

人工智能之深度学习


第一章 深度学习入门-BP-讲义

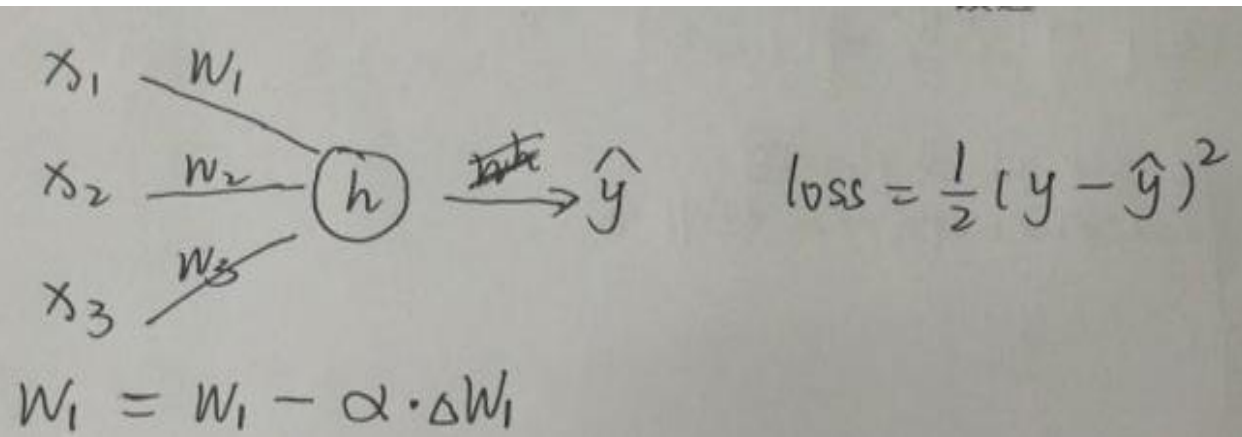
主讲人：Vincent Ying

上海育创网络科技有限公司



课程目录

- 一 深度学习的应用场景
- 二 神经网络的起源
- 三 神经网络的基本结构
- 四 反向传播(BP)神经网络 



手动画图演示这个梯度下降的数学推导

改进

Diagram illustrating a linear model with three inputs x_1, x_2, x_3 and weights w_1, w_2, w_3 . The inputs are multiplied by their respective weights and summed in a circle labeled h . The output is \hat{y} . The loss function is given as $\text{loss} = \frac{1}{2} (y - \hat{y})^2$.

Weight update rule: $w_1 = w_1 - \alpha \cdot \Delta w_1$

(1) 正向传播

$h = w_1 \cdot x_1 + w_2 \cdot x_2 + w_3 \cdot x_3$

$\hat{y} = h$

(2) 反向传播

$$\frac{\partial \text{Loss}}{\partial w_1} = \frac{\partial \text{Loss}}{\partial \hat{y}} \cdot \frac{\partial \hat{y}}{\partial h} \cdot \frac{\partial h}{\partial w_1}$$

$$= -(y - \hat{y}) \times 1 \cdot \frac{\partial (w_1 x_1 + w_2 x_2 + w_3 x_3)}{\partial w_1}$$

$$= -(y - \hat{y}) \times 1 \times x_1$$

$$= (\hat{y} - y) \cdot x_1$$

(3):

$$\begin{bmatrix} w_1 - \alpha \cdot \Delta w_1 \\ w_2 - \alpha \cdot \Delta w_2 \\ w_3 - \alpha \cdot \Delta w_3 \end{bmatrix} = \begin{bmatrix} w_1 - \alpha (\hat{y} - y) x_1 \\ w_2 - \alpha (\hat{y} - y) x_2 \\ w_3 - \alpha (\hat{y} - y) x_3 \end{bmatrix}$$

反向传播练习

激活函数均使用sigmoid; sigmoid的导数是?

$\text{sigmoid}(-0.02)=0.495.$

$\text{sigmoid}(0.0495)=0.512$

Y值为: 1

学习率 η 为: 0.5

问题: 求输入层--隐藏层 和 隐藏层--输出层 权重更新步长!

1、sigmoid 函数的导数 $f'(W * a) = f(W * a)(1-f(W * a))$

2、输出节点的误差项 (error term) 可表示为

$\delta_o = (y - \hat{y}) f'(W * a) =$

3、隐藏节点的误差项

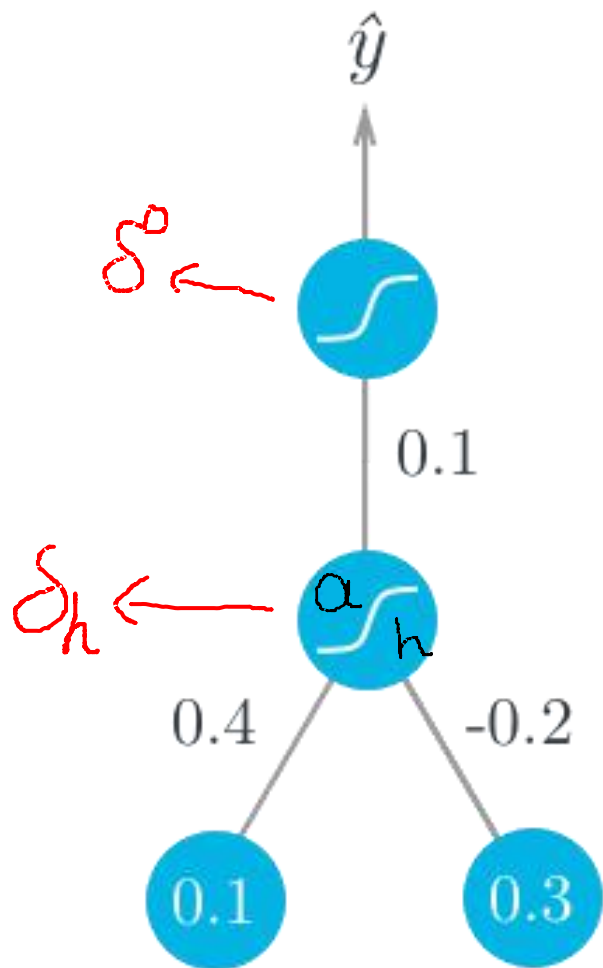
$\delta_h = W * \delta_o * f'(h) =$

4、隐藏层-输出层权重更新步长=学习率*输出节点误差*隐藏节点激活值

$\Delta W = \eta * \delta_o * a =$

5、输入-隐藏层权重 $w_i =$ 学习率*隐藏节点误差*输入值

$\Delta W_i = \eta * \delta_h * X_i =$





sigmoid - 阶导数

$$\left(\frac{1}{1+e^{-x}} \right)' = - \frac{1}{(1+e^{-x})^2} (1+e^{-x})'$$

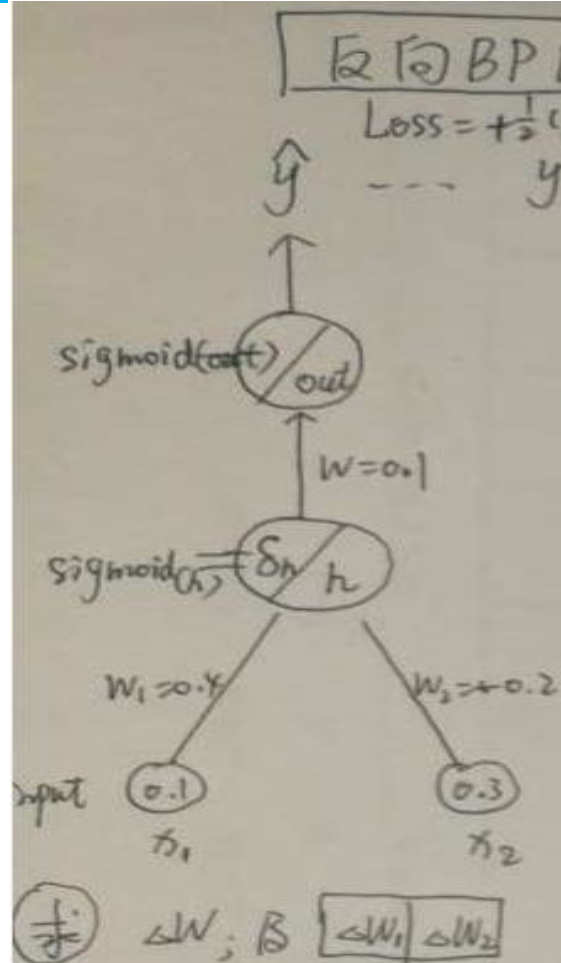
$$= - \frac{1}{(1+e^{-x})^2} e^{-x} (-x)'$$

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

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

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

$$= \frac{1}{(1+e^{-x})} \times \left(1 - \frac{1}{1+e^{-x}} \right)$$



<1> 正向传播.

$$h = \begin{bmatrix} x_1 & x_2 \end{bmatrix} \times \begin{bmatrix} W_1 \\ W_2 \end{bmatrix} = -0.02$$

$$\delta_h = \text{sigmoid}(h) = 0.495$$

$$out = \delta_h \times W = 0.0495$$

$$\hat{y} = \text{sigmoid}(out) = 0.512$$

<2> 反向BP

$$\Delta W = \frac{\partial \text{Loss}}{\partial W} = \frac{\partial \text{Loss}}{\partial \hat{y}} \cdot \frac{\partial \hat{y}}{\partial out} \cdot \frac{\partial out}{\partial W} = \underbrace{-(y - \hat{y}) \cdot \hat{y} \cdot (1 - \hat{y})}_{\text{output-error term}} \cdot \delta_h$$

$$= (0.512 - 1) \cdot (0.512) \cdot (1 - 0.512) \cdot 0.495$$

$$\begin{bmatrix} \Delta W_1 & \Delta W_2 \end{bmatrix} = \begin{bmatrix} \frac{\partial \text{Loss}}{\partial W_1} & \frac{\partial \text{Loss}}{\partial W_2} \end{bmatrix} = \begin{bmatrix} \underbrace{\frac{\partial \text{Loss}}{\partial \hat{y}} \cdot \frac{\partial \hat{y}}{\partial out} \cdot \frac{\partial out}{\partial \delta_h}}_{\text{output-error-term}} \cdot \underbrace{\frac{\partial \delta_h}{\partial h}}_{\text{hidden-error-term}} \cdot \frac{\partial h}{\partial W_1} & \frac{\partial h}{\partial W_2} \end{bmatrix}$$

$$= \begin{bmatrix} \underbrace{\text{output-error-term}}_{-0.122} \times \underbrace{W}_{0.1} \times \underbrace{(\delta_h) \times (1 - \delta_h)}_{0.495 \times (1 - 0.495)} \times \underbrace{x_1}_{0.1} & \text{hidden-error-term} \times x_2 \end{bmatrix}$$

$$= \text{hidden-error-term} \cdot \begin{bmatrix} x_1 & x_2 \end{bmatrix} = -0.003 \cdot \begin{bmatrix} 0.1 & 0.3 \end{bmatrix}$$

$\times \text{学习率} \alpha = 0.5$



$$\begin{bmatrix} \Delta W_1 & \Delta W_2 \end{bmatrix}$$

反向传播-答案

Step 1 正向传播

$$h = \sum W_i * X_i = 0.1 \times 0.4 - 0.2 \times 0.3 = -0.02$$

隐藏层节点的输出 $a = f(h) = \text{sigmoid}(-0.02) = 0.495$.

该神经网络的output为 $\hat{Y} = f(W \cdot a) = \text{sigmoid}(0.1 \times 0.495) = 0.512$

Step 2 反向传播

1、sigmoid 函数的导数 $f'(W * a) = f(W * a)(1 - f(W * a))$

2、输出节点的误差项 (error term) 可表示为

$$\delta_o = (y - \hat{Y}) f'(W * a) = (1 - 0.512) \times 0.512 \times (1 - 0.512) = 0.122$$

3、隐藏节点的误差项

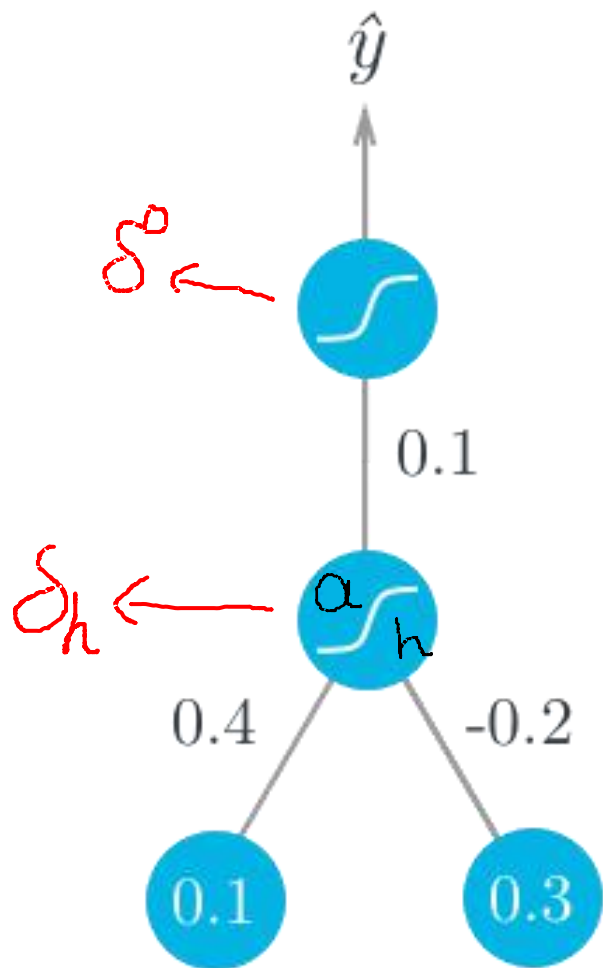
$$\delta_h = W * \delta_o * f'(h) = 0.1 \times 0.122 \times 0.495 \times (1 - 0.495) = 0.003$$

4、隐藏层-输出层权重更新步长=学习率*输出节点误差*隐藏节点激活值

$$\Delta W = \eta * \delta_o * a = 0.5 \times 0.122 \times 0.495 = 0.0302$$

5、输入-隐藏层权重 $w_{_i}$ = 学习率*隐藏节点误差*输入值

$$\Delta W_i = \eta * \delta_h * X_i = (0.5 \times 0.003 \times 0.1, 0.5 \times 0.003 \times 0.3) = (0.00015, 0.00045)$$



02Code反向传播练习

GRE反向-伪代码

GRE 反向-伪代码

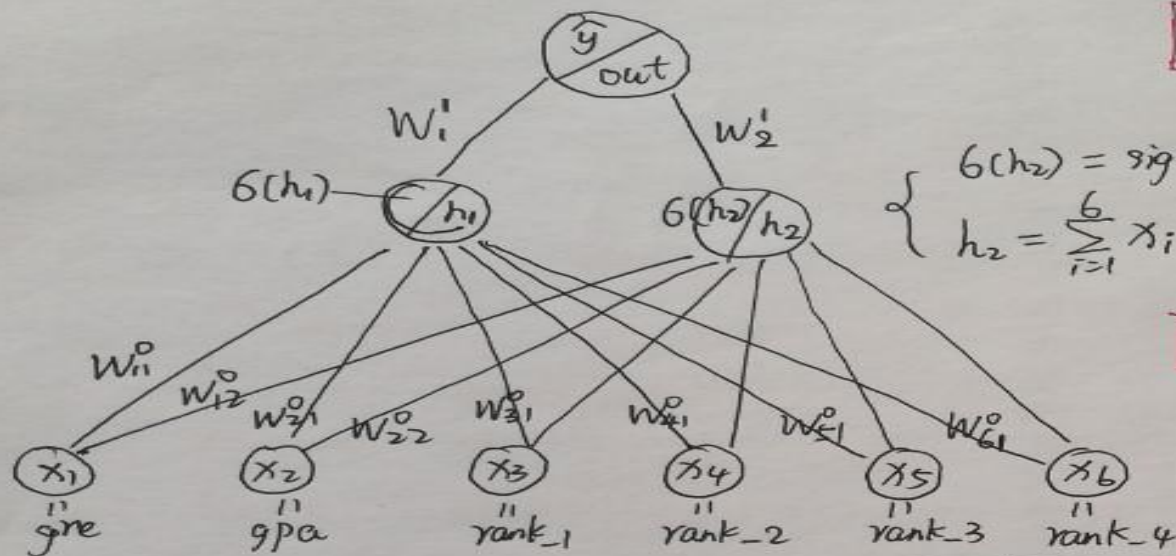
```

n_records, n_features = features.shape
last_loss = None
初始化权重:  $W_{input2h}$ ,  $W_{h2output}$ 
for e in range(epochs):
     $\Delta W_{input2h}$ ,  $\Delta W_{h2output}$  = np.zeros( $W_{input2h}$ ), np.zeros( $W_{h2output}$ )
    for X, y in zip(features.values, targets):
        ① 正向传播
        ② 获得 error
        ③ 反向传播
         $\Delta W_{input2h} +=$ 
         $\Delta W_{h2output} +=$ 
     $W_{input2h} -= \Delta W_{input2h} \cdot learning\_rate / n\_records$ 
     $W_{h2output} -= learning\_rate \times \Delta W_{h2output} / n\_records$ 
    每 10 个 epoch:
        正向传播, 求出当前 loss, 若  $loss > last\_loss$ , 则 break
    用 test 数据, 做正向传播, 求 acc
    
```


GRE反向传播-推导1

①

$$MSE: Loss = \frac{1}{2m} (y - \hat{y})^2$$



$$\begin{bmatrix} \sigma(h_1) & \sigma(h_2) \end{bmatrix} \cdot \begin{bmatrix} W_1' \\ W_2' \end{bmatrix} = out$$

$$\begin{cases} \sigma(h_2) = \text{sigmoid}(h_2) \\ h_2 = \sum_{i=1}^6 x_i W_{i2}^0 \end{cases}$$

$$\begin{bmatrix} x_1 & x_2 & x_3 & x_4 & x_5 & x_6 \end{bmatrix} \cdot \begin{bmatrix} W_{11}^0 & W_{12}^0 \\ W_{21}^0 & W_{22}^0 \\ W_{31}^0 & W_{32}^0 \\ W_{41}^0 & W_{42}^0 \\ W_{51}^0 & W_{52}^0 \\ W_{61}^0 & W_{62}^0 \end{bmatrix} = \begin{bmatrix} h_1 & h_2 \end{bmatrix}$$

样本1						
样本2						

一、正向传播

$$\begin{bmatrix} h_1 & h_2 \end{bmatrix} = \sum_{i=1}^m x_i W_i^0 = x_i \cdot W^0$$

$$hidden_output = \begin{bmatrix} \sigma(h_1) & \sigma(h_2) \end{bmatrix} = \begin{bmatrix} \text{sigmoid}(h_1) & \text{sigmoid}(h_2) \end{bmatrix}$$

$$out = \sigma(h_1) \cdot W_1' + \sigma(h_2) \cdot W_2'$$

$$\hat{y} = \text{sigmoid}(out)$$

GRE反向传播--推导

二、反向传播

① input-to-hidden

$$W_{11}^o = \frac{\partial \text{Loss}}{\partial w_{11}^o} = \frac{\partial \text{Loss}}{\partial \hat{y}} \cdot \frac{\partial \hat{y}}{\partial \text{out}} \cdot \frac{\partial \text{out}}{\partial \phi(h_1)} \cdot \frac{\partial \phi(h_1)}{\partial h_1} \cdot \frac{\partial h_1}{\partial w_{11}^o}$$

$$= \underbrace{-(y - \hat{y})}_{\text{error}} \cdot \underbrace{\text{sigmoid}(\text{out})[1 - \text{sigmoid}(\text{out})]}_{\text{output_error_term}} \cdot W_1' \cdot \underbrace{\text{sigmoid}(h_1)[1 - \text{sigmoid}(h_1)]}_{\text{hidden_error_term}} \cdot X_1$$

$$\text{hidden_error} = \text{output_error_term} \cdot [W_1', W_2'] \quad \boxed{\quad \quad}$$

$$\text{hidden_error_term} = \text{hidden_error} \cdot [\text{sigmoid}(h_1)[1 - \text{sigmoid}(h_1)], \text{sigmoid}(h_2)[1 - \text{sigmoid}(h_2)]] \quad \boxed{\quad \quad}$$

shape
 $\boxed{\quad \quad}$

$$\Delta W^o = \begin{bmatrix} \Delta W_{11}^o & \Delta W_{12}^o \\ \Delta W_{21}^o & \Delta W_{22}^o \\ \Delta W_{31}^o & \Delta W_{32}^o \\ \Delta W_{41}^o & \Delta W_{42}^o \\ \Delta W_{51}^o & \Delta W_{52}^o \\ \Delta W_{61}^o & \Delta W_{62}^o \end{bmatrix} = \underbrace{\boxed{\quad \quad}}_{\text{hidden_error_term}} \cdot \begin{bmatrix} X_1 \\ X_2 \\ X_3 \\ X_4 \\ X_5 \\ X_6 \end{bmatrix}$$

GRE 反向 BP

$$\Delta W^0 = \begin{bmatrix} \Delta W_{11}^0 & \Delta W_{12}^0 \\ \Delta W_{21}^0 & \Delta W_{22}^0 \\ \Delta W_{31}^0 & \Delta W_{32}^0 \\ \Delta W_{41}^0 & \Delta W_{42}^0 \\ \Delta W_{51}^0 & \Delta W_{52}^0 \\ \Delta W_{61}^0 & \Delta W_{62}^0 \end{bmatrix} = \begin{bmatrix} x_1 \cdot h_error_term_1 & x_1 \cdot h_error_term_2 \\ x_2 \cdot h_error_term_1 & x_2 \cdot h_error_term_2 \\ x_3 & x_3 \\ x_4 & x_4 \\ x_5 & x_5 \\ x_6 & x_6 \end{bmatrix} \dots$$

$$= \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \\ x_5 \\ x_6 \end{bmatrix} \cdot \begin{bmatrix} h_error_term_1 & h_error_term_2 \end{bmatrix}$$

$$\text{batch_size} \times \text{Weights1}(3, 4) = (\text{batch_size}, 4)$$

$$\begin{bmatrix} b(h_1), b(h_2), b(h_3), b(h_4) \\ b(h_1), b(h_2), b(h_3), b(h_4) \end{bmatrix}$$

$$b(\text{batch_size}, 4)$$

$$b(\text{batch_size}, 4) \times \text{Weights2}(4, 1) = (\text{batch_size}, 1) = \hat{y}_i$$

$$\text{Loss} = \frac{1}{2m} (\hat{y}_i - y_i)^2$$

$$\frac{\partial \text{Loss}}{\partial w_i} = \frac{1}{m} (\hat{y}_i - y_i) \quad \text{Batches FP}$$

$$\frac{\partial \text{Loss}}{\partial w_i} = \frac{\partial \frac{1}{2m} (\hat{y}_i - y_i)^2}{\partial \hat{y}_i} \cdot 1 \cdot \frac{\partial \text{out}}{\partial w_i}$$

$$\frac{1}{m} (\hat{y}_i - y_i) \cdot 1 \cdot A^T$$

$$\frac{1}{2} \begin{bmatrix} \hat{y}_1 - y_1 \\ \hat{y}_2 - y_2 \end{bmatrix} \begin{bmatrix} & & & \\ & & & \\ & & & \\ & & & \end{bmatrix} = \text{hidden_output}^T$$

$$\frac{\partial \text{Loss}}{\partial \text{weight}_2} = \begin{bmatrix} b(h_1) & b(h_2) \\ b(h_2) & b(h_2) \\ b(h_4) & b(h_3) \\ b(h_4) & b(h_4) \end{bmatrix} \cdot \begin{bmatrix} \hat{y}_1 - y_1 = \text{err}_1 \\ \hat{y}_2 - y_2 = \text{err}_2 \end{bmatrix}$$

$$\begin{bmatrix} \text{err}_1 \cdot b(h_1) + \text{err}_2 \cdot b(h_2) \\ \text{err}_1 \cdot b(h_2) + \text{err}_2 \cdot b(h_3) \\ \text{err}_1 \cdot b(h_4) + \text{err}_2 \cdot b(h_4) \end{bmatrix}$$

// k=batch_size
hidden_output^T

$$\frac{\partial \text{Loss}}{\partial \text{Weights1}} = \frac{1}{\text{batch}} (\hat{y}_i - y_i) \cdot \text{weight}_2 \cdot \begin{bmatrix} b(h_1)(1-b(h_1)) & b(h_2)(1-b(h_2)) & b(h_3)(1-b(h_3)) & b(h_4)(1-b(h_4)) \\ b(h_1)(1-b(h_1)) & b(h_2)(1-b(h_2)) & b(h_3)(1-b(h_3)) & b(h_4)(1-b(h_4)) \end{bmatrix}$$

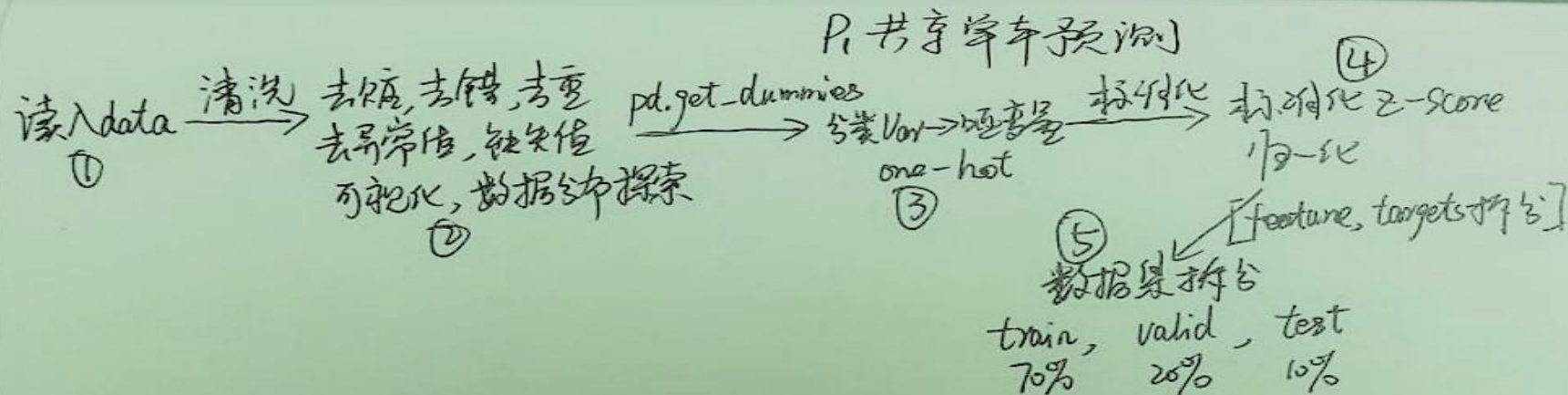
$$= \frac{1}{\text{batch}} \begin{bmatrix} \text{err}_1 \\ \text{err}_2 \end{bmatrix} \cdot \begin{bmatrix} w_1^2 & w_2^2 & w_3^2 & w_4^2 \end{bmatrix}$$

$$\begin{bmatrix} \text{err}_1 \cdot w_1^2 & \text{err}_1 \cdot w_2^2 & \text{err}_1 \cdot w_3^2 & \text{err}_1 \cdot w_4^2 \\ \text{err}_2 \cdot w_1^2 & \text{err}_2 \cdot w_2^2 & \text{err}_2 \cdot w_3^2 & \text{err}_2 \cdot w_4^2 \end{bmatrix}$$

$$X_i \text{ shape} = (\text{batch_size}, 3)$$

$$X_i^T \cdot \begin{bmatrix} & & & \\ & & & \\ & & & \end{bmatrix} = (3, 4)$$

// batch_size



① Model 结构 (hidden Nodes, layer 层数)
 $W_{i \rightarrow h}$: shape, $W_{h \rightarrow o}$: shape, initialize values.

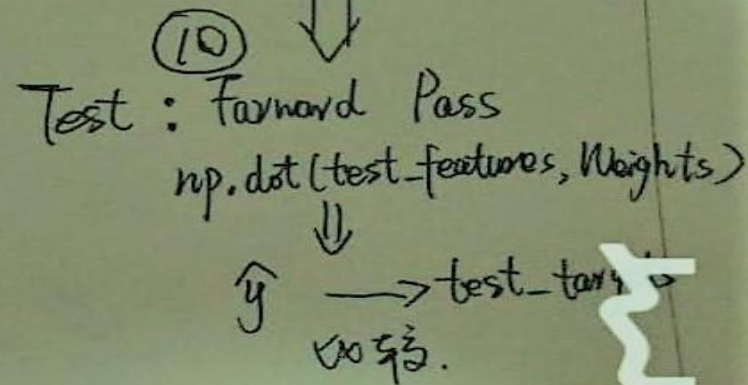
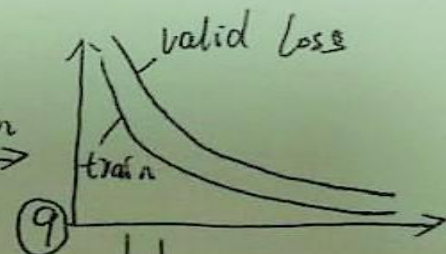
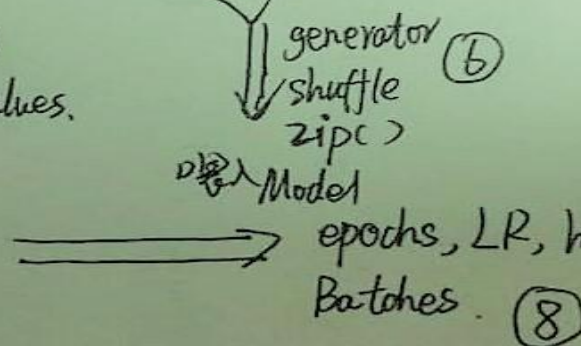
② activation func, prime
Loss func. $MSE = \frac{1}{2m} (\hat{y} - y)^2$

③ Forward Pass
 $np.matmul() + activation$

④ Backward Pass → 获取 gradient
Update $W_{i \rightarrow h}, W_{h \rightarrow o}$ (LR, n-records)

⑤ Valid Loss 获取, 记录

⑥ Save. train loss / valid loss
作图



单车反向传播-伪代码

共享单车伪代码

```

{
  n_record, n_feature = x_train.shape()
  初始化 W
  W_input2h, W_h2output

  for e in range(epochs):
    ΔW_input2h, ΔW_h2output = np.zeros(W_input2h), np.zeros(W_h2output)

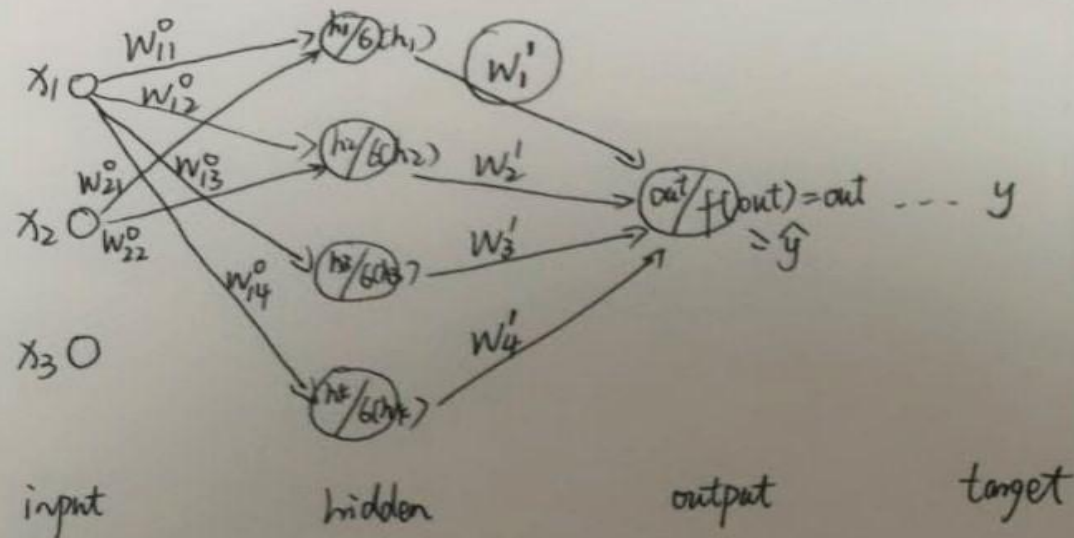
    for X, y in Zip(x_train, targets):
      ① 正向传播
      ② 获得 error
      ③ 反向传播
      ΔW_input2h += , ΔW_h2output +=

      W_input2h -= learning_rate * ΔW_input2h / n_record
      W_h2output -= learning_rate * ΔW_h2output / n_record
    if e // 10 == 0:
      计算损失, 并打印出来。

  使用 test-data, 进行正向传播, 获得 pred, 测试模型效果!

```

反向传播



正向 Pass

$$\begin{bmatrix} x_1 & x_2 & x_3 \end{bmatrix} \begin{bmatrix} W_{11}^0 & W_{12}^0 & W_{13}^0 & W_{14}^0 \\ W_{21}^0 & W_{22}^0 & W_{23}^0 & W_{24}^0 \\ W_{31}^0 & W_{32}^0 & W_{33}^0 & W_{34}^0 \end{bmatrix} = [h_1, h_2, h_3, h_4] \quad \text{hidden}$$

↓ sigmoid

$$[b(h_1), b(h_2), b(h_3), b(h_4)]$$

$$\begin{bmatrix} b(h_1), b(h_2), b(h_3), b(h_4) \end{bmatrix} \cdot \begin{bmatrix} W_1^1 \\ W_2^1 \\ W_3^1 \\ W_4^1 \end{bmatrix} = out \rightarrow f(out) = \hat{y} \quad \text{output layer}$$

求 $\Delta W_{h \rightarrow o}$ 梯度. Loss func $MSE = \frac{1}{2m} (\hat{y} - y)^2$

$$\begin{aligned} \frac{\partial MSE}{\partial w_i^1} &= \frac{\partial \frac{1}{2m} (\hat{y} - y)^2}{\partial \hat{y}} \cdot \frac{\partial f(out)}{\partial out} \cdot \frac{\partial out}{\partial w_i^1} \\ &= \frac{1}{m} (\hat{y} - y) \cdot 1 \cdot \frac{\partial (w_1^1 \cdot b(h_1) + w_2^1 \cdot b(h_2) + w_3^1 \cdot b(h_3) + w_4^1 \cdot b(h_4))}{\partial w_i^1} \\ &= \frac{1}{m} (\hat{y} - y) \cdot b(h_i) \end{aligned}$$

$$\Delta W_{h \rightarrow o} = \frac{(\hat{y} - y)}{m} \cdot \begin{bmatrix} b(h_1) \\ b(h_2) \\ b(h_3) \\ b(h_4) \end{bmatrix} = \boxed{\frac{1}{m} (\hat{y} - y) \cdot \text{hidden_outputs}}$$

改进

求 $\Delta W_{i \rightarrow h}$ 梯度

$$\begin{aligned} \frac{\partial MS_E}{\partial w_{11}^0} &= \frac{\partial \frac{1}{2m}(\hat{y} - y)^2}{\partial \hat{y}} \cdot \frac{\partial \hat{y}}{\partial out} \cdot \frac{\partial out}{\partial b(h_1)} \cdot \frac{\partial b(h_1)}{\partial h_1} \cdot \frac{\partial h_1}{\partial w_{11}^0} \\ &= \frac{1}{m}(\hat{y} - y) \cdot 1 \cdot \frac{\partial (w_1' \cdot b(h_1) + w_2' \cdot b(h_2) + \dots)}{\partial b(h_1)} \cdot b(h_1)(1 - b(h_1)) \cdot \frac{\partial (x_1 \cdot w_{11}^0 + x_2 \cdot w_{21}^0 + x_3 \cdot w_{31}^0)}{\partial w_{11}^0} \\ &= \frac{1}{m}(\hat{y} - y) \cdot w_1' \cdot b(h_1)(1 - b(h_1)) \cdot x_1 \end{aligned}$$

方法

梯度矩阵

$$\begin{bmatrix} \frac{\partial MS_E}{\partial w_{11}^0} & \frac{\partial MS_E}{\partial w_{12}^0} & \frac{\partial MS_E}{\partial w_{13}^0} & \frac{\partial MS_E}{\partial w_{14}^0} \\ \frac{\partial MS_E}{\partial w_{21}^0} & - & - & - \\ \frac{\partial MS_E}{\partial w_{31}^0} & \frac{\partial MS_E}{\partial w_{32}^0} & - & \frac{\partial MS_E}{\partial w_{34}^0} \end{bmatrix} = \frac{1}{m}(\hat{y} - y) \begin{bmatrix} w_1' \cdot b(h_1)(1 - b(h_1)) \cdot x_1 & w_2' \cdot b(h_2)(1 - b(h_2)) \cdot x_1 & w_3' \cdot b(h_3)(1 - b(h_3)) \cdot x_1 & w_4' \cdot b(h_4)(1 - b(h_4)) \cdot x_1 \\ w_1' \cdot b(h_1)(1 - b(h_1)) \cdot x_2 & w_2' \cdot b(h_2)(1 - b(h_2)) \cdot x_2 & & x_2 \\ w_1' \cdot b(h_1)(1 - b(h_1)) \cdot x_3 & & x_3 & x_3 \end{bmatrix}$$

改进

$$\frac{1}{m} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} \cdot (\hat{y} - y) [b(h_1)(1 - b(h_1)), b(h_2)(1 - b(h_2)), b(h_3)(1 - b(h_3)), b(h_4)(1 - b(h_4))] \cdot (w_1', w_2', w_3', w_4')$$

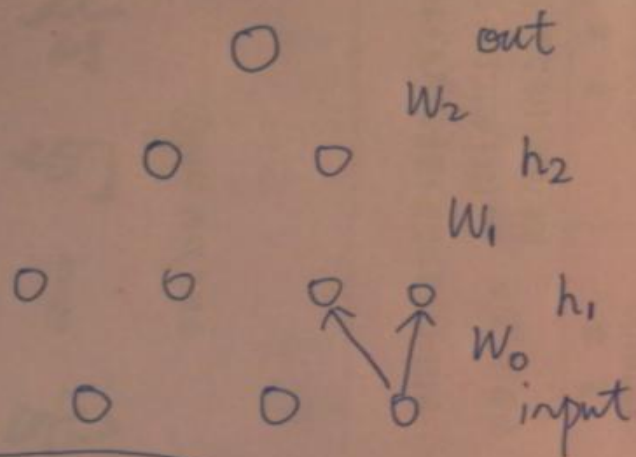
hidden-error

向量相乘

矩阵乘法

让坚持成就梦想

天才出自勤奋。



正向 pass

$$\begin{cases} \text{input}(\text{Batch}, 3) \times W_0(3, 4) = h_1(\text{Batch}, 4) \\ g(h_1)(\text{Batch}, 4) \end{cases}$$

$$\begin{cases} g(h_1)(\text{Batch}, 4) \times W_1(4, 2) = h_2(\text{Batch}, 2) \\ g(h_2)(\text{Batch}, 2) \end{cases}$$

$$\begin{cases} g(h_2)(\text{Batch}, 2) \times W_2(2, 1) = \text{out}(\text{Batch}, 1) \\ \hat{y} = \text{out}(\text{Batch}, 1) \Rightarrow \text{Loss} = \frac{1}{2m} (y - \hat{y})^2 \end{cases}$$

反向: W_2 的梯度

$$\frac{\partial \text{Loss}}{\partial W_2} = \frac{\partial \text{Loss}}{\partial \hat{y}} \cdot \frac{\partial \hat{y}}{\partial \text{out}} \cdot \frac{\partial \text{out}}{\partial W_2}$$

$$= (\hat{y} - y) \cdot 1 \cdot g(h_2)$$

batch 记为 2

$$= \begin{bmatrix} \hat{y}_1 - y_1 \\ \hat{y}_2 - y_2 \end{bmatrix} \cdot 1 \cdot g(h_2) \leftarrow (\text{Batch}, 2)$$

$$= \begin{bmatrix} \text{error}_1 \\ \text{error}_2 \end{bmatrix} \cdot 1 \cdot \begin{bmatrix} g(h_2)^T \end{bmatrix} \leftarrow (2, \text{Batch})$$

(2, 1) 2

$$= (2, 1) / \text{batch} \xrightarrow{\text{shape 等于}} W_2$$

(W₁ 的梯度)

$$\frac{\partial \text{Loss}}{\partial W_1} = \frac{\partial \text{Loss}}{\partial \hat{y}} \cdot \frac{\partial \hat{y}}{\partial \text{out}} \cdot \frac{\partial \text{out}}{\partial b(h_2)} \cdot \frac{\partial b(h_2)}{\partial h_2} \cdot \frac{\partial h_2}{\partial W_1}$$

$$= \begin{bmatrix} \text{error}_1 \\ \text{error}_2 \end{bmatrix} \cdot 1 \cdot W_2 \cdot \begin{bmatrix} b(h_2^{11})(1-b(h_2^{11})) & b(h_2^{12})(1-b(h_2^{12})) \\ b(h_2^{21})(1-b(h_2^{21})) & b(h_2^{22})(1-b(h_2^{22})) \end{bmatrix} \cdot b(h_1)$$

\downarrow (batch, 1) \downarrow (2, 1) 权重 \downarrow (2, 2) batch \downarrow (Batch, 4) 估计结果

$$= \begin{bmatrix} \text{error}_1 \\ \text{error}_2 \end{bmatrix} \cdot W_2^T \cdot \begin{bmatrix} \text{batch}, 2 \end{bmatrix} \cdot b(h_1)^T$$

\downarrow (batch, 1) \downarrow (1, 2) \downarrow (4, batch) 这个提前

$$= \begin{bmatrix} \text{batch}, 2 \end{bmatrix}$$

\downarrow 元素相乘 \downarrow (batch, 2) \downarrow (4, 2) / batch shape 和 W₁ 同

$$\frac{\partial \text{Loss}}{\partial W_0} = \frac{\partial \text{Loss}}{\partial \hat{y}} \cdot \frac{\partial \hat{y}}{\partial \text{out}} \cdot \frac{\partial \text{out}}{\partial b(h_2)} \cdot \frac{\partial b(h_2)}{\partial h_2} \cdot \frac{\partial h_2}{\partial b(h_1)} \cdot \frac{\partial b(h_1)}{\partial h_1} \cdot \frac{\partial h_1}{\partial W_0}$$

$$= \text{hidden_error_term}(h_2) \cdot W_1 \cdot (b(h_1))' \cdot \text{input}^T$$

(batch, 2) (4, 2) (Batch, 4) (3, batch)

\downarrow
 W_1^T
 (2, 4)

\swarrow np.dot \searrow
 (batch, 4)

元素级乘法
 (batch, 4)

\swarrow np.dot \searrow
 (3, 4) / batch

shape 和 W_0 相同.

input - 2 - h_1 层的梯度

(这个在之前)