ANN - DIGIT CLASSIFICATION

Part 1 - Problem Statement:

A Digit classifying model of MNIST Dataset (Kaggle)

Part 2 - Math

pixel value : 0 → 255
black white

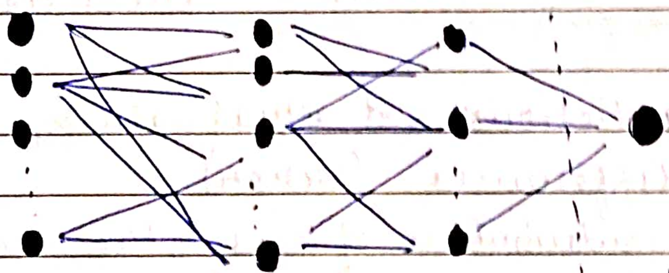
$\boxed{784}$ 28 px

28 px

I/P ⇒ $X = \begin{bmatrix} — x^{[1]} — \\ — x^{[2]} — \\ \vdots \\ — x^{[m]} — \end{bmatrix}^T = \begin{bmatrix} | & | & & | \\ x^{[1]} & x^{[2]} & \cdots & x^{[m]} \\ | & | & & | \end{bmatrix}$

m examples

$x^{[1]}$ contains 784 rows



Input node : 784   10   10   1

layer : [0]   [1]   [2]   [3]

input   Hidden   Final   Output

Forward Propagation:

1st layer

$$A^{[0]} = X \ (784 \times m)$$

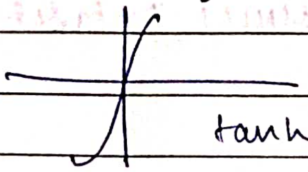
$$\underset{(10 \times m)}{Z^{[1]}} = \underset{\substack{\text{weight} \\ (10 \times 784)}}{W^{[1]}} \cdot \underset{\substack{\text{i/p} \\ (784 \times m)}}{A^{[0]}} + \underset{\substack{\text{bias} \\ (10 \times 1)}}{b^{[1]}}$$

2nd layer

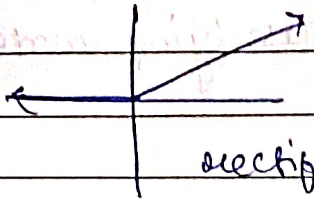
$$A^{[1]} = g(Z^{[1]}) = \mathrm{Relu}(Z^{[1]})$$

$$\underset{(10 \times m)}{Z^{[2]}} = \underset{(10 \times 10)}{W^{[2]}} \underset{(10 \times m)}{A^{[1]}} + \underset{(10 \times 1)}{b^{[2]}}$$

Activation functions:



tanh



$\begin{cases} \text{if } x > 0, & x \\ \text{if } x \le 0, & 0 \end{cases}$

rectified linear unit (ReLu)

→ to avoid nodes being the linear combination of previous layer.

3rd Layer $\{ [A^{[3]} = \text{softmax}(z^{[2]})]$

Output layer $\left\{ \begin{bmatrix} 1.3 \\ 5.1 \\ 2.2 \\ \vdots \\ 1.1 \end{bmatrix} \Rightarrow \begin{bmatrix} \dfrac{e^{z_i}}{\sum_{j=0}^{*} e^{z_j}} \end{bmatrix} \Rightarrow \begin{bmatrix} 0.02 \\ 0.91 \\ 0.03 \\ \vdots \\ 0.1 \end{bmatrix} \right.$ — ✓

softmax activation — predictions

Next Goal:
- Get the predictions and actual values,
- Get their differences (error)
- Look for contribution of bias to the error
- Make changes accordingly for better result.

Eg: if $y = 4$,

apply One hot encoding $\Rightarrow \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 1 \\ 0 \\ \vdots \end{bmatrix}$

apply backward propagation,

layer2
$$dz^{[2]} = A^{[2]} - y \qquad \{\text{error in 2nd layer}\}$$
$$\underset{(10\times10)}{dw^{[2]}} = \frac{1}{m}\, \underset{(10\times m)}{dz^{[2]}}\; \underset{(m\times10)}{A^{[1]T}}$$
$$\underset{(10\times1)}{db^{[2]}} = \frac{1}{m} \underset{(10\times1)}{\sum dz^{[2]}}$$

layer 1

$$\underset{(10\times m)}{dZ^{[1]}} = \underset{(10\times 10)}{W^{[2]T}} \; \underset{(10\times m)}{dZ^{[2]}} \; .* \; g'(Z^{[1]})$$

↳ derivative of activation function to unwrap it.

$$\underset{(10\times 728)}{dW} = \frac{1}{m} \; \underset{(10\times m)}{dZ} \; \underset{(m\times 728)}{X^{T}}$$

$$\underset{(10\times 1)}{db} = \frac{1}{m} \sum \underset{(10\times 1)}{dZ^{[1]}}$$

Error Reduction (Gradient Descent):

Repeat until convergence {

$$W^{[1]} = W^{[1]} - \alpha \, dW^{[1]}$$
$$b^{[1]} = b^{[1]} - \alpha \, db^{[1]}$$
$$W^{[2]} = W^{[2]} - \alpha \, dW^{[2]}$$
$$b^{[2]} = b^{[2]} - \alpha \, db^{[2]}$$

} α = learning rate (hyperparameter)