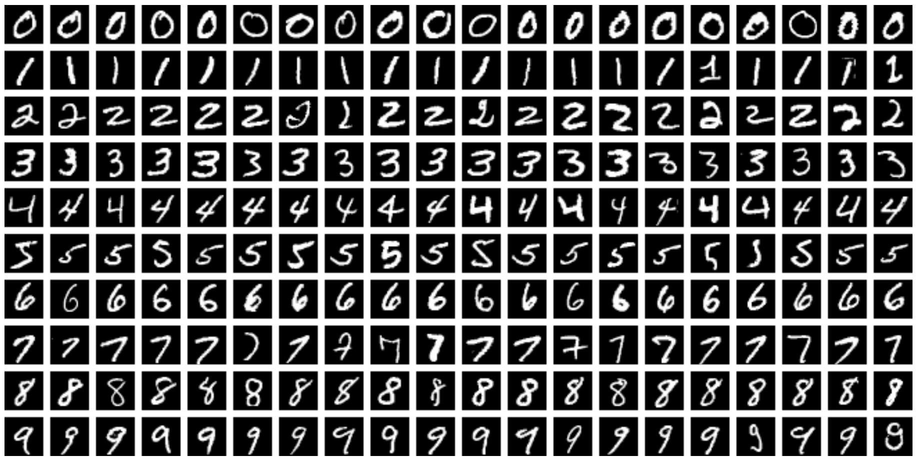
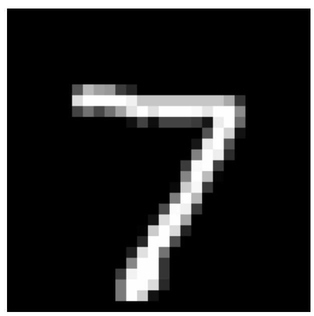
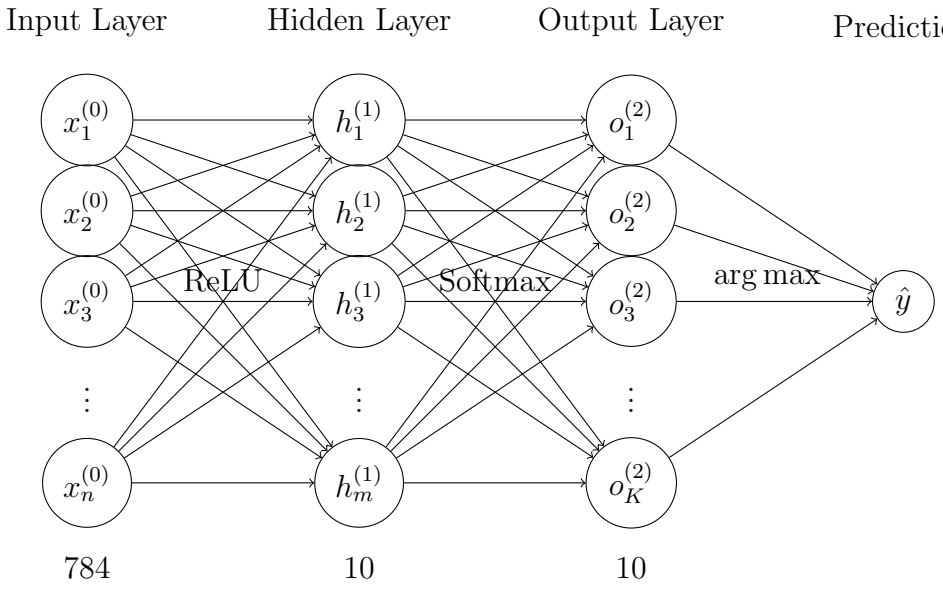
MNIST





```
[[  0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0]
 [  0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0]
 [  0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0]
 [  0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0]
 [  0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0]
 [  0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0]
 [  0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0]
 [  0   0   0   0   0   0  84 185 159 151  60  36   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0]
 [  0   0   0   0   0   0 222 254 254 254 254 241 198 198 198 198 198 198 198 198 170  52   0   0   0   0   0   0]
 [  0   0   0   0   0   0  67 114  72 114 163 227 254 225 254 254 254 250 229 254 254 140   0   0   0   0   0   0]
 [  0   0   0   0   0   0   0   0   0   0   0  17  66  14  67  67  67  59  21 236 254 106   0   0   0   0   0   0]
 [  0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0  83 253 209  18   0   0   0   0   0   0]
 [  0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0  22 233 255  83   0   0   0   0   0   0   0]
 [  0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0 129 254 238  44   0   0   0   0   0   0   0]
 [  0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0  59 249 254  62   0   0   0   0   0   0   0   0]
 [  0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0 133 254 187   5   0   0   0   0   0   0   0   0]
 [  0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   9 205 248  58   0   0   0   0   0   0   0   0   0]
 [  0   0   0   0   0   0   0   0   0   0   0   0   0   0   0 126 254 182   0   0   0   0   0   0   0   0   0   0]
 [  0   0   0   0   0   0   0   0   0   0   0   0   0   0  75 251 240  57   0   0   0   0   0   0   0   0   0   0]
 [  0   0   0   0   0   0   0   0   0   0   0   0   0  19 221 254 166   0   0   0   0   0   0   0   0   0   0   0]
 [  0   0   0   0   0   0   0   0   0   0   0   0   3 203 254 219  35   0   0   0   0   0   0   0   0   0   0   0]
 [  0   0   0   0   0   0   0   0   0   0   0   0  38 254 254  77   0   0   0   0   0   0   0   0   0   0   0   0]
 [  0   0   0   0   0   0   0   0   0   0   0  31 224 254 115   1   0   0   0   0   0   0   0   0   0   0   0   0]
 [  0   0   0   0   0   0   0   0   0   0   0 133 254 254  52   0   0   0   0   0   0   0   0   0   0   0   0   0]
 [  0   0   0   0   0   0   0   0   0   0  61 242 254 254  52   0   0   0   0   0   0   0   0   0   0   0   0   0]
 [  0   0   0   0   0   0   0   0   0   0 121 254 254 219  40   0   0   0   0   0   0   0   0   0   0   0   0   0]
 [  0   0   0   0   0   0   0   0   0   0 121 254 207  18   0   0   0   0   0   0   0   0   0   0   0   0   0   0]
 [  0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0   0]]
```

Neural Network



- **Neuron** - Basic computational unit that processes inputs
- **Weight** (w) - Connection strength between neurons
- **Bias** (b) - Offset value added to weighted sum
- **Forward propagation** - Computing outputs from inputs through the network
- **Logits** (z) - Pre-activation output
- **Activation** (a) - Non-linear function applied to neuron outputs
- **Backward propagation** - Computing gradients from outputs back to inputs
- **One-hot** - Binary vector representation of categorical labels
- **Loss** (l) - Measure of prediction error to be minimized
- **Gradient** (d) - Derivative of loss with respect to a parameter
- **Cross-entropy** - Loss function that measures prediction accuracy
- **Gradient descent** - Optimization algorithm that updates weights and biases
- **Learning rate** (α) - Step size for parameter updates in gradient descent

1. Forward propagation produces predictions

2. Loss measures how wrong we are
3. Backward propagation computes gradients from the loss
4. Gradient descent uses those gradients to update parameters

Data Structure

$$X = \begin{bmatrix} \cdots & x^{(1)} & \cdots \\ \cdots & x^{(2)} & \cdots \\ \vdots & \vdots & \vdots \\ \cdots & x^{(m)} & \cdots \end{bmatrix}^{\top} = \begin{bmatrix} x_1^{(1)} & x_1^{(2)} & \cdots & x_1^{(m)} \\ x_2^{(1)} & x_2^{(2)} & \cdots & x_2^{(m)} \\ \vdots & \vdots & \ddots & \vdots \\ x_n^{(1)} & x_n^{(2)} & \cdots & x_n^{(m)} \end{bmatrix}$$

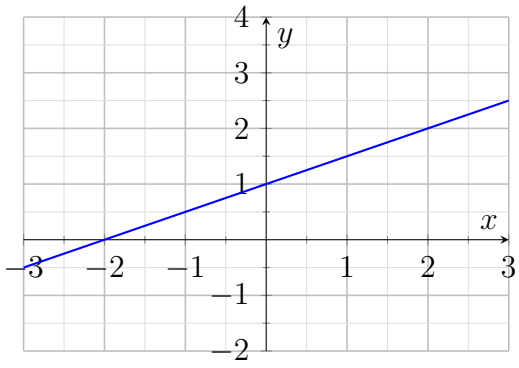
Matrix Multiplication

$$C_{ij} = \sum_{p=1}^{k} A_{ip} \cdot B_{pj}$$

$$\begin{bmatrix} 0.1 & 0.2 & 0.3 \\ 0.4 & 0.5 & 0.6 \end{bmatrix} \begin{bmatrix} 1 \\ 2 \\ 3 \end{bmatrix} = \begin{bmatrix} (0.1 \times 1) + (0.2 \times 2) + (0.3 \times 3) \\ (0.4 \times 1) + (0.5 \times 2) + (0.6 \times 3) \end{bmatrix} = \begin{bmatrix} 1.4 \\ 3.2 \end{bmatrix}$$

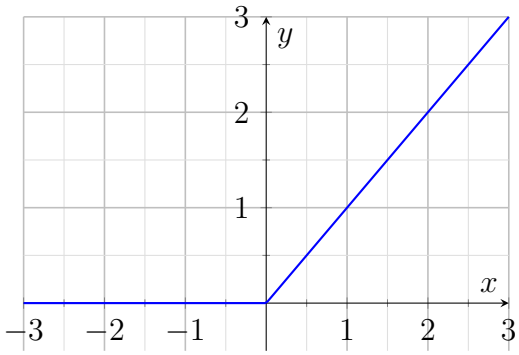
Linear Equation

$$y = ax + b$$



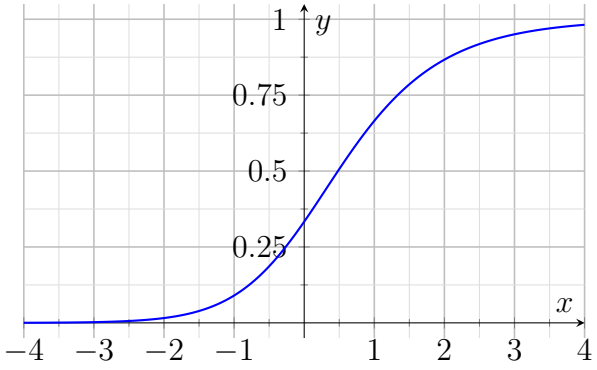
ReLU

$$\text{ReLU}(x) = \max(0, x)$$



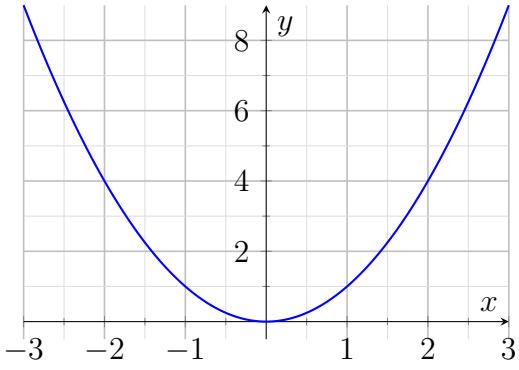
Softmax

$$\text{softmax}(z_i) = \frac{e^{z_i}}{\sum_{j=1}^{K} e^{z_j}} \quad \text{for } i = 1, \ldots, K.$$

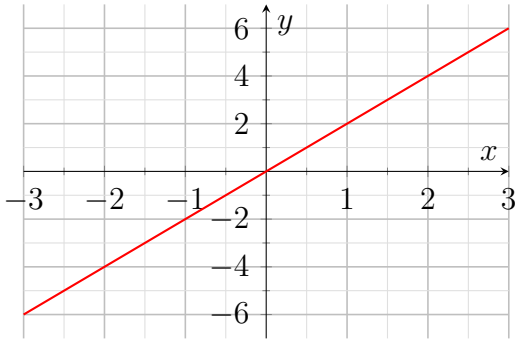


Derivative

$$f(x) = x^2$$



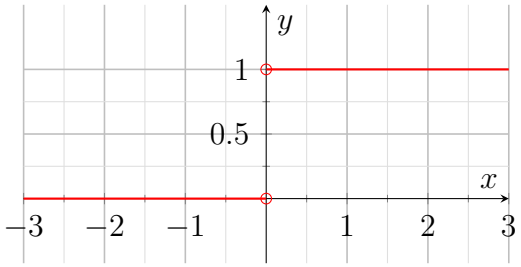
$$f'(x) = 2x$$



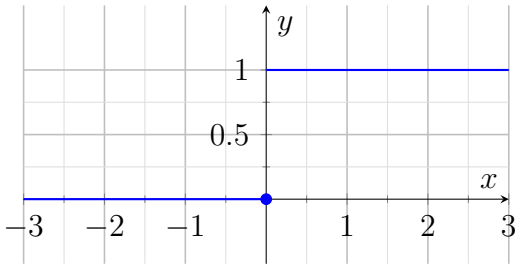
The derivative tells us the **rate of change**: how much the output changes when we make a change to the input.

ReLU Derivative

$$\text{ReLU}'(x) = \begin{cases} 0, & x < 0, \\ 1, & x > 0, \\ \text{undefined}, & x = 0 \end{cases}$$



$$\text{ReLU}'(x) = \begin{cases} 0, & x \leq 0, \\ 1, & x > 0 \end{cases}$$



Cross-Entropy Loss

$$L = -\sum_{i=0}^{C-1} y_i \log(\hat{y}_i)$$

Where:

- C = number of classes
- y_i = true label (one-hot encoded): 1 if class i is correct, 0 otherwise
- $\hat{y}_i$ = predicted probability for class i (output from softmax)

Key Properties:

- If prediction is perfect ($\hat{y} = 1$ for correct class): $L = -\log(1) = 0$
- If prediction is terrible ($\hat{y} \to 0$ for correct class): $L \to \infty$
- Only the correct class contributes to the loss (all other $y_i = 0$)