03]
 [9.99948145e-01]
 [1.09665522e-02]
 [9.99945050e-01]
 [2.19123640e-02]
 [9.99886612e-01]
 [2.55207219e-02]
 [9.70166226e-01]
 [1.85346477e-02]
 [9.76375336e-03]
 [9.68869183e-01]
 [4.44009815e-03]
 [9.99864100e-01]
 [2.01260273e-03]
 [9.99948230e-01]
 [9.91201675e-04]
 [9.99959502e-01]
 [5.48682851e-04]
 [9.99963286e-01]]
```

B. Show the accuracy of your prediction

- Linear dataset

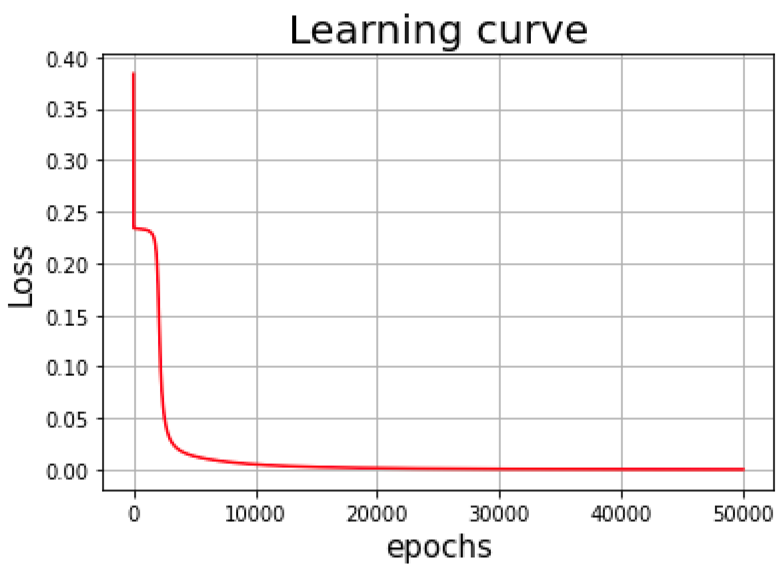
```
Accuracy : 0.98
```

- XOR dataset

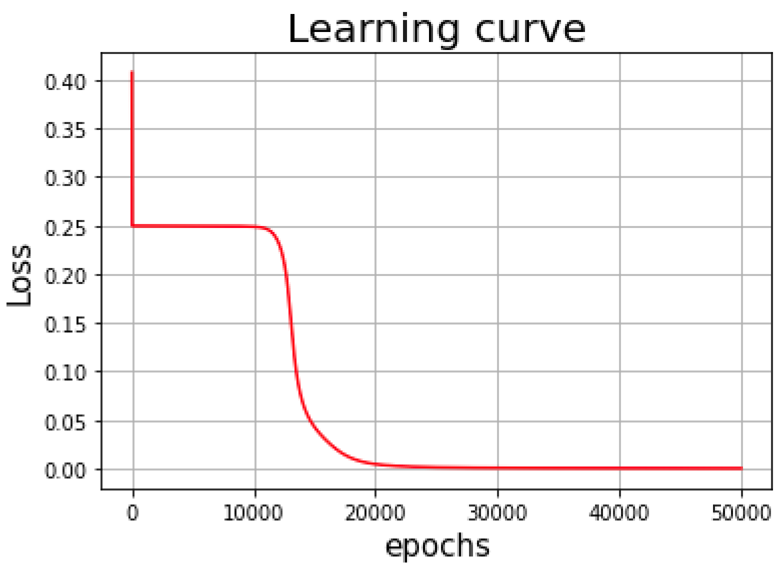
```
Accuracy : 0.9047619047619048
```

C. Learning curve (loss, epoch curve)

- Linear dataset



- XOR dataset



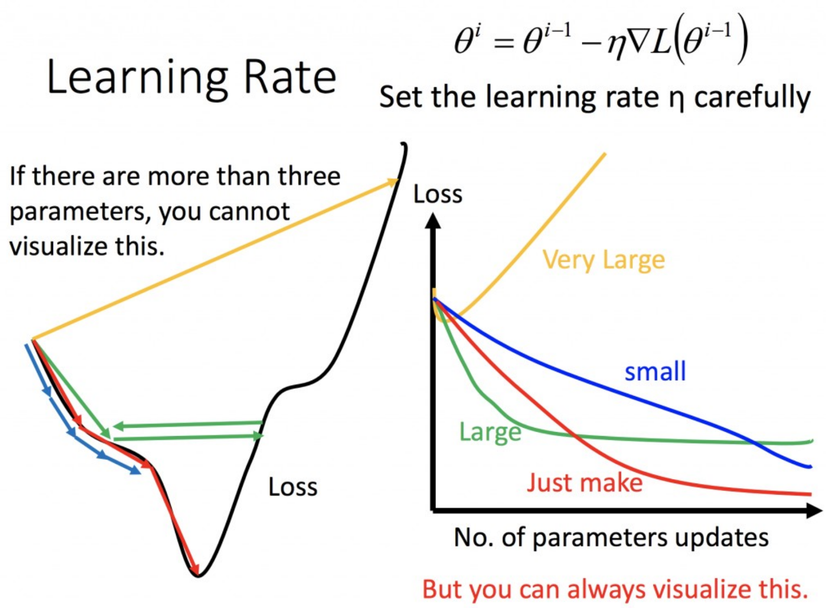
D. Anything you want to present

起初計算 back propagation 時，認為只要內積後 dimension 算起來沒有問題，就能使 loss 下降並得到不錯的 prediction，但結果卻是 loss 有下降，但 predict output 錯的離譜。後來檢查發現，原來內積後 dimension 的結果一樣，但內積時將 layer output 做 transpose 再與 gradient output 做內積，出來的 prediction 結果才是對的。

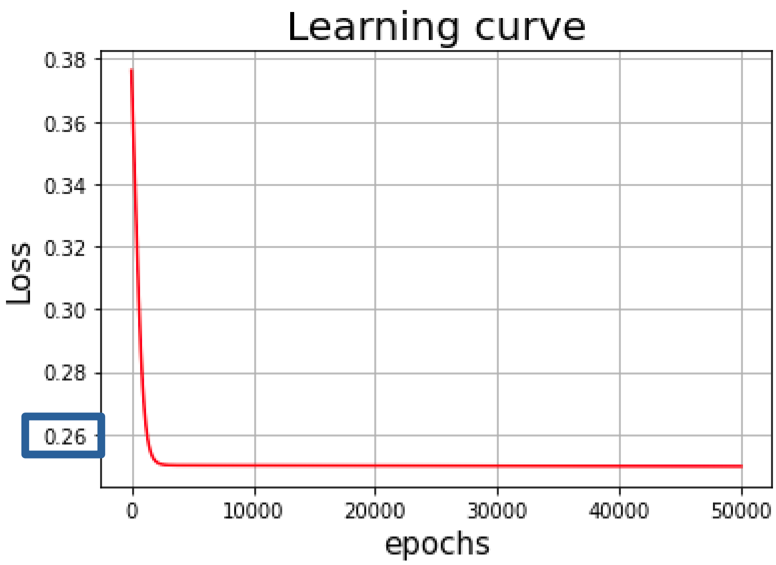
4. Discussion

A. Try different learning rates

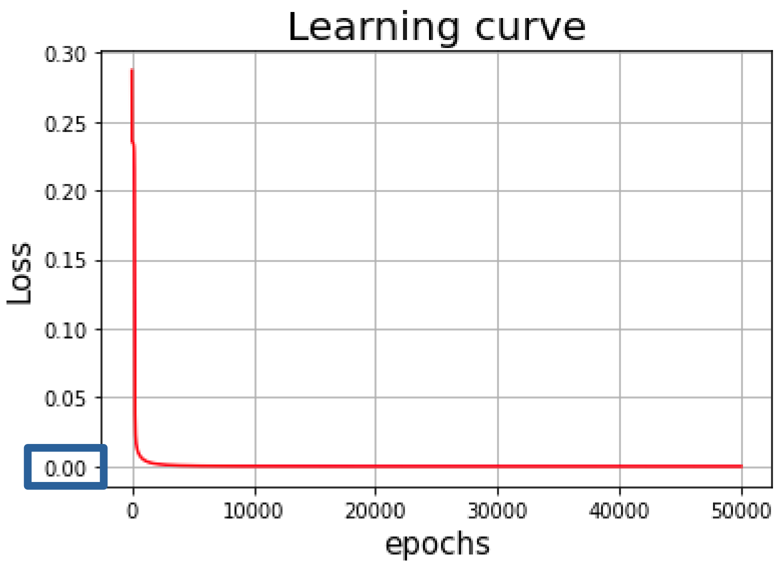
以 linear dataset 為例，當 learning rate 太低時，會造成 loss 下降太慢，無法到達最佳解，除非將 epoch 增加。而當 learning rate 太高時，會使 loss 下降到某個程度便無法再繼續下降，最終沒辦法收斂。因此 learning rate 是個決定能不能 train 成功很重要的參數之一。



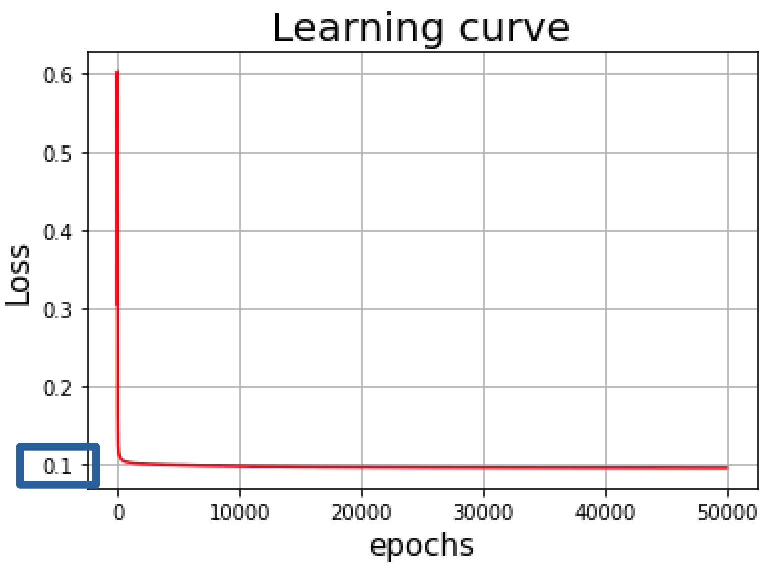
- learning rate = 0.0001



- learning rate = 0.1



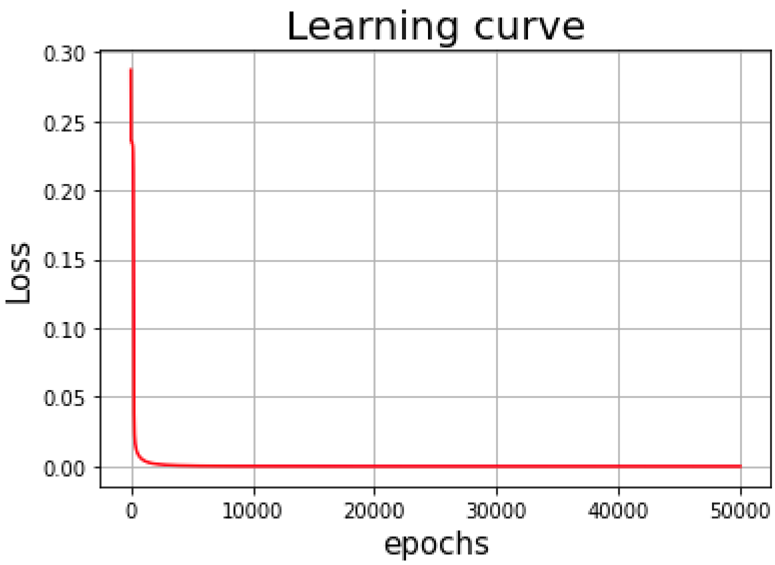
- learning rate = 1



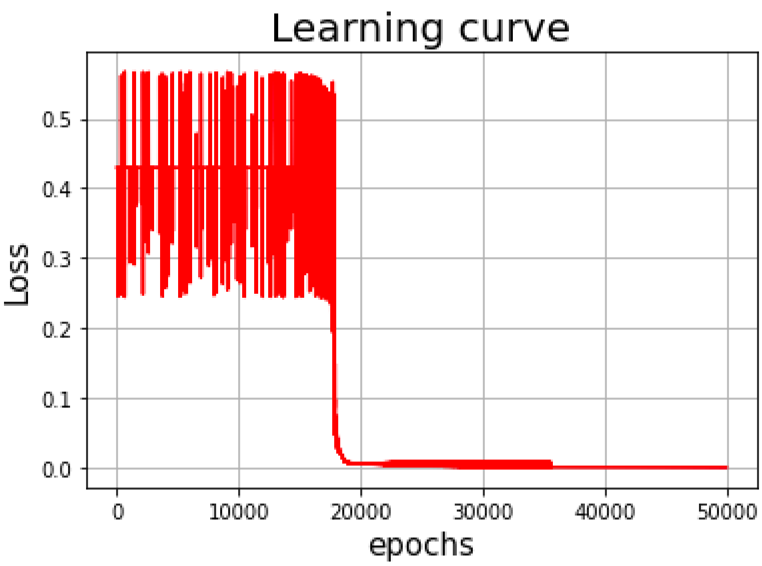
B. Try different numbers of hidden units

當 hidden units 變多時，會因為 neuron 數太多，造成計算速度下降，training 時間變長，前半部 epochs 的 loss 振幅較大(linear dataset)，原因是因為 units 增加，沒辦法在前面的 epochs 就將所有 units 訓練好，所以遇到新 data 時 loss 就會上升。對於 accuracy 的話則是沒有太大的影響，除非 units 真的太多，會造成整個 network 訓練時無法收斂，prediction 也跟著錯誤。

- Units = 4



- Units = 20



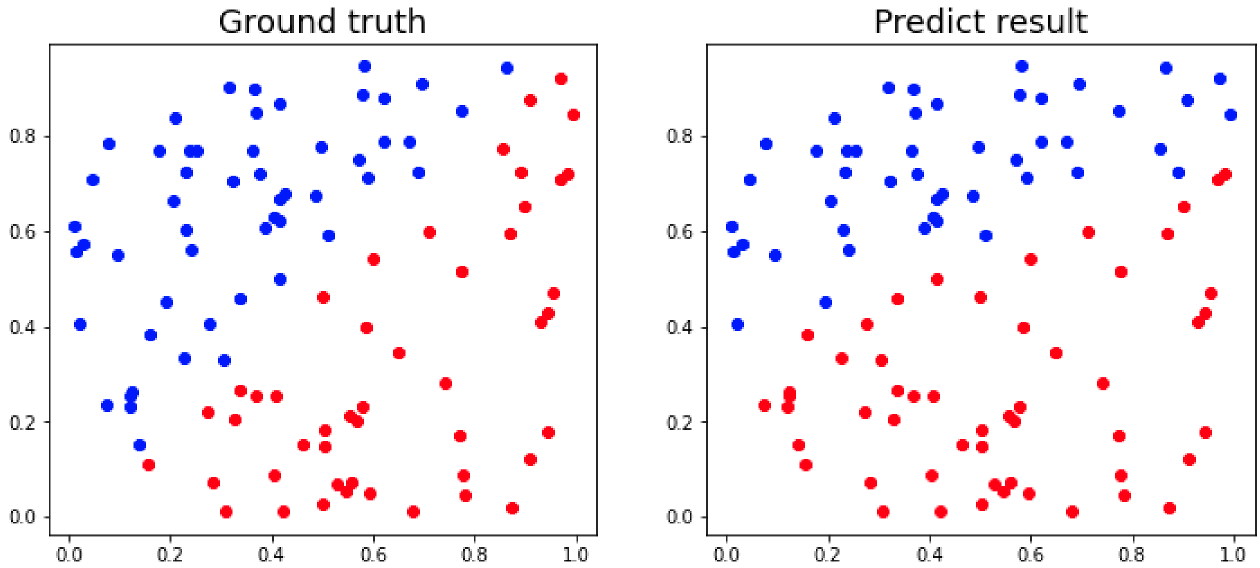
C. Try without activation functions

這邊設置兩種 dataset 的 training epochs 為 50000，learning rate 為 0.0001。當沒有 activation function 做非線性轉換時，可以發現 linear dataset 預測出來的結果有 6 成以上，而 XOR dataset 預測出來的只剩 5 成。原因是 linear dataset 本身就呈現線性分佈，能找出一條線將其分開，但 XOR dataset 的非線性分佈，若只用一條線分開，

則會造成準確度降低。

- linear dataset

```
Accuracy : 0.67
```



- XOR dataset

```
Accuracy : 0.5238095238095238
```