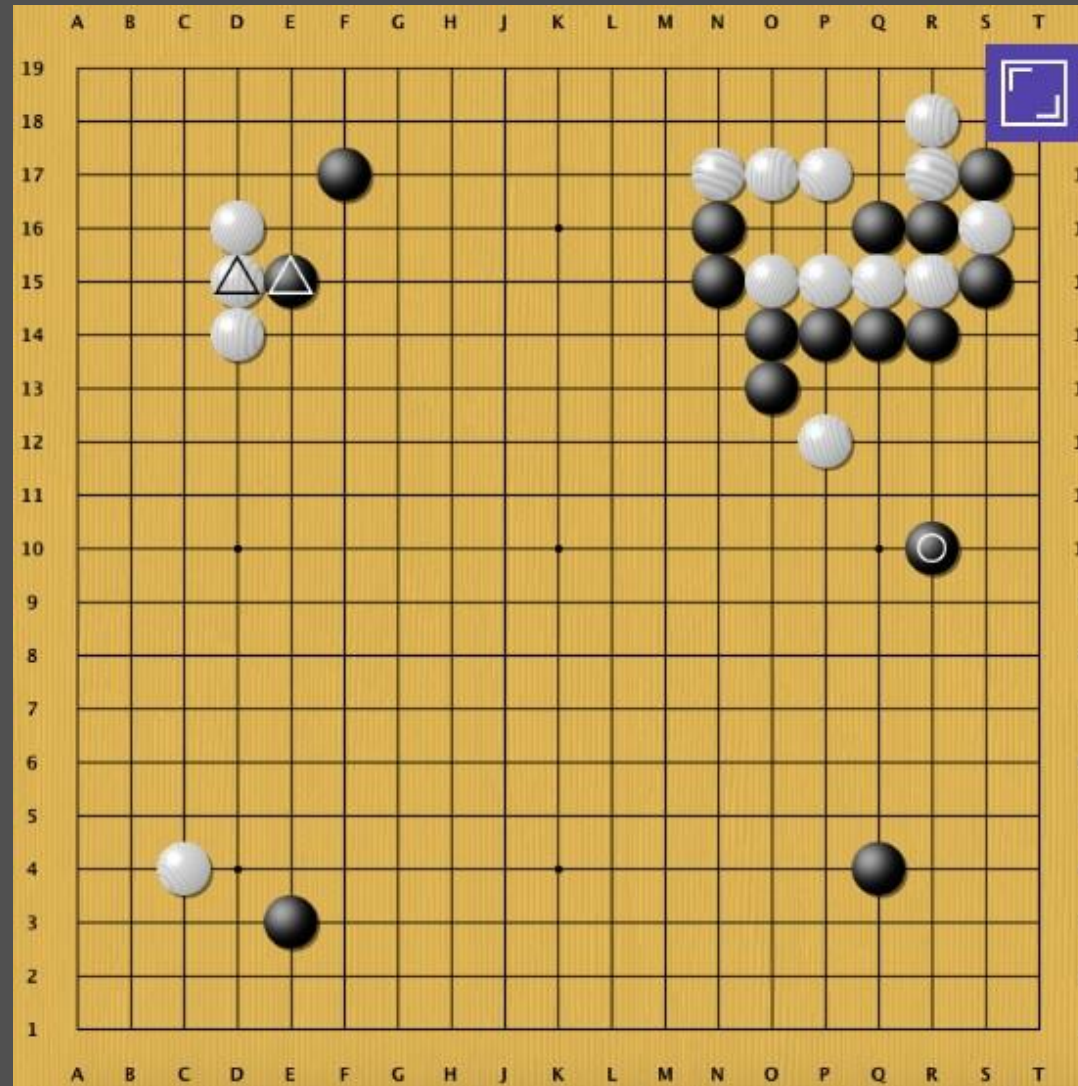


NeuralNets 1

with Neural Nets
MODALG SoSe18

News & Motivation









Robot teaches itself how to dress people





Topics:

- Repetition Classification & Regression
- Learning Rate
- Perceptron & Training
- Multilayer Perceptron

Machine Learning

**Supervised
Learning**

Classification

Regression

**Reinforcement
Learning**

**Unsupervised
Learning**

Clustering

Supervised Learning

Classification

Discrete: $y \in N$



Regression

Continuous: $y \in R$



Data

#Rooms/1	Area/m ²	Price/\$
4	70	1.000
3	65	800
2	40	400
...

Data

Feature 1



#Rooms/1	Area/m ²	Price/\$
4	70	1.000
3	65	800
2	40	400
...

Data

Feature 2



#Rooms/1	Area/m ²	Price/\$
4	70	1.000
3	65	800
2	40	400
...

Data

Feature 3

#Rooms/1	Area/m ²	Price/\$
4	70	1.000
3	65	800
2	40	400
...

Data

Example 1

#Rooms/1	Area/m ²	Price/\$
4	70	1.000
3	65	800
2	40	400
...

Data

Example 2

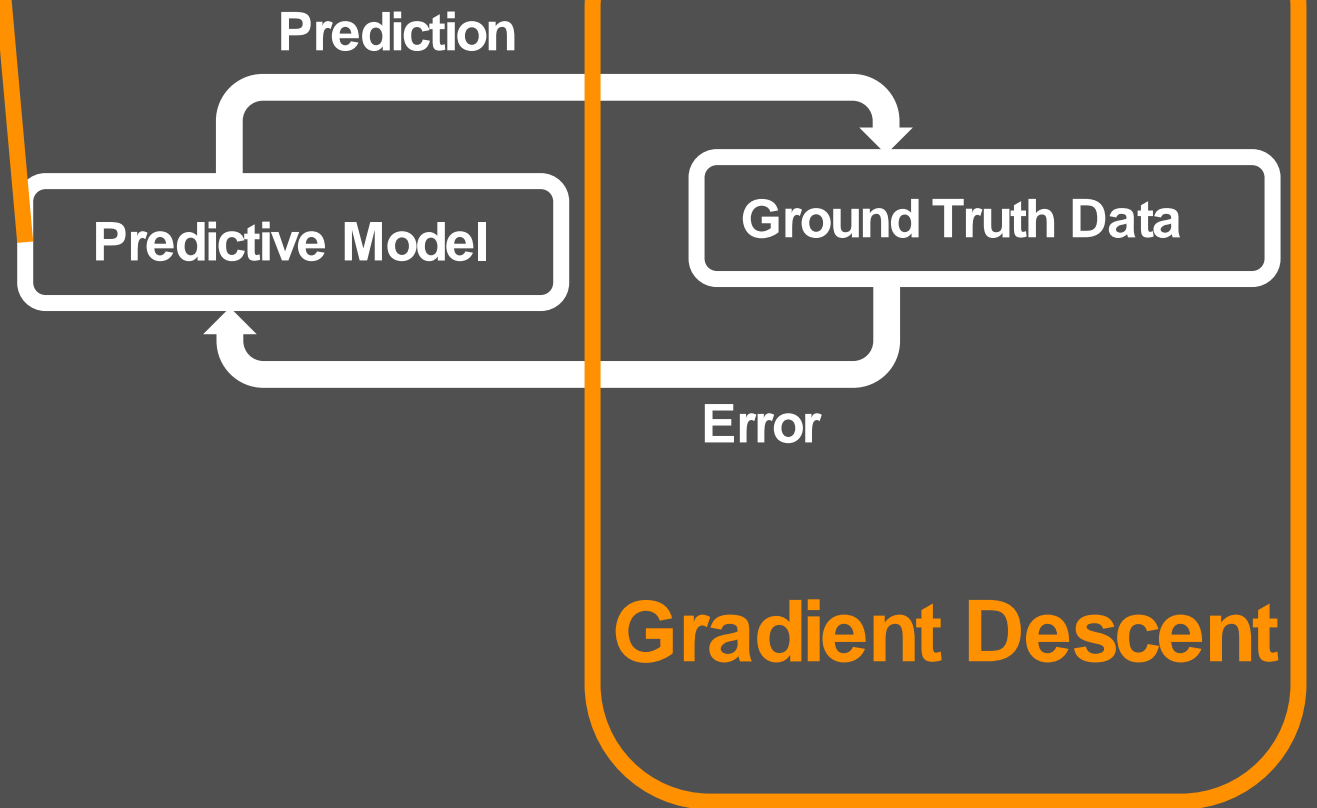
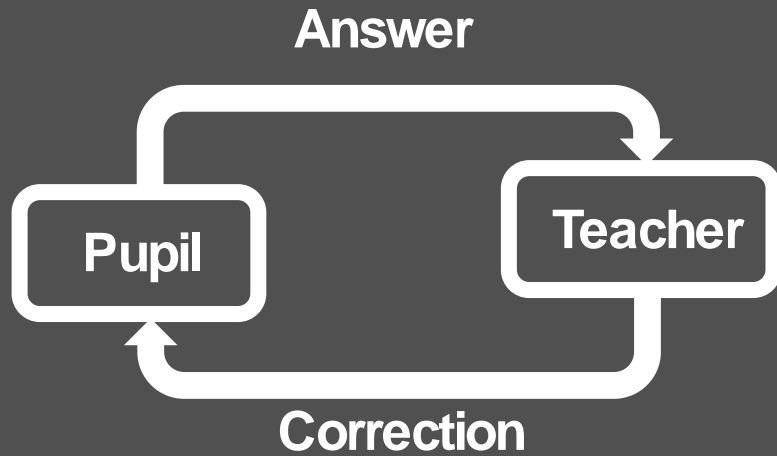
#Rooms/1	Area/m ²	Price/\$
4	70	1.000
3	65	800
2	40	400
...

Data

Example 3

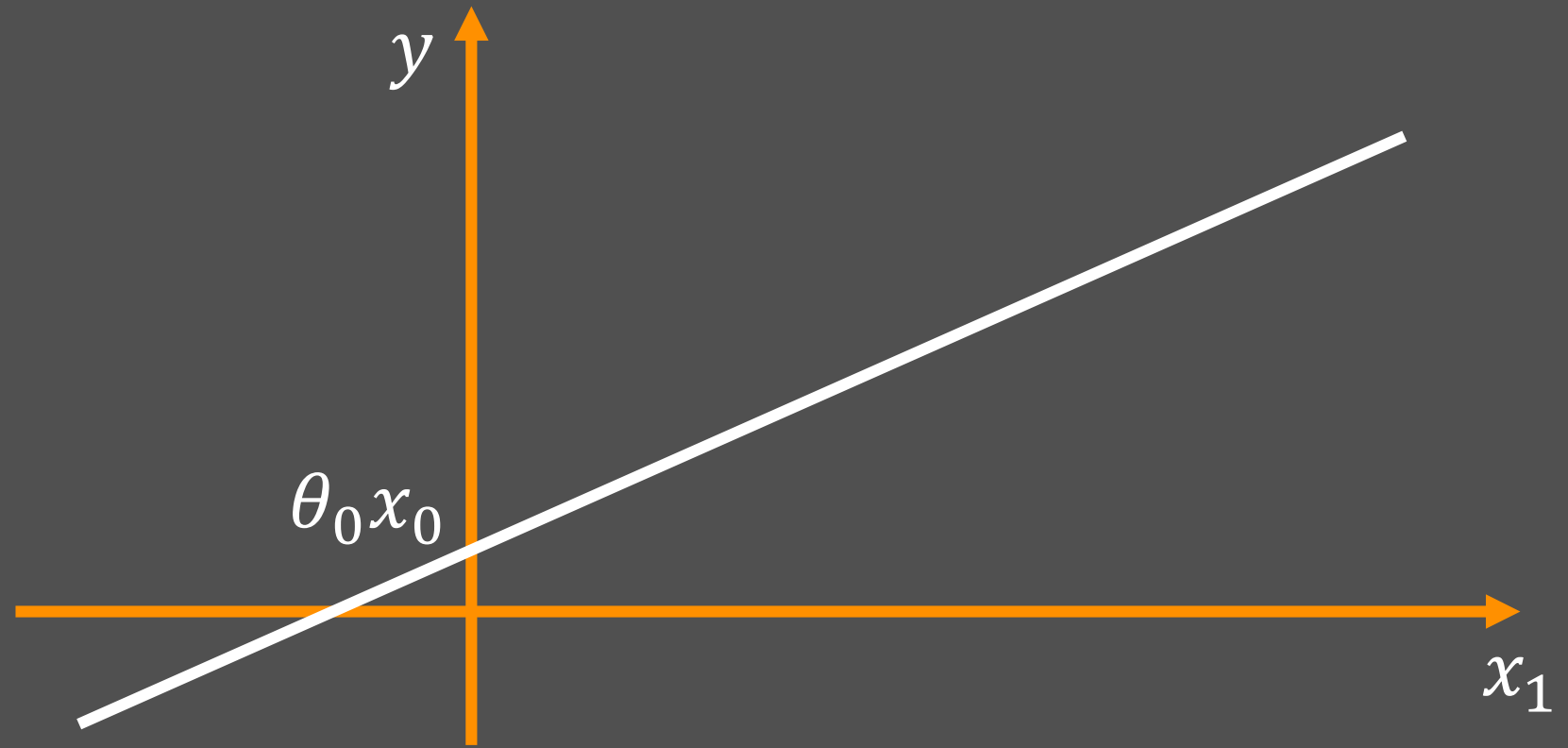
#Rooms/1	Area/m ²	Price/\$
4	70	1.000
3	65	800
2	40	400
...

Linear Regression

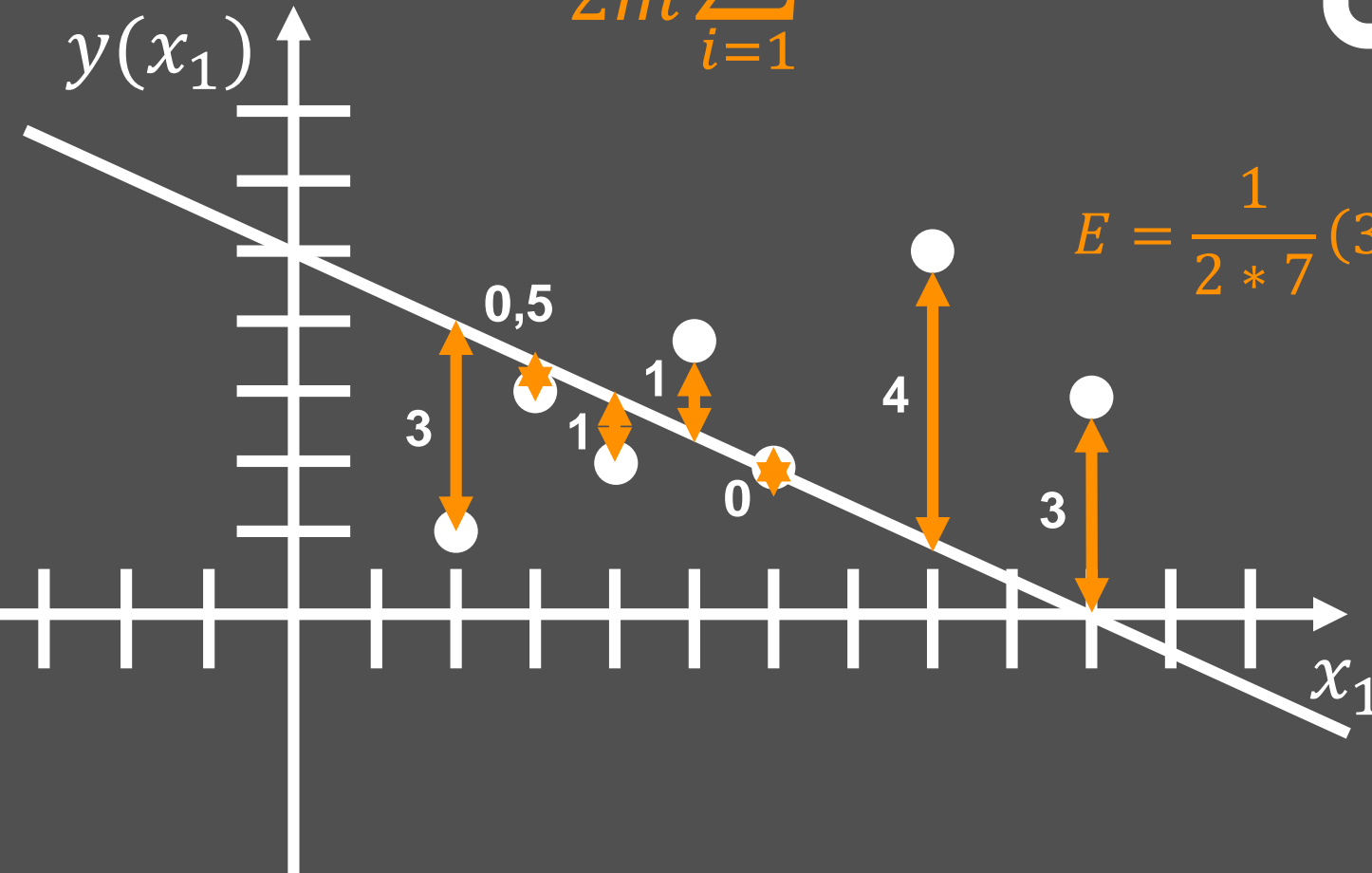
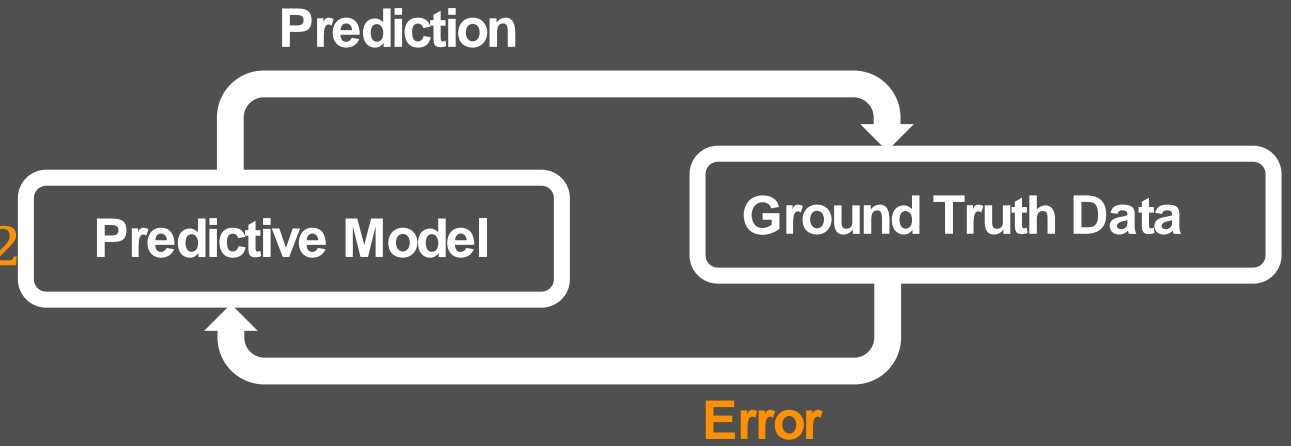


Slope of the line depends on weights θ_0, θ_1

$$\hat{y} = \theta_1 x_1 + \theta_0 x_0$$



$$E = \frac{1}{2m} \sum_{i=1}^m (\hat{y}_i - y_i)^2$$

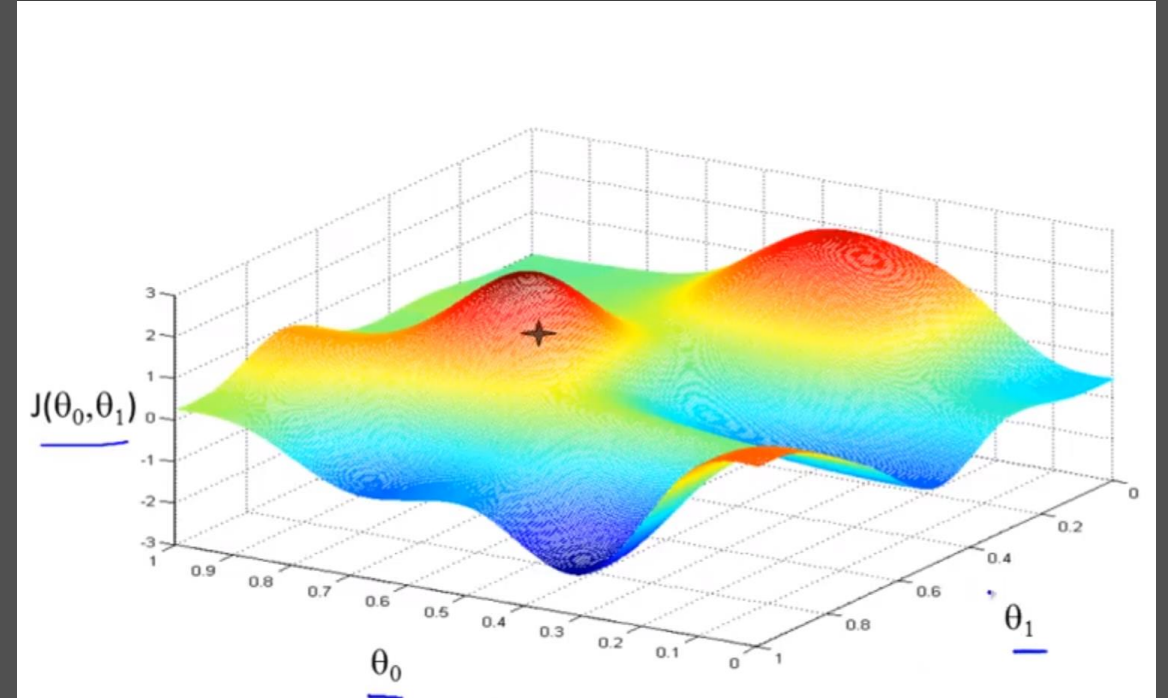
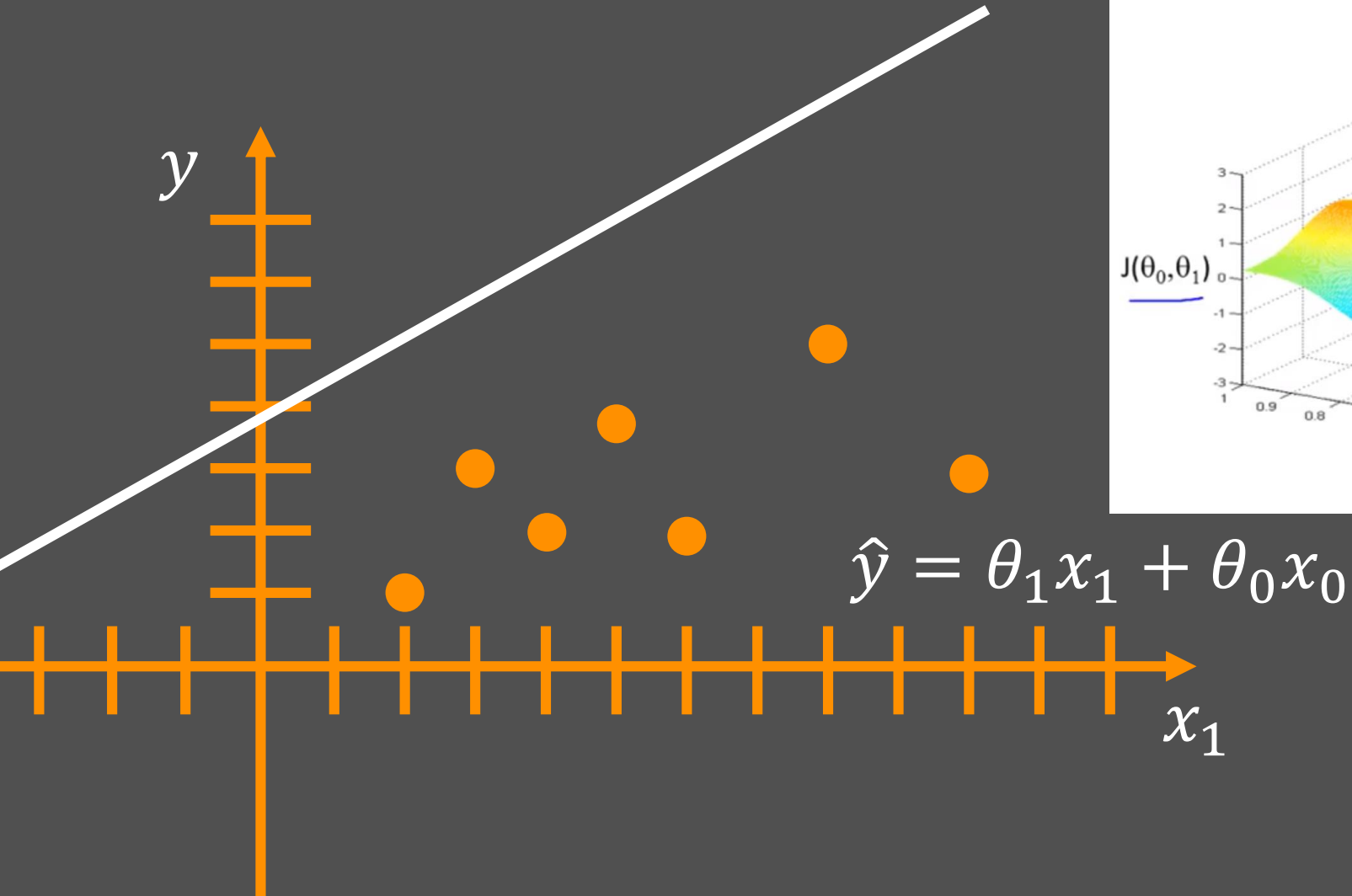


$$E = \frac{1}{2 * 7} (3^2 + 0,5^2 + 1^2 + 1^2 + 0^2 + 4^2 + 3^2)$$

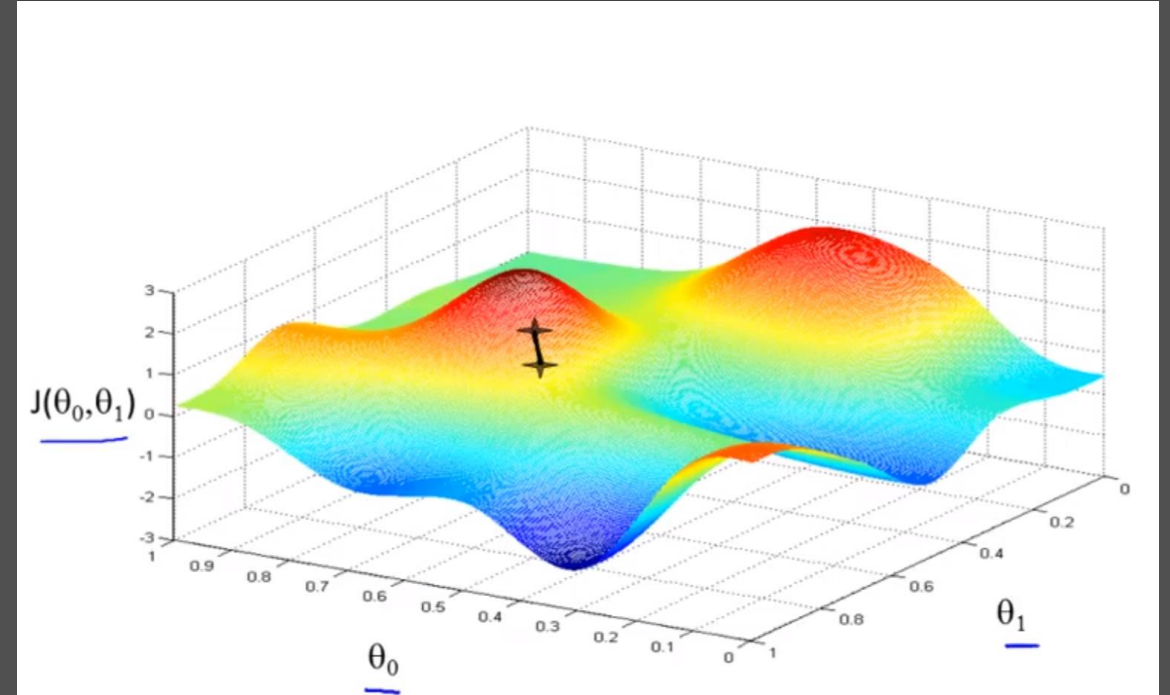
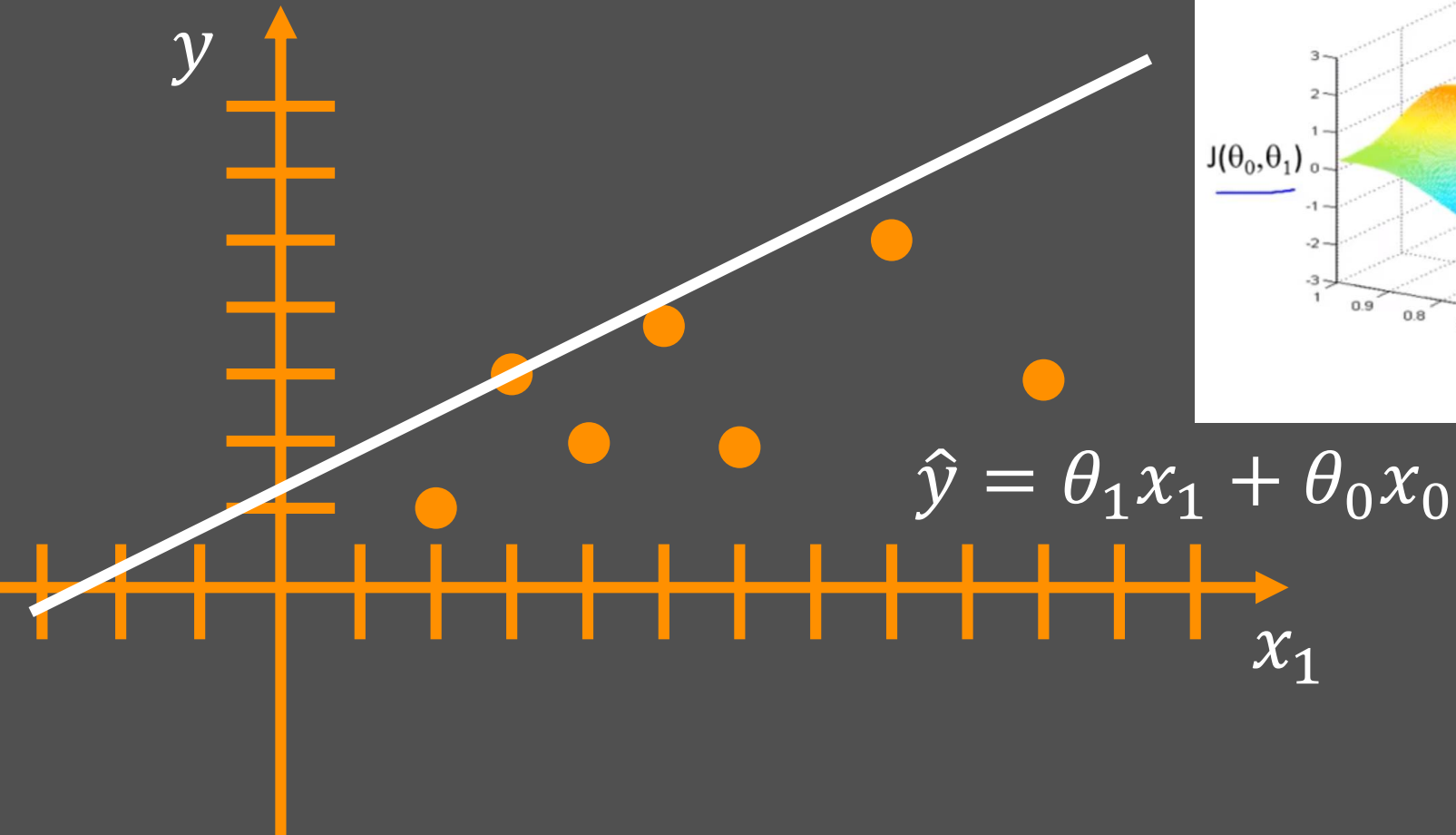
$$E = 2.859$$

$$m = 7$$

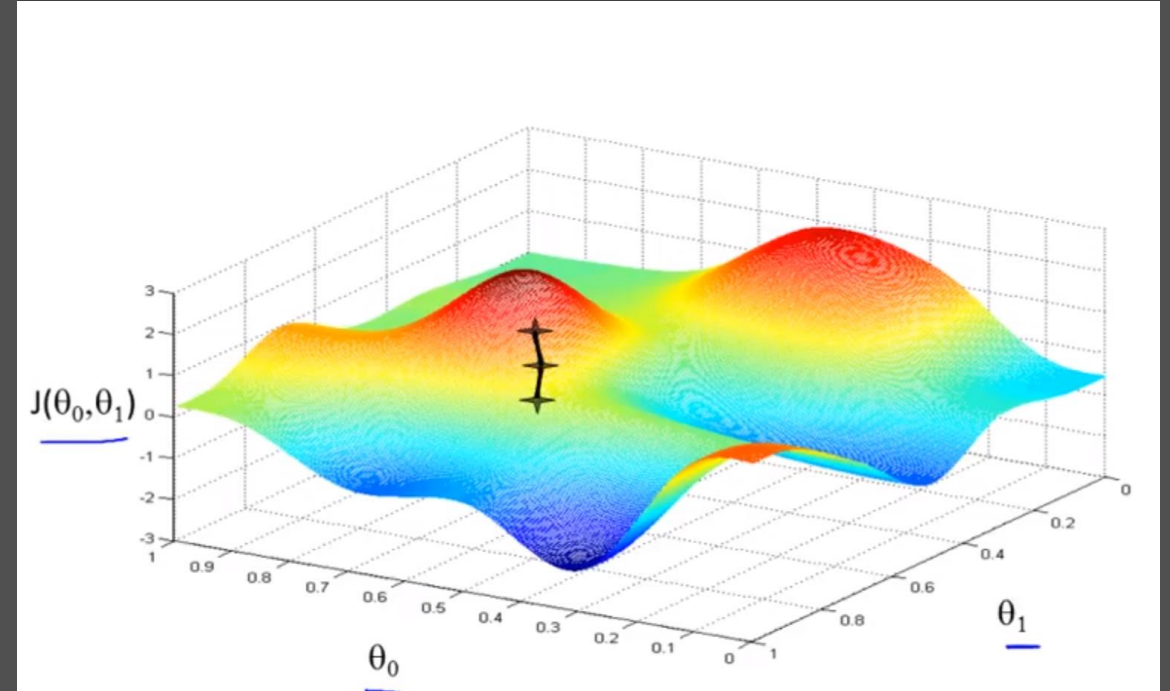
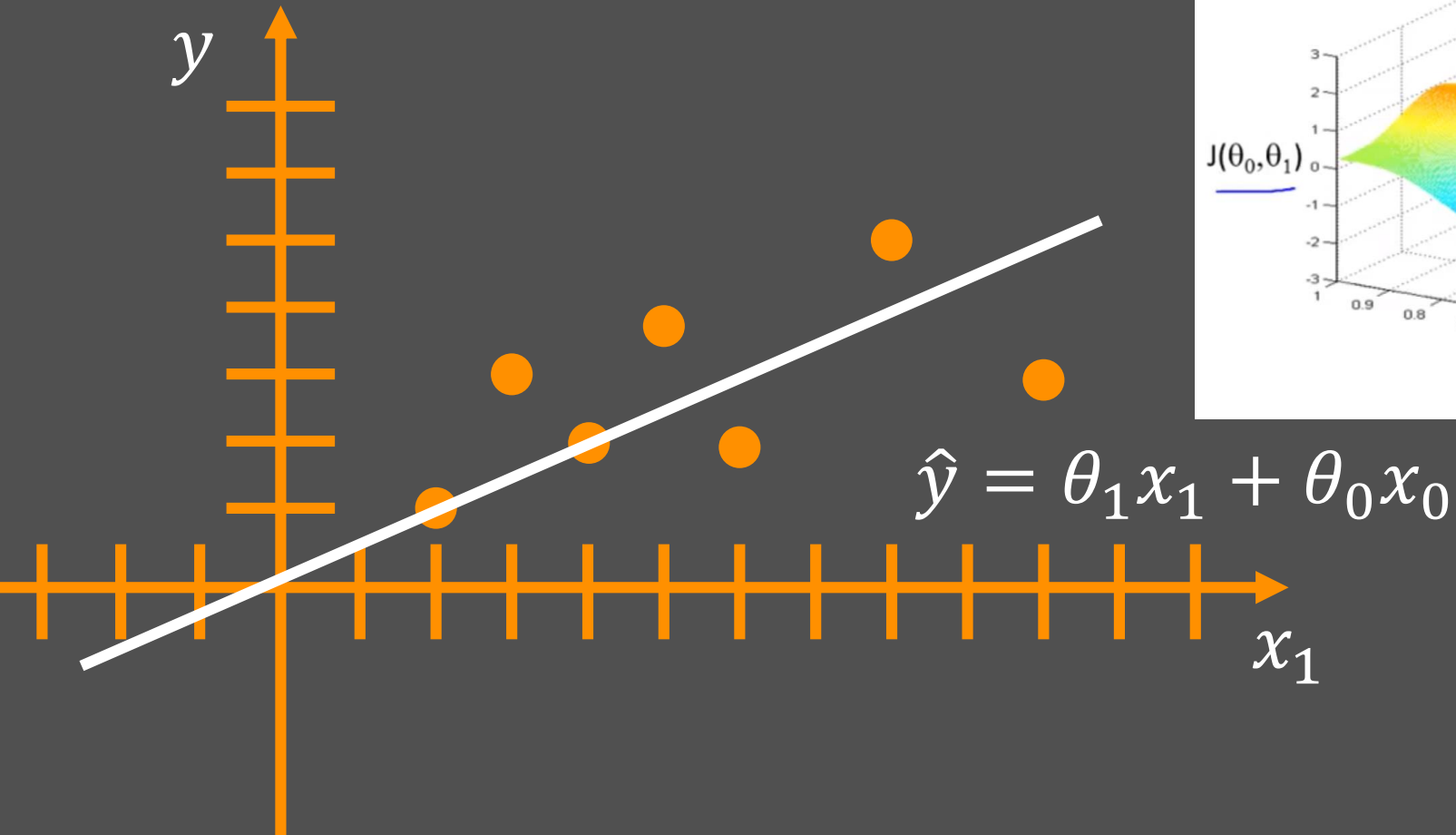
Gradient Descent



Gradient Descent



Gradient Descent



Learning Rate

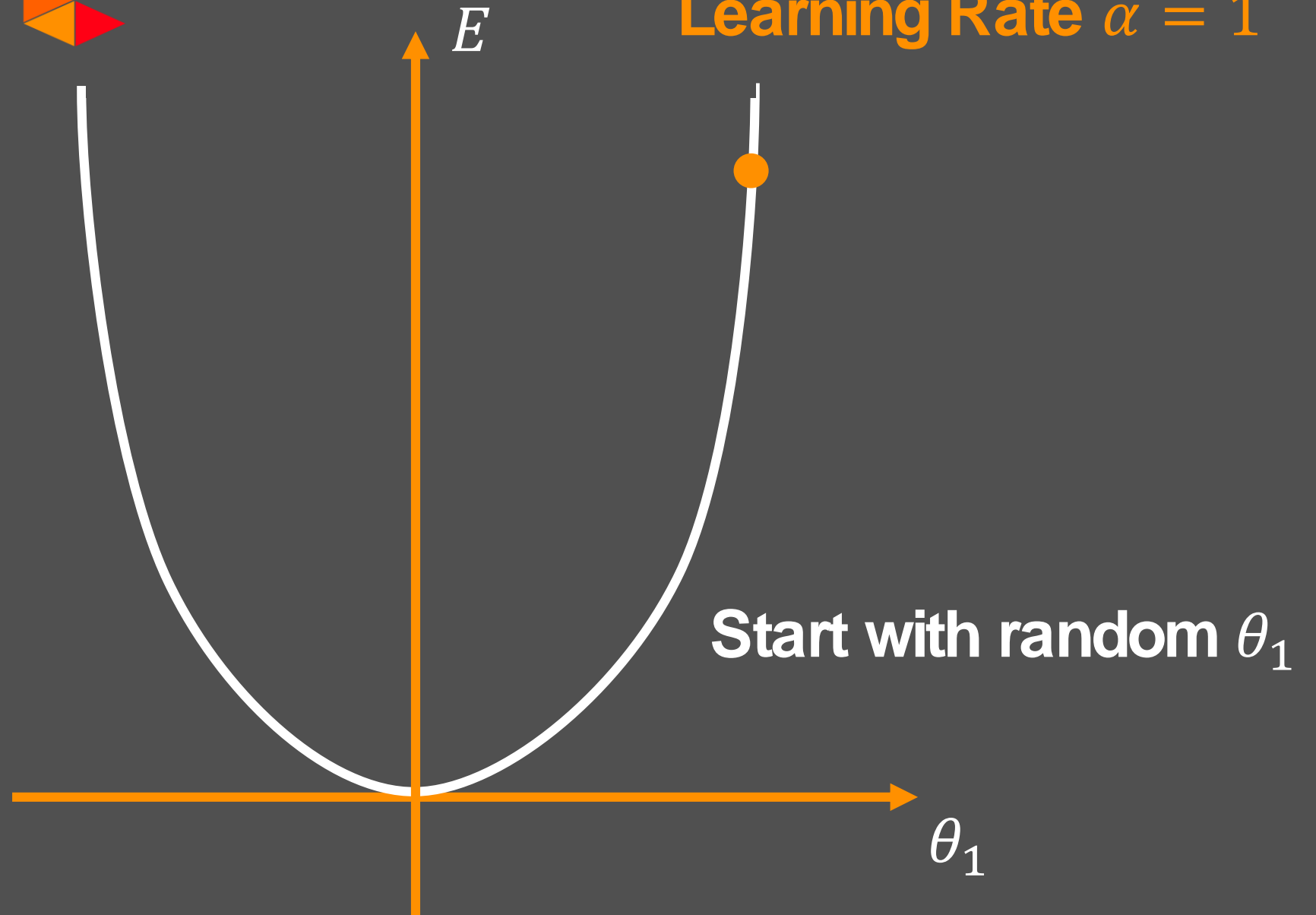
Start with random θ_0, θ_1

repeat until convergence{

$$\theta_0 := \theta_0 - \alpha * \frac{\partial E(\theta_0, \theta_1)}{\partial \theta_0}$$

$$\theta_1 := \theta_1 - \alpha * \frac{\partial E(\theta_0, \theta_1)}{\partial \theta_1}$$

}



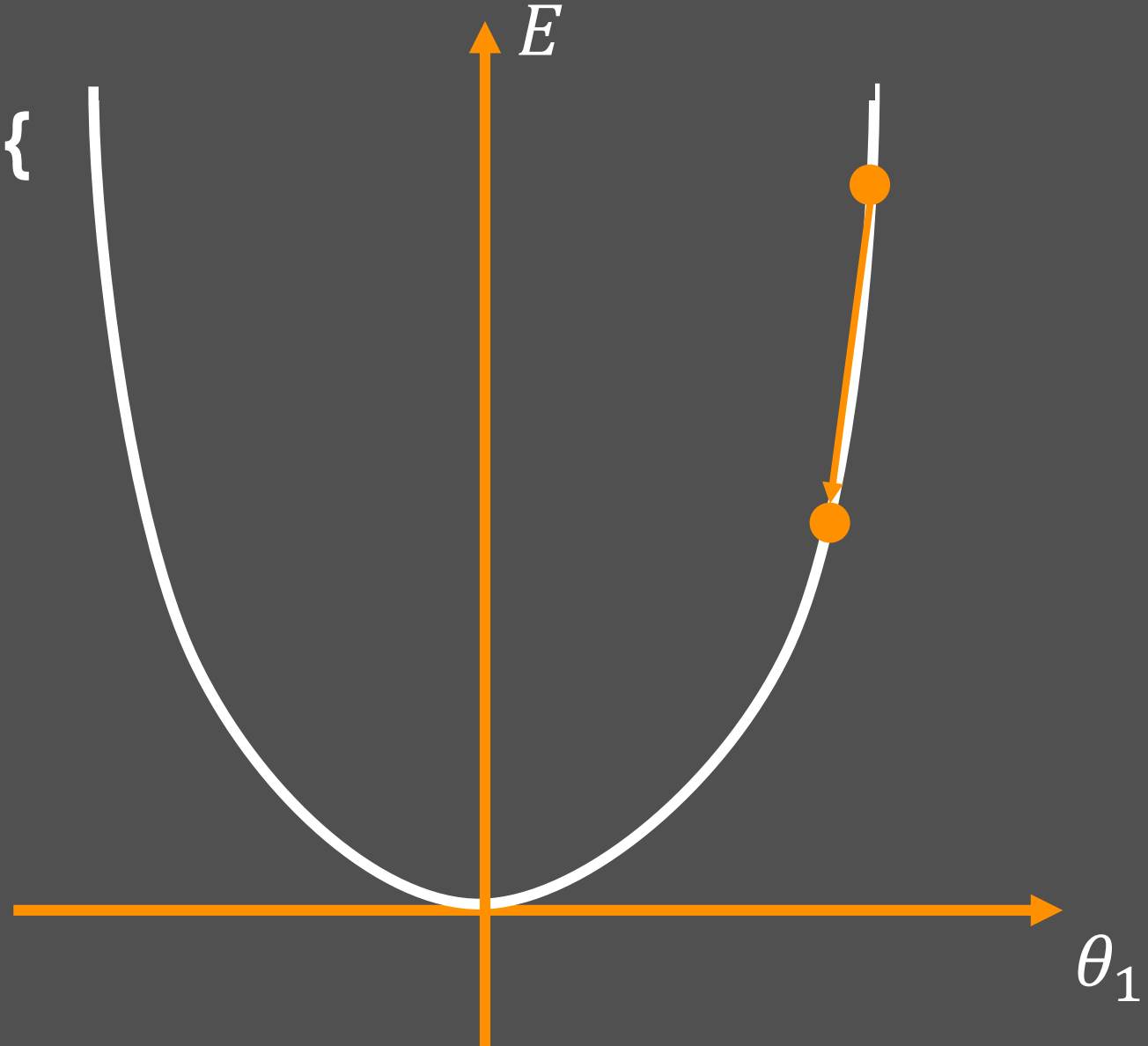
repeat until convergence{

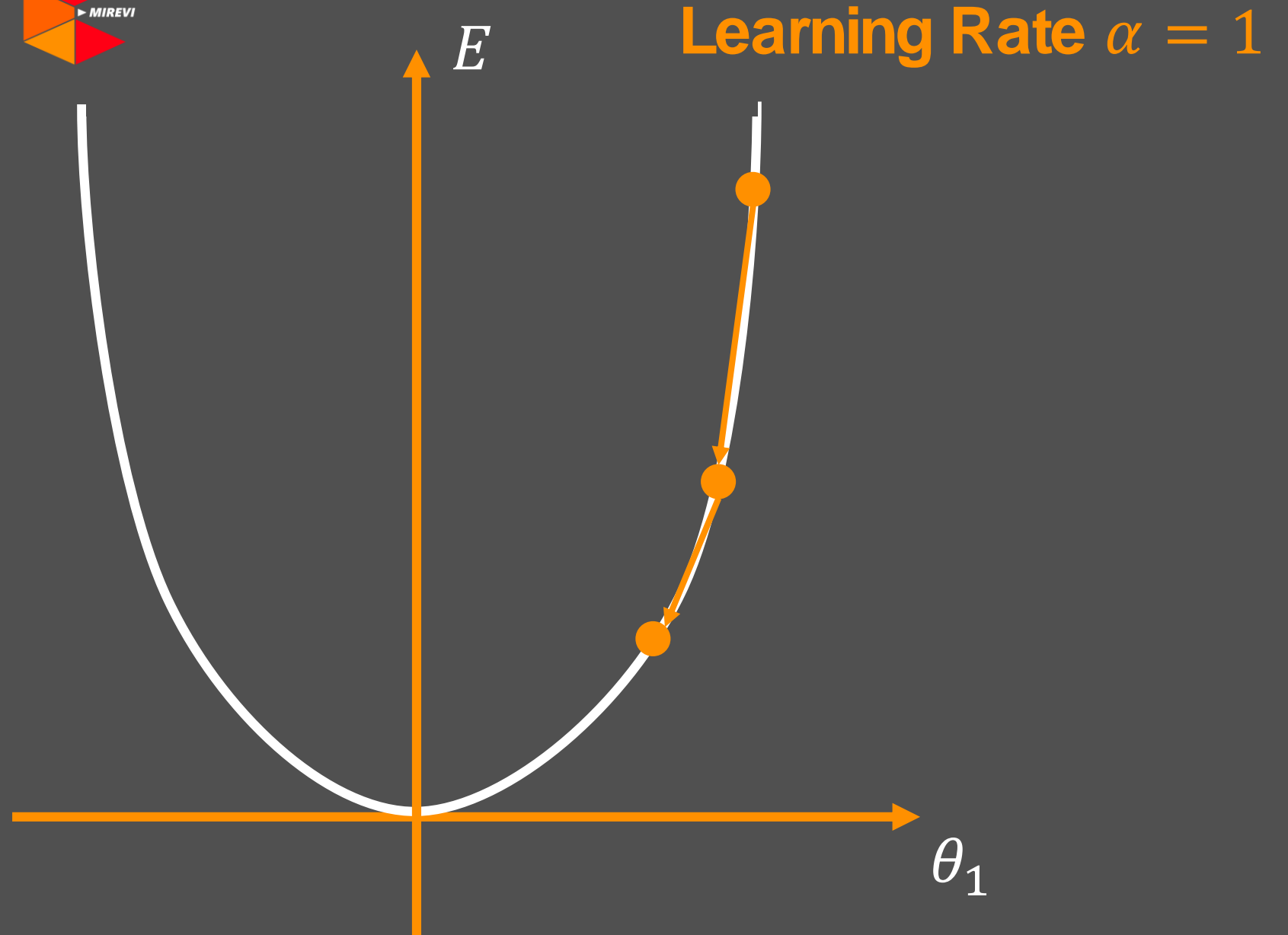
$$\theta_1 := \theta_1 - \alpha * \frac{\partial E(\theta_1)}{\partial \theta_1}$$

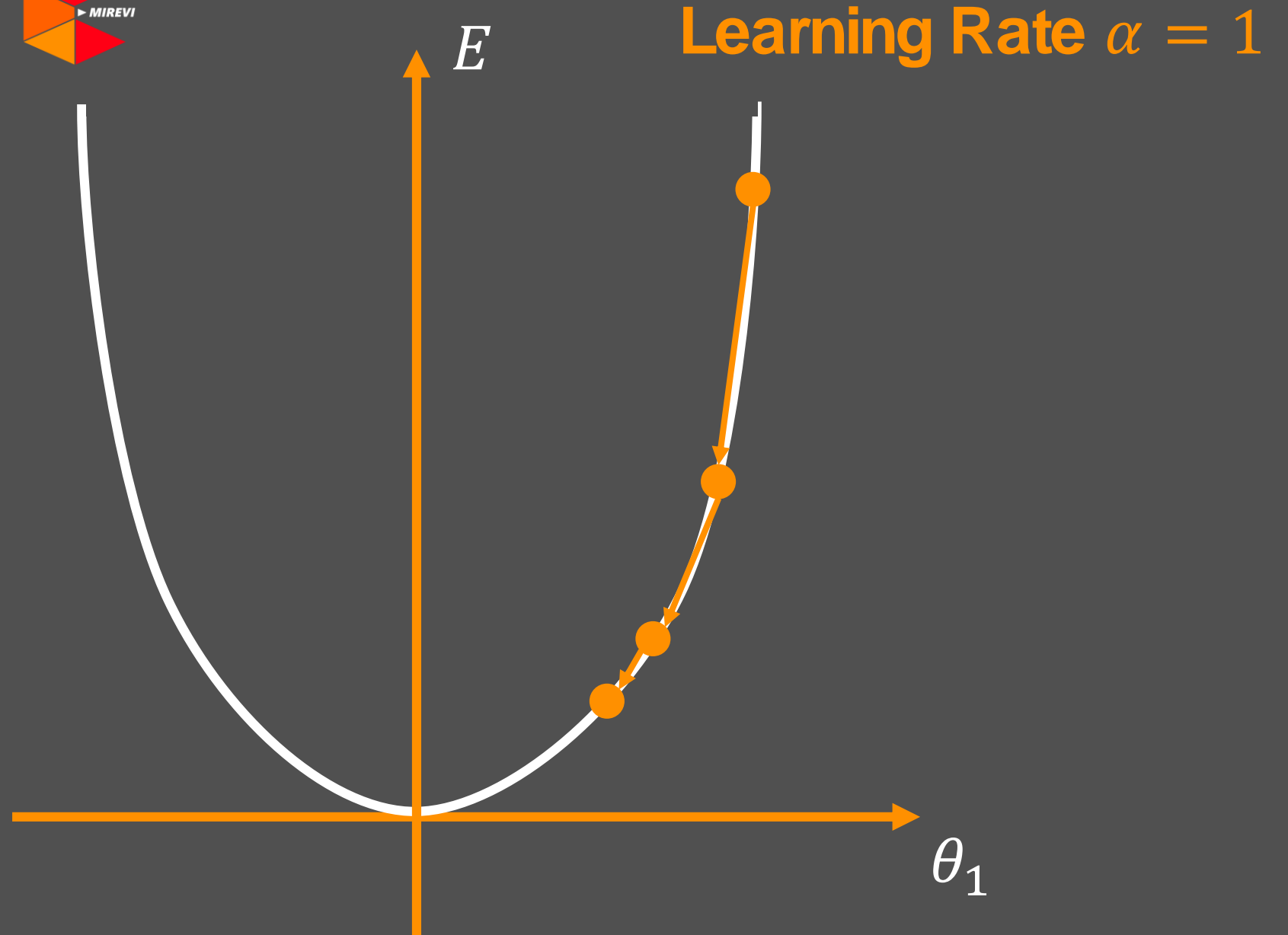
}

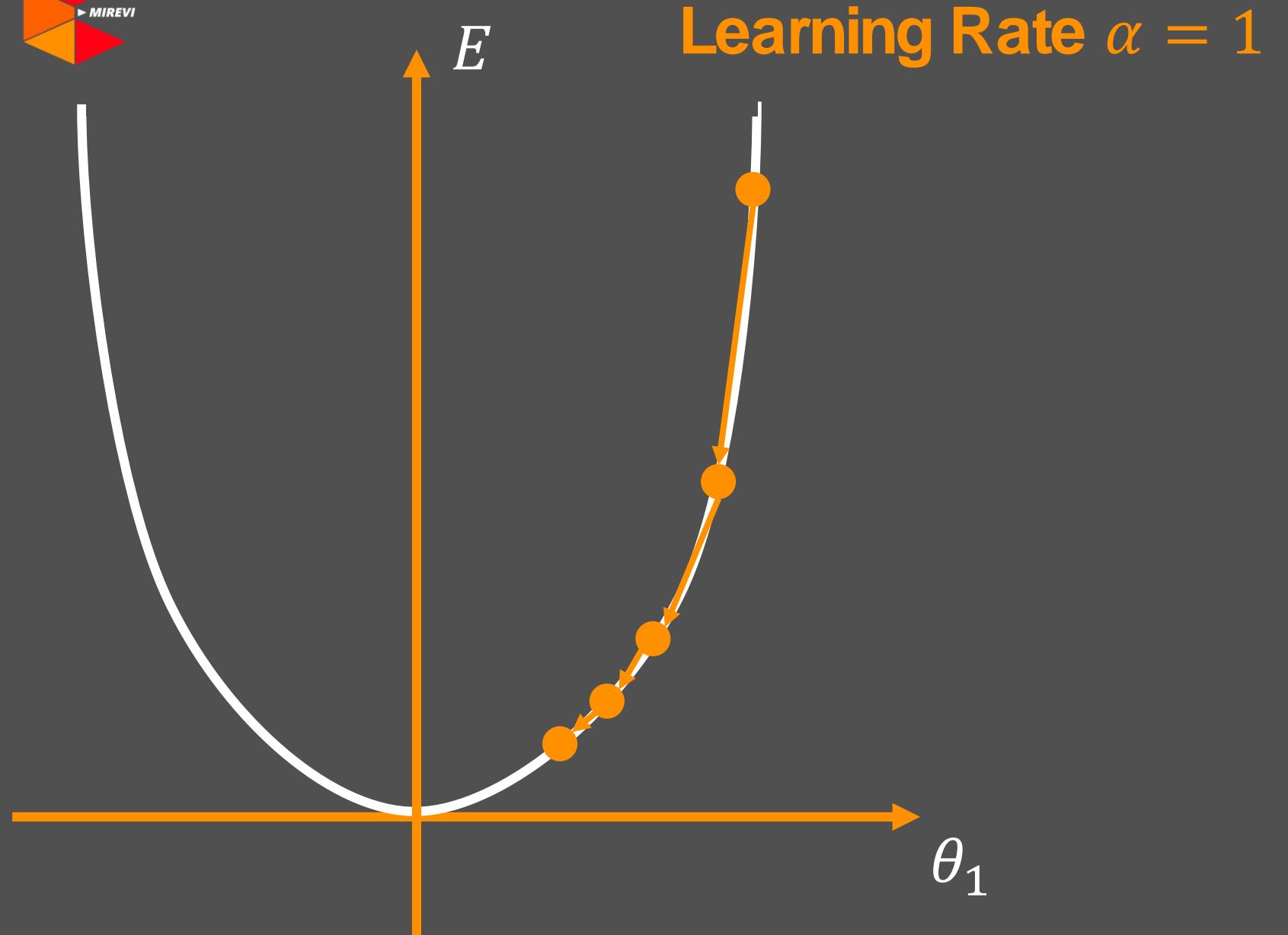
α : Learning Rate

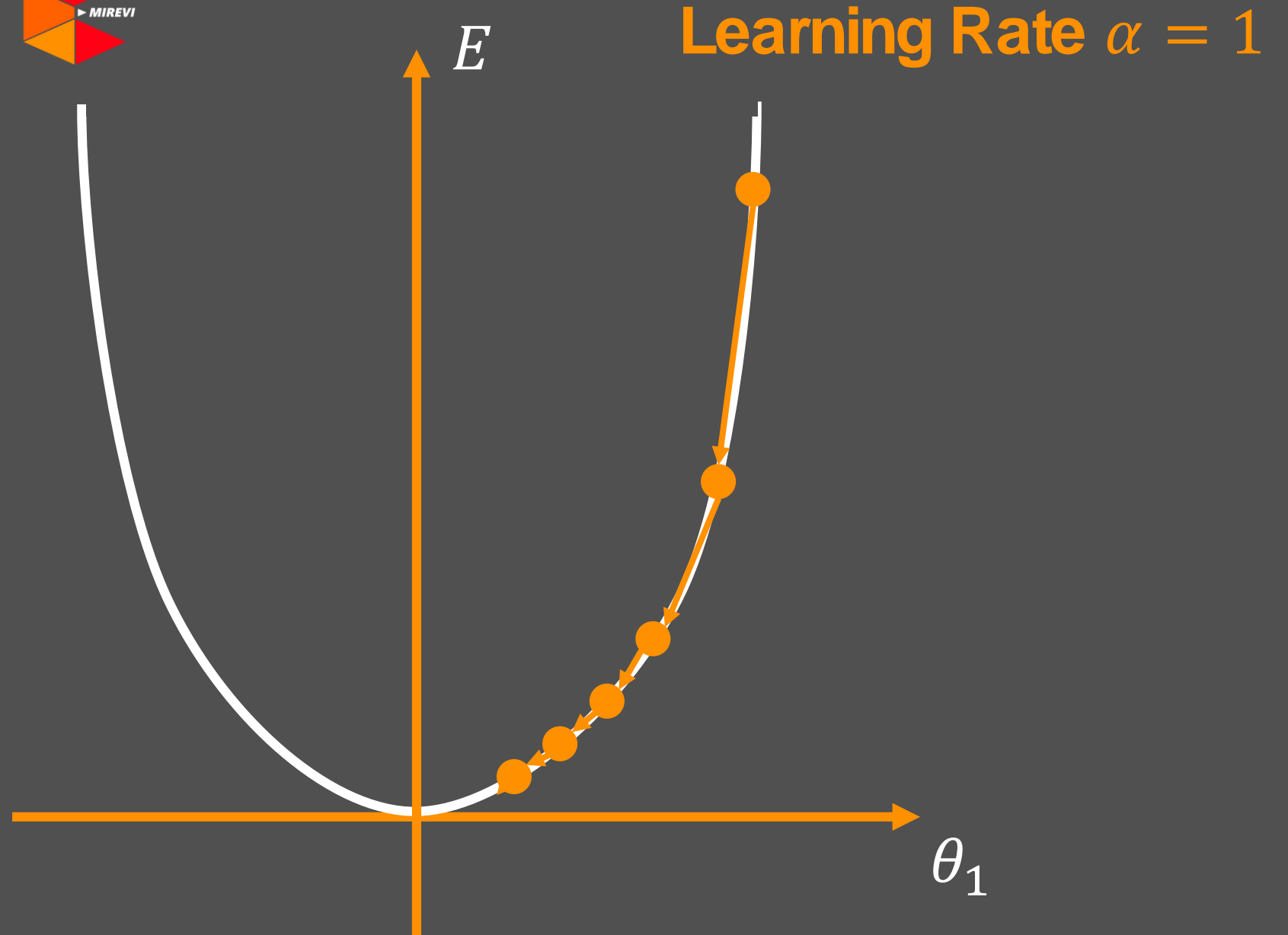
$$\alpha = 1$$





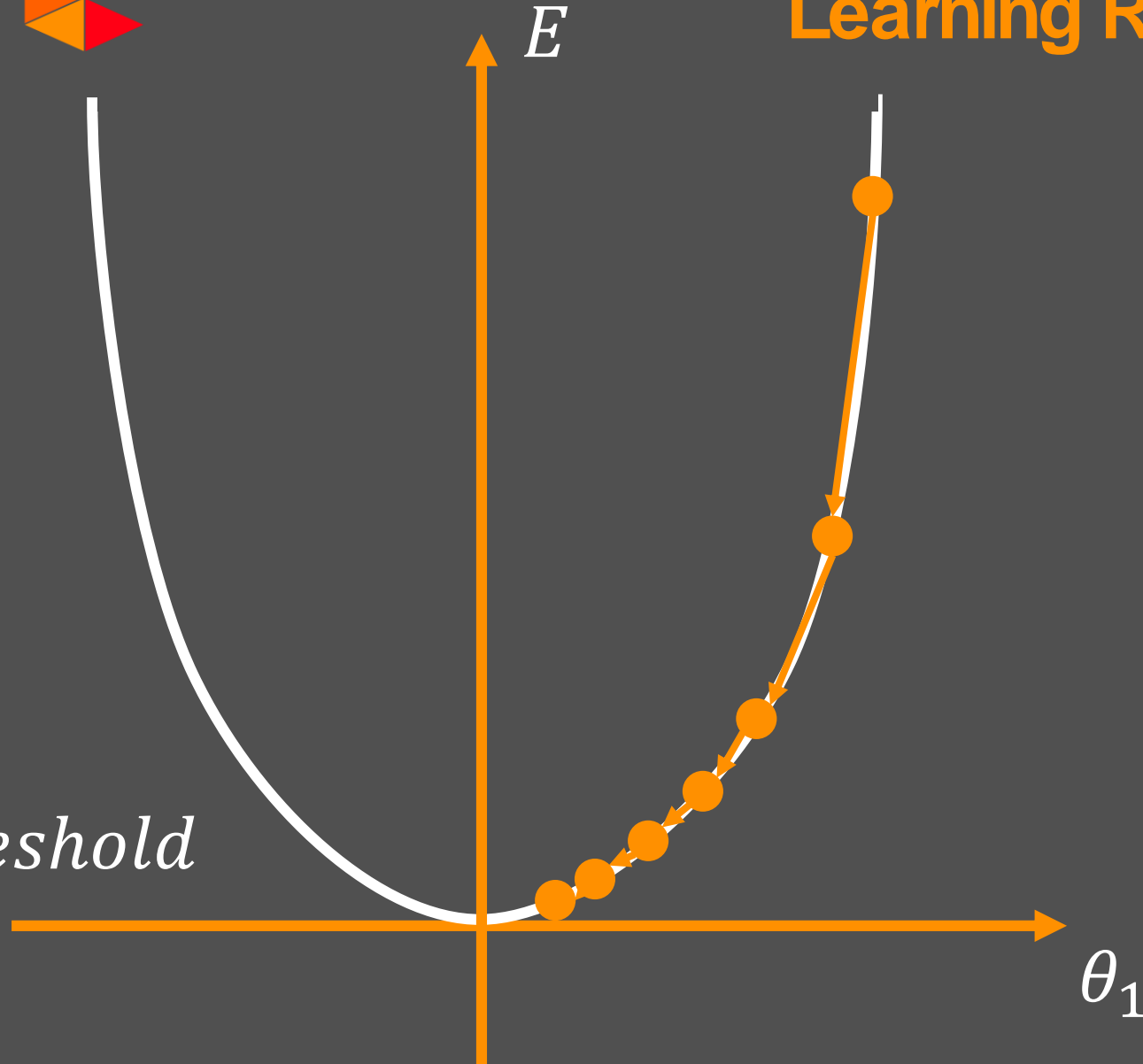




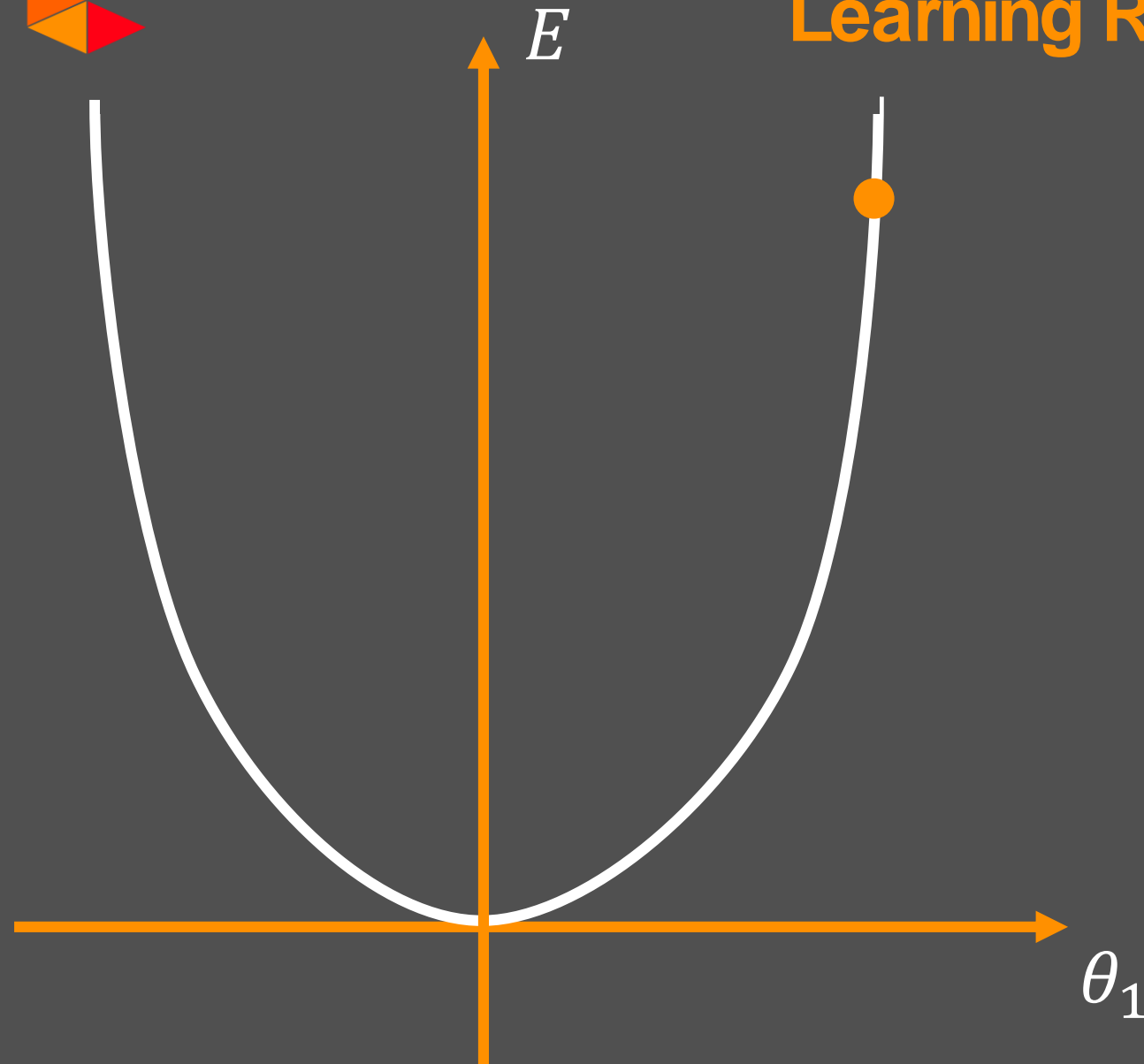


Learning Rate $\alpha = 1$

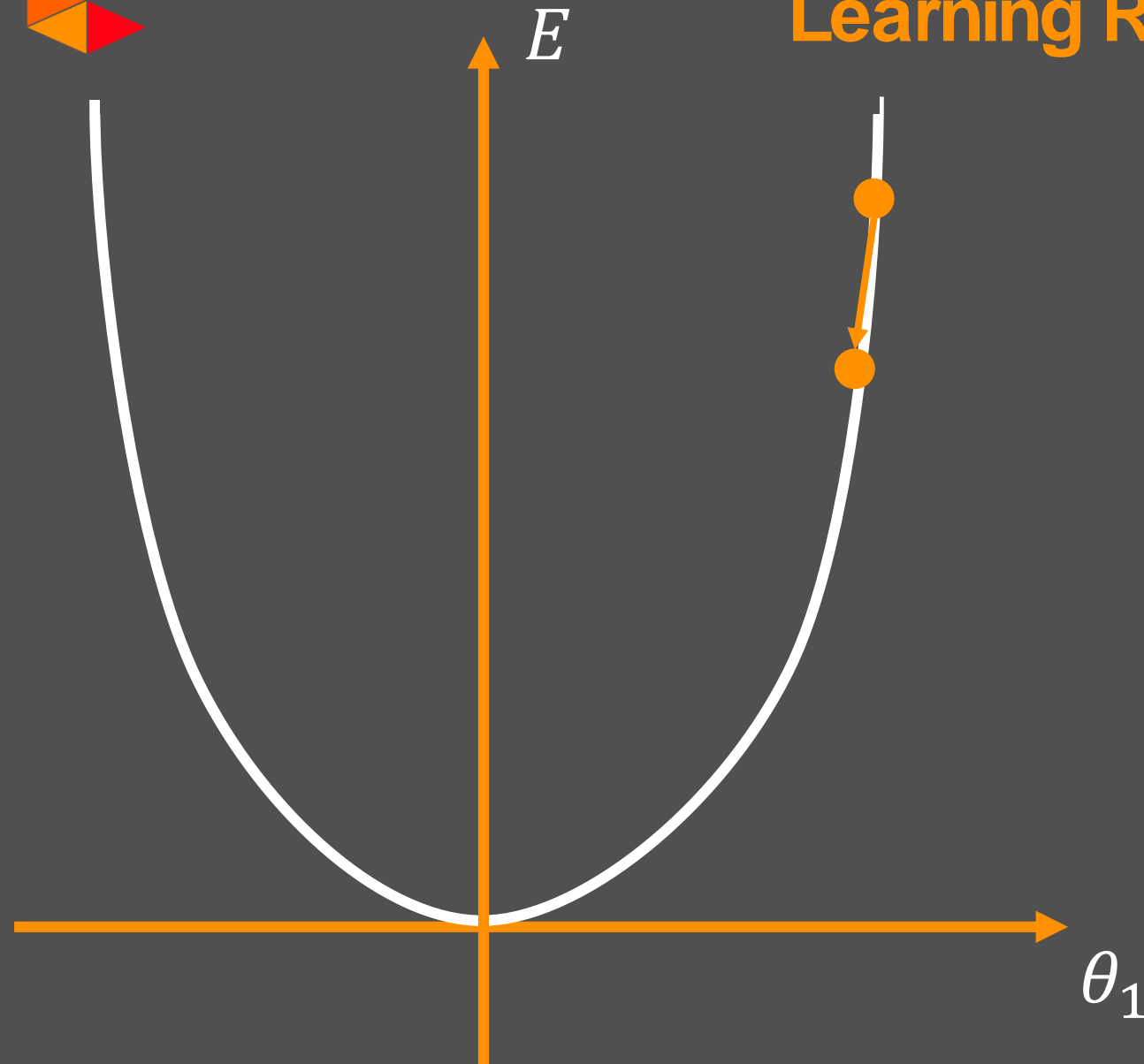
e.g.: $E < \text{threshold}$

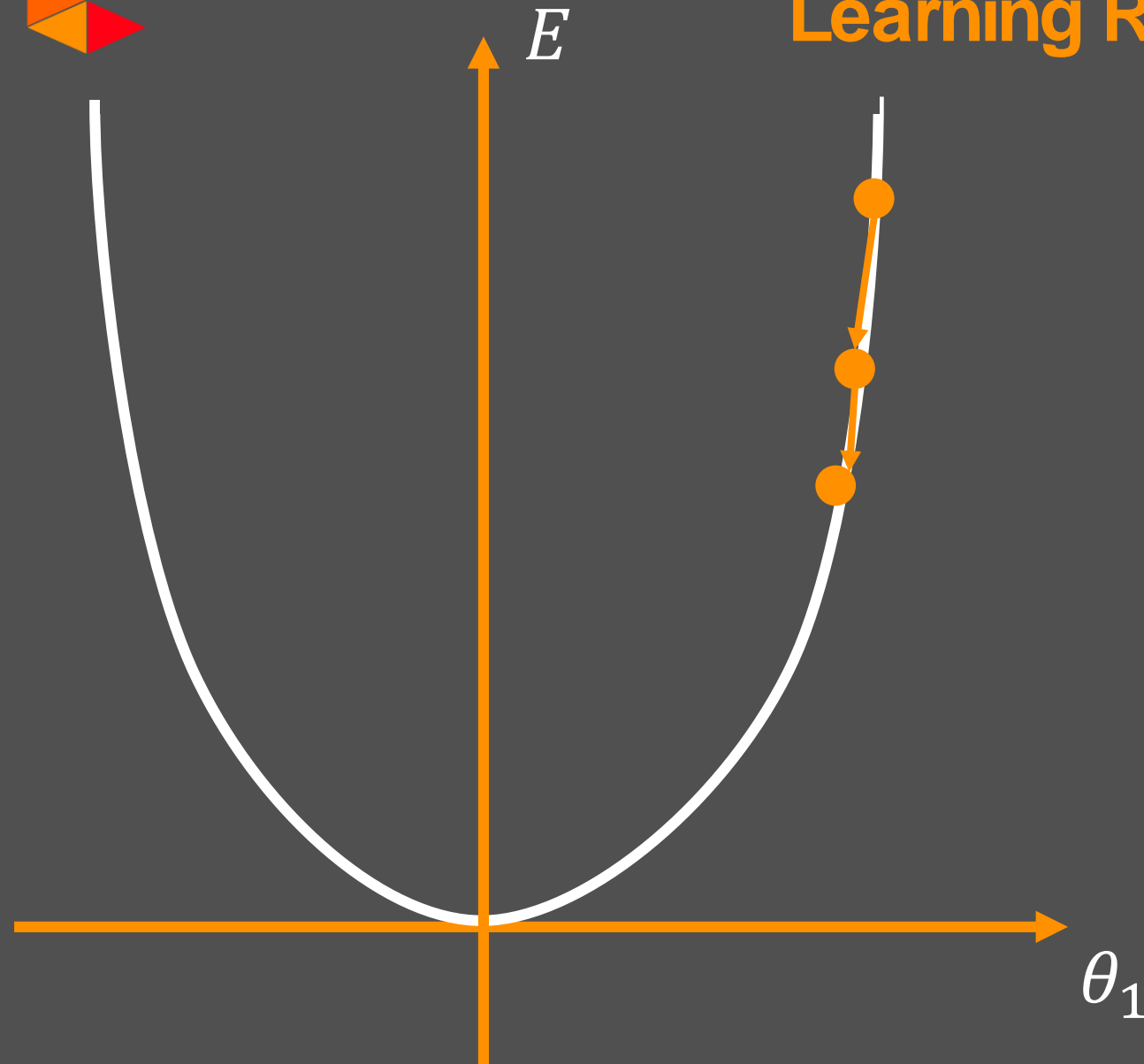


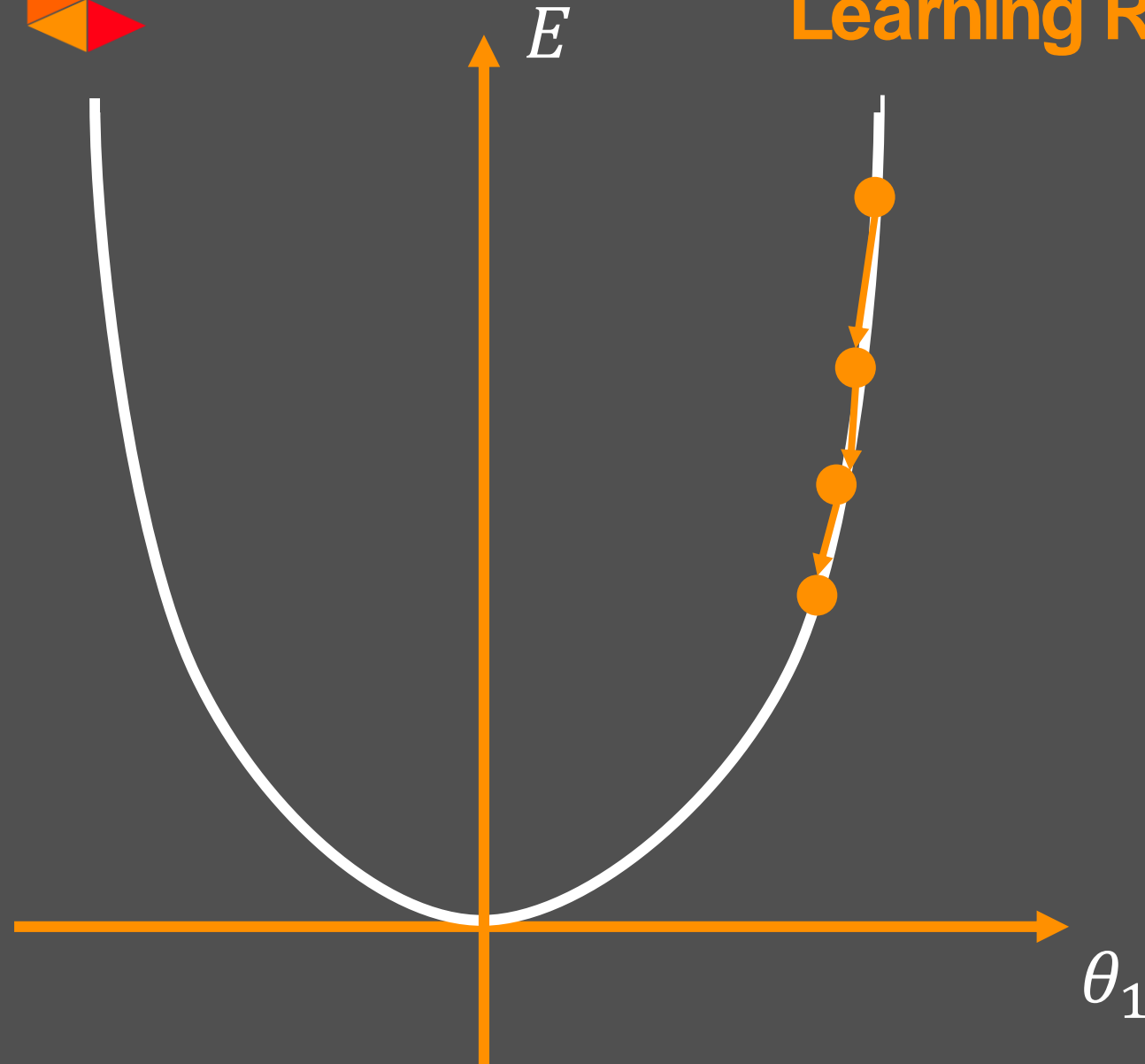
$$\alpha = 0.5$$

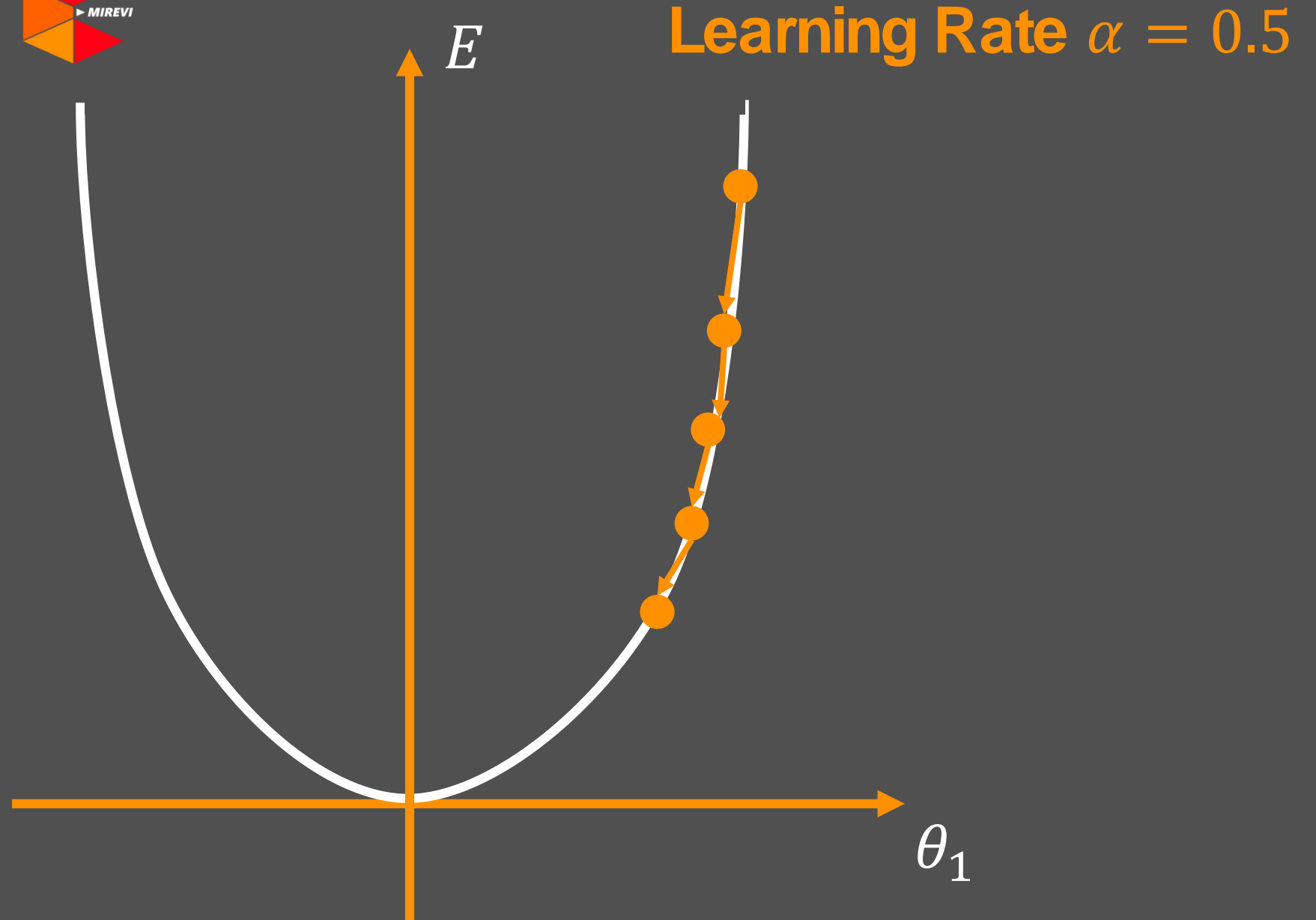


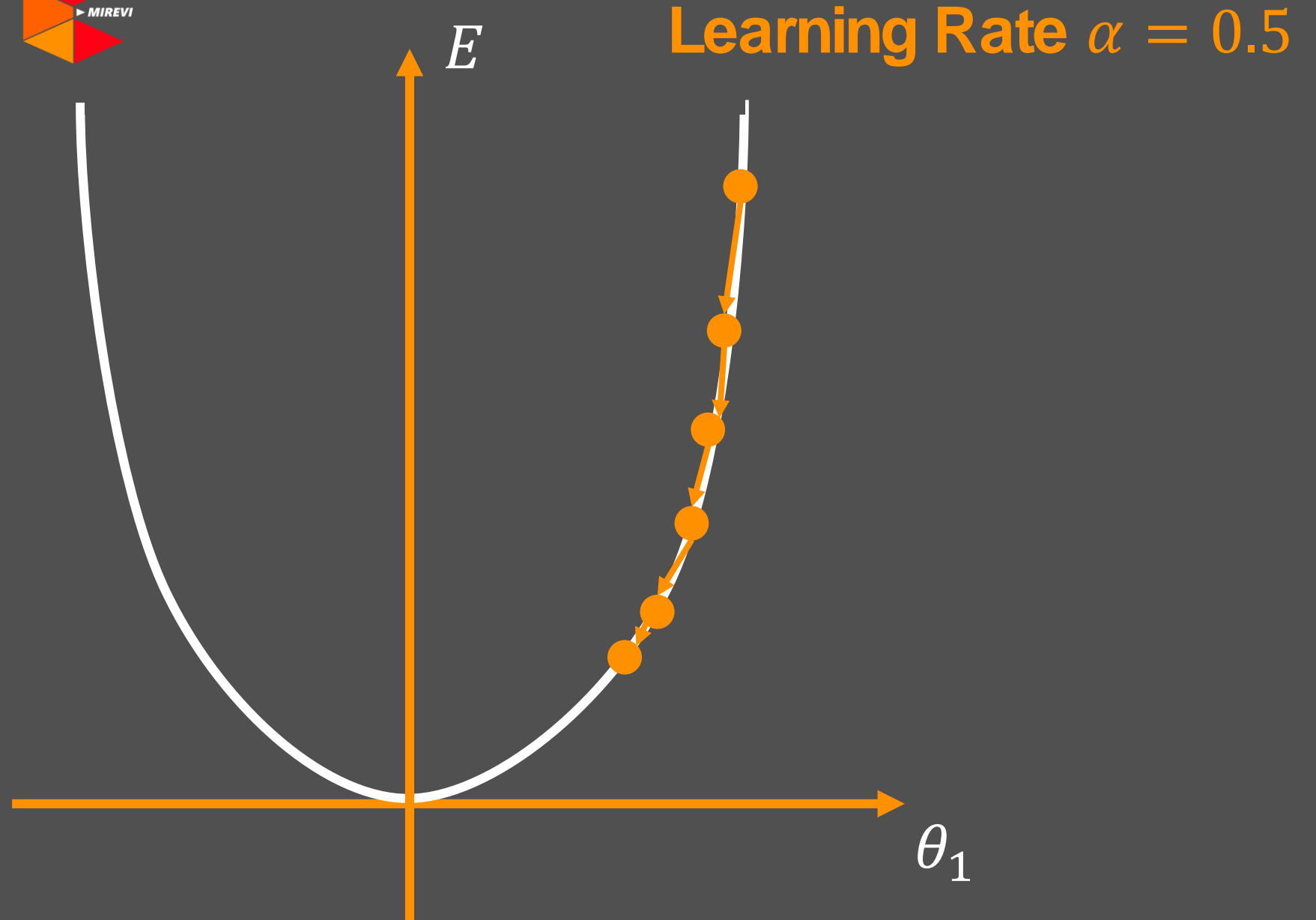
Learning Rate $\alpha = 0.5$

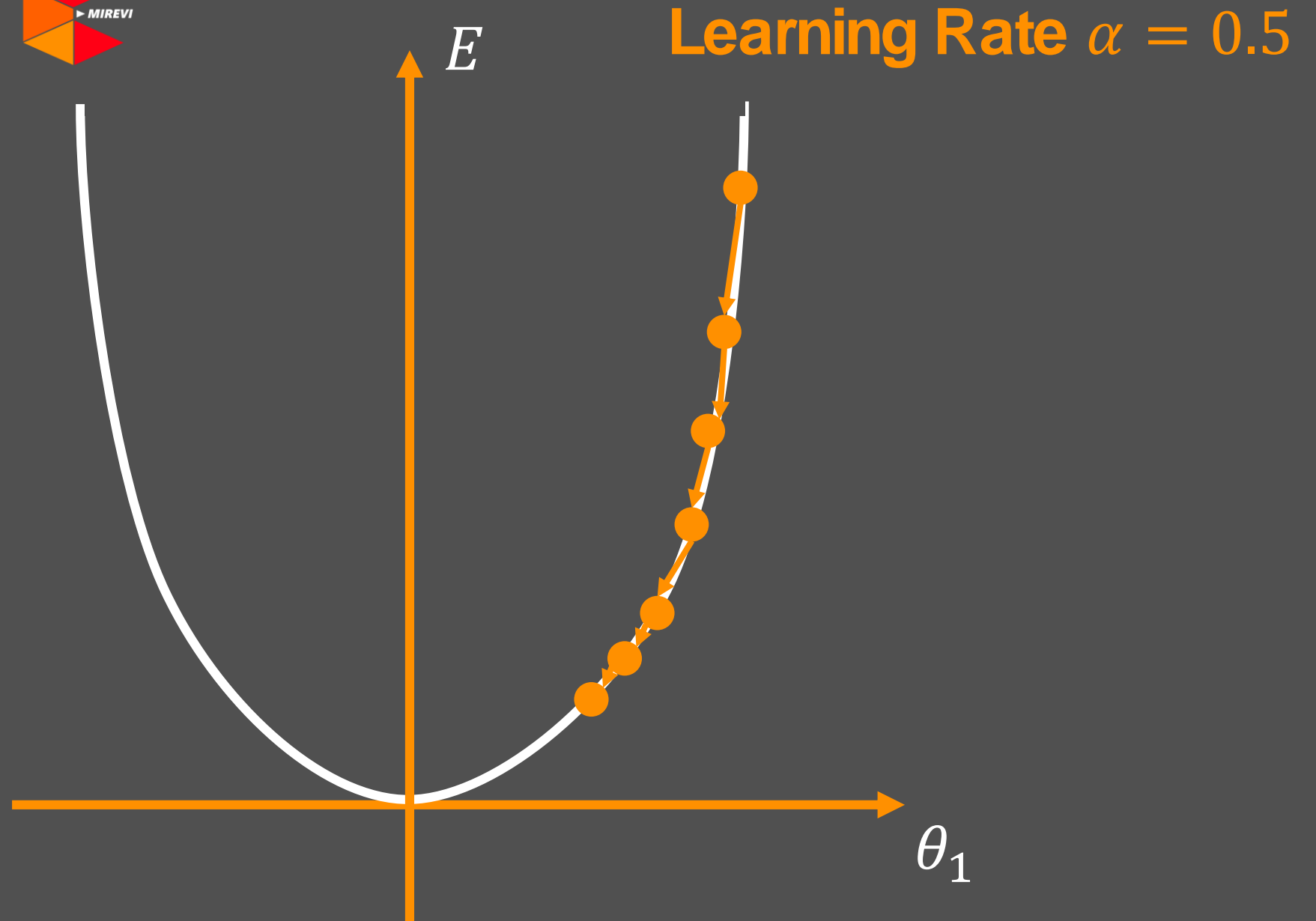


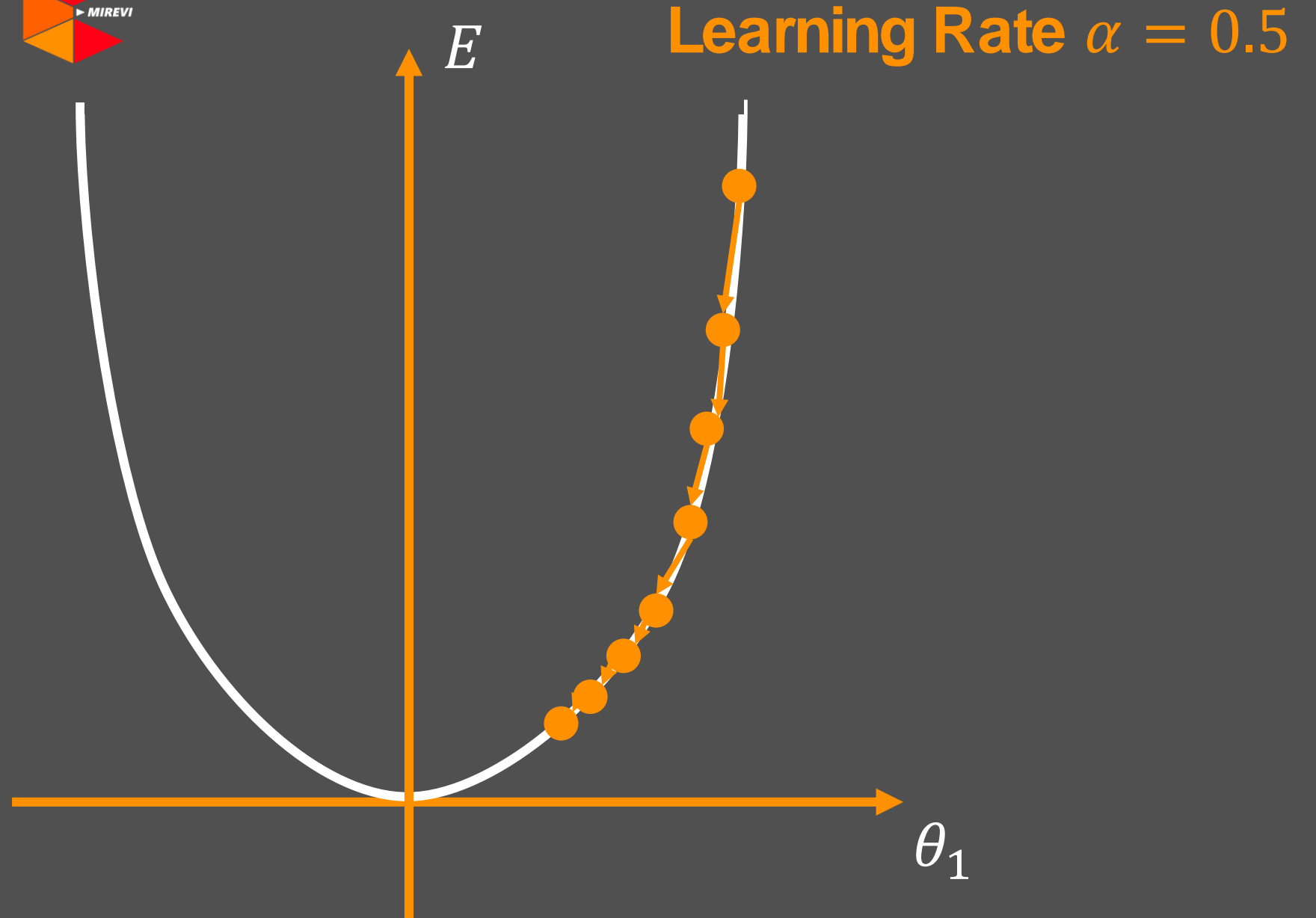


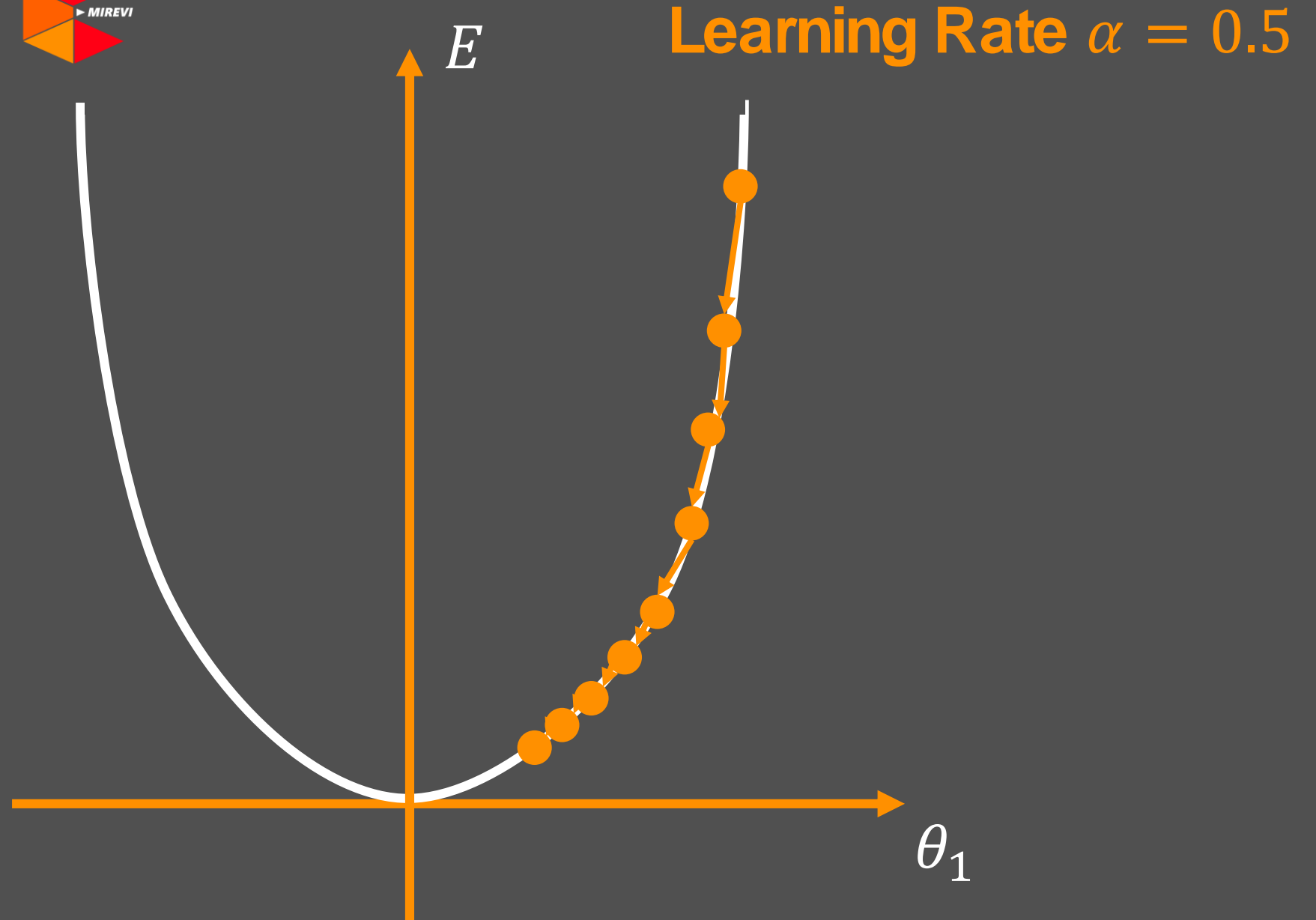


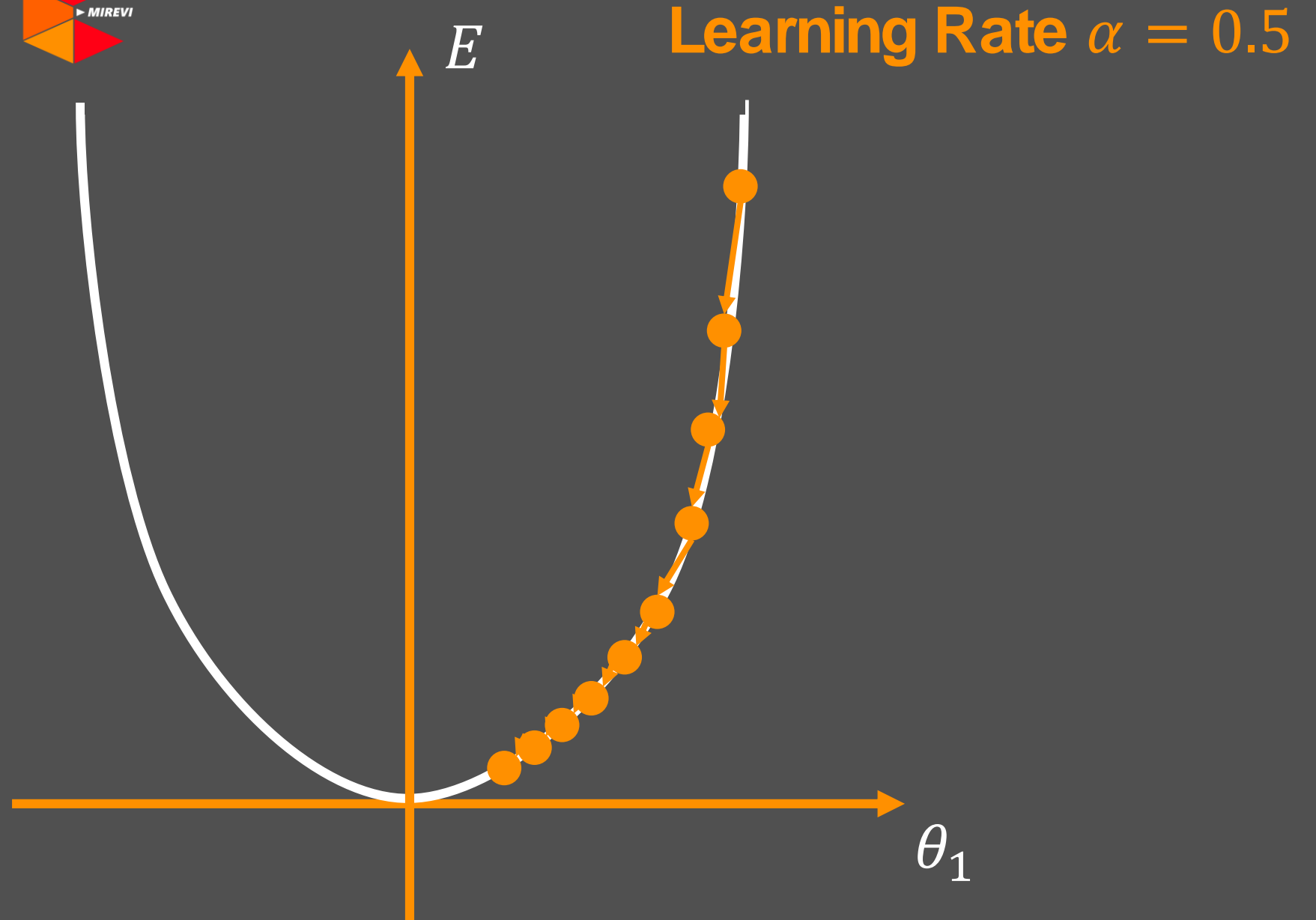




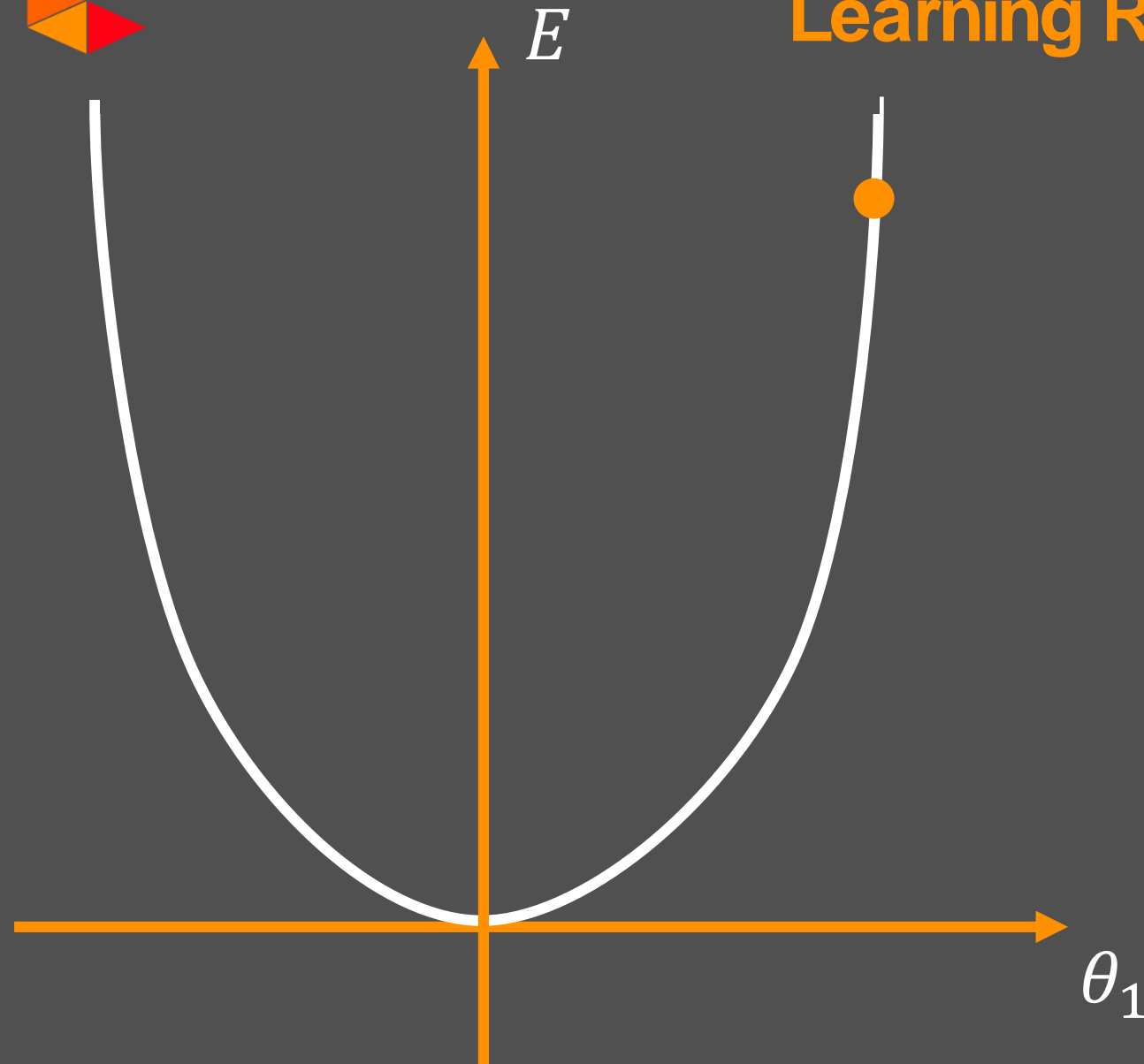




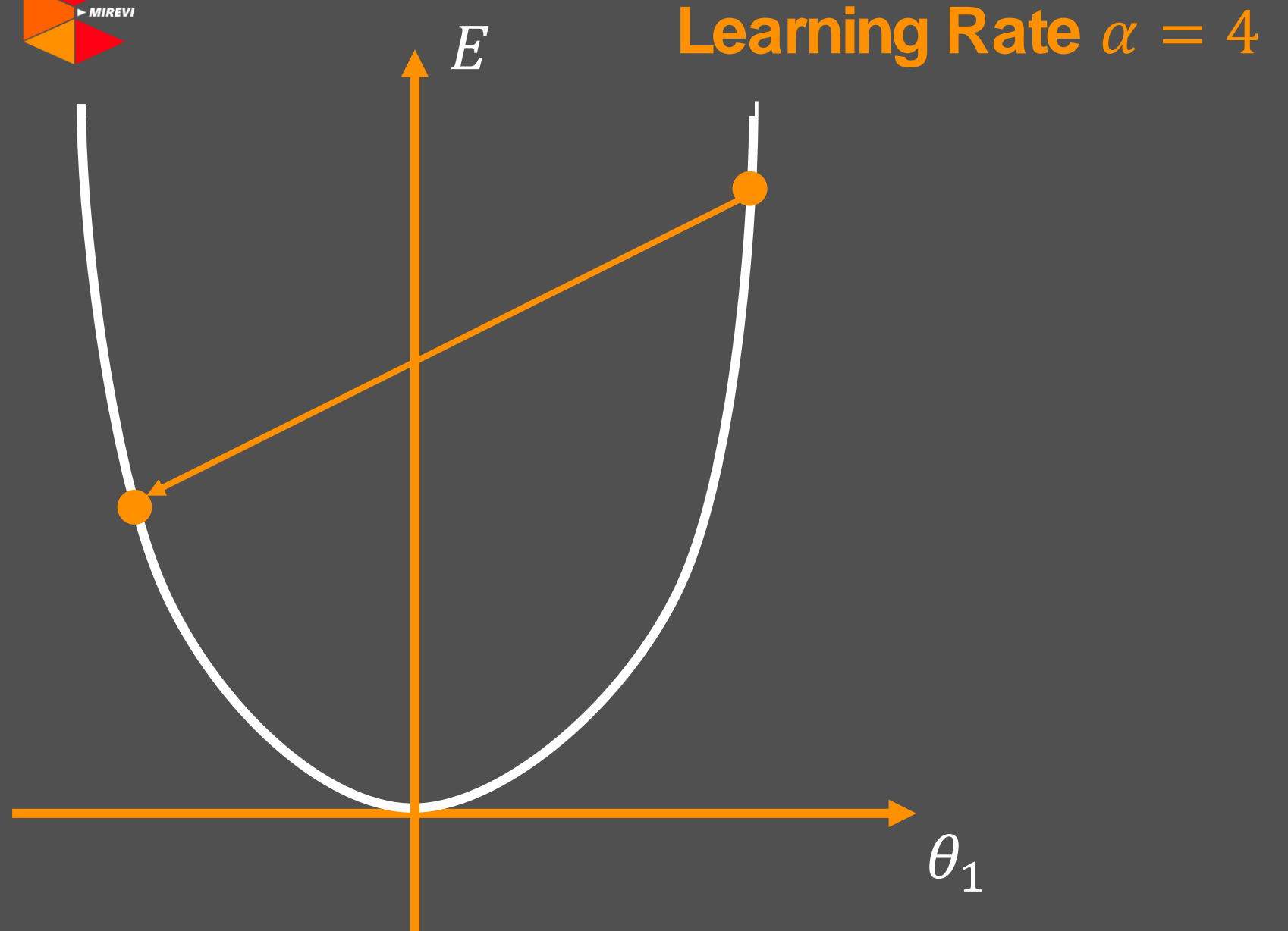


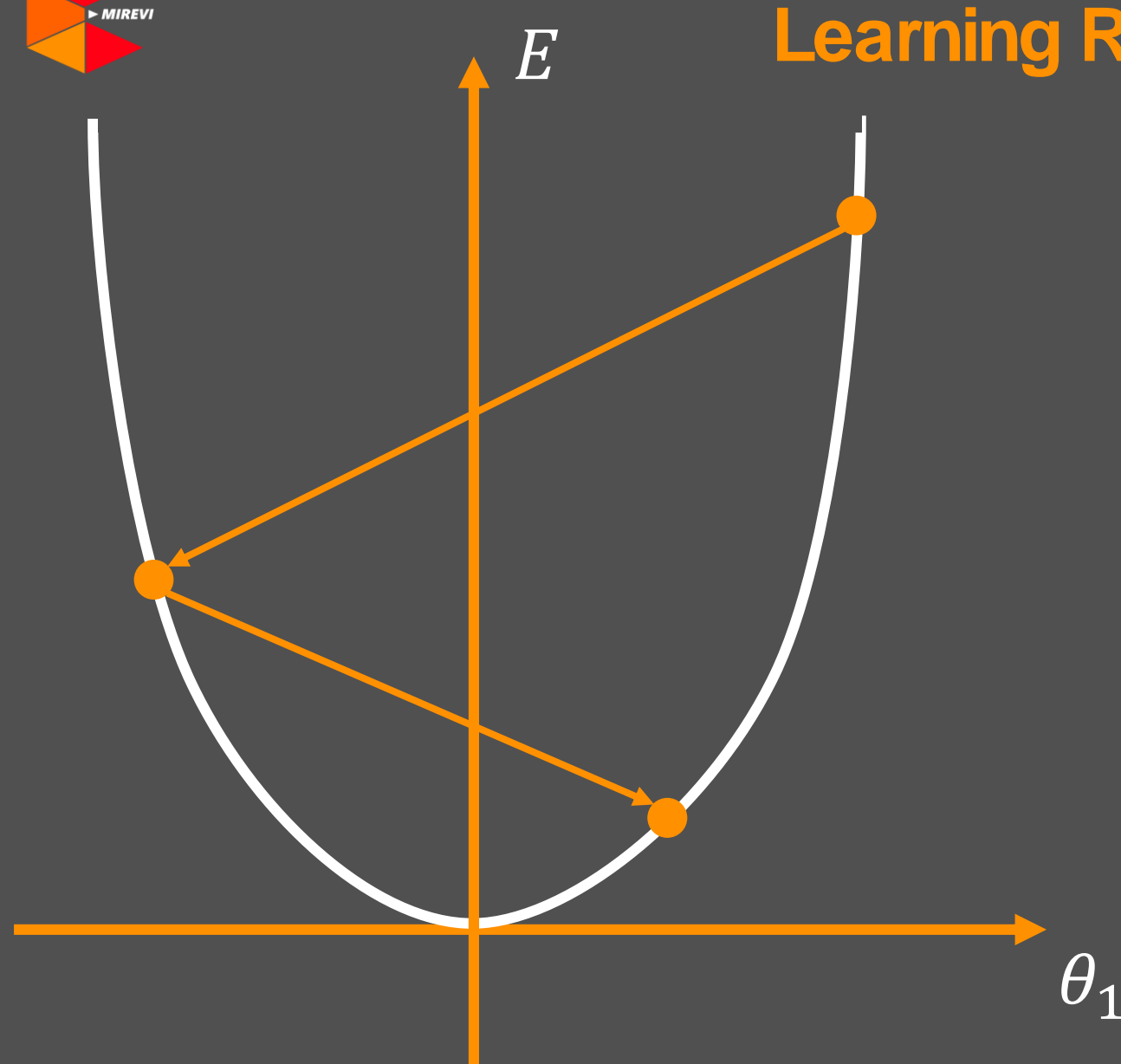


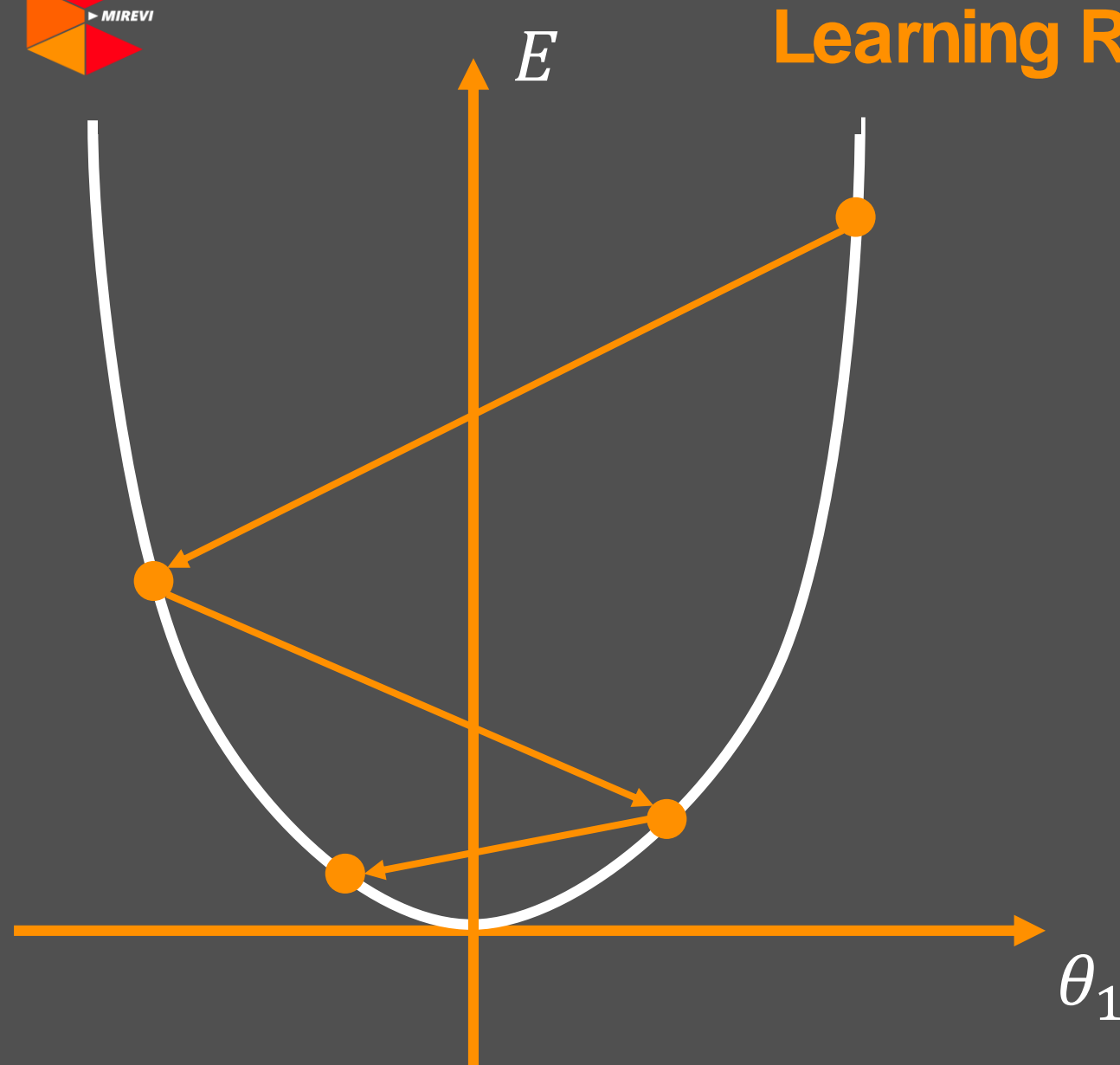
$$\alpha = 4$$

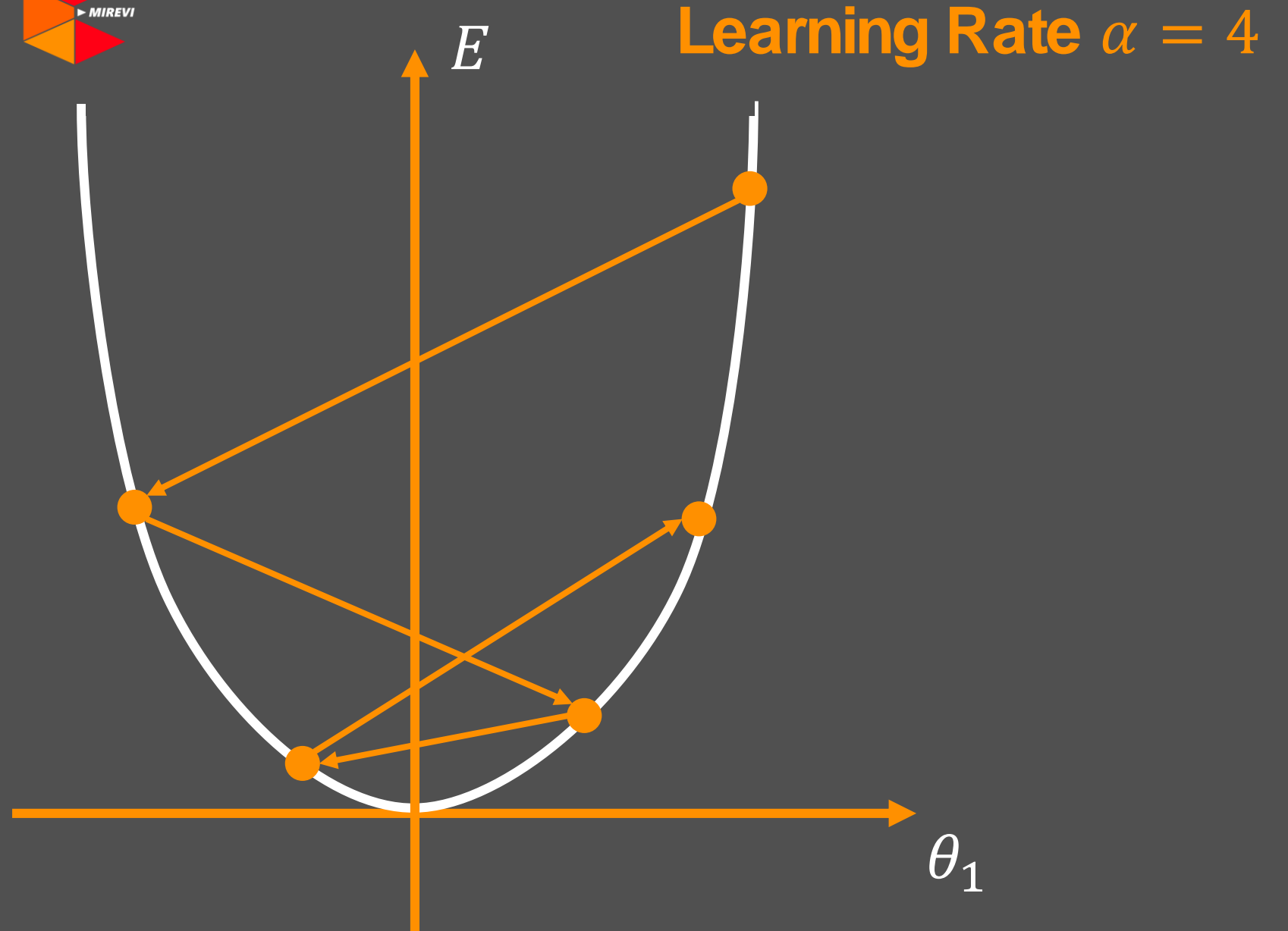


Learning Rate $\alpha = 4$

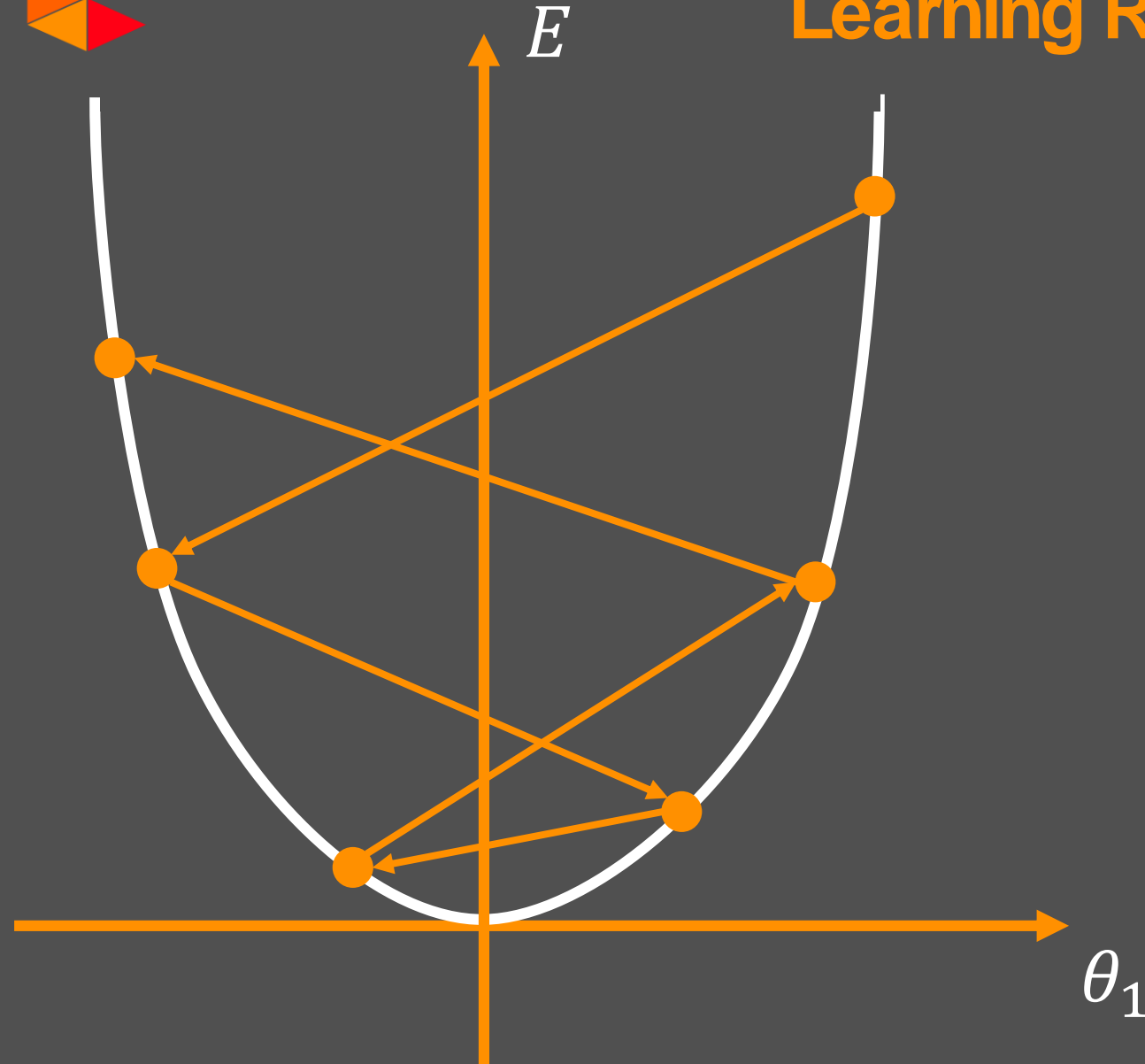


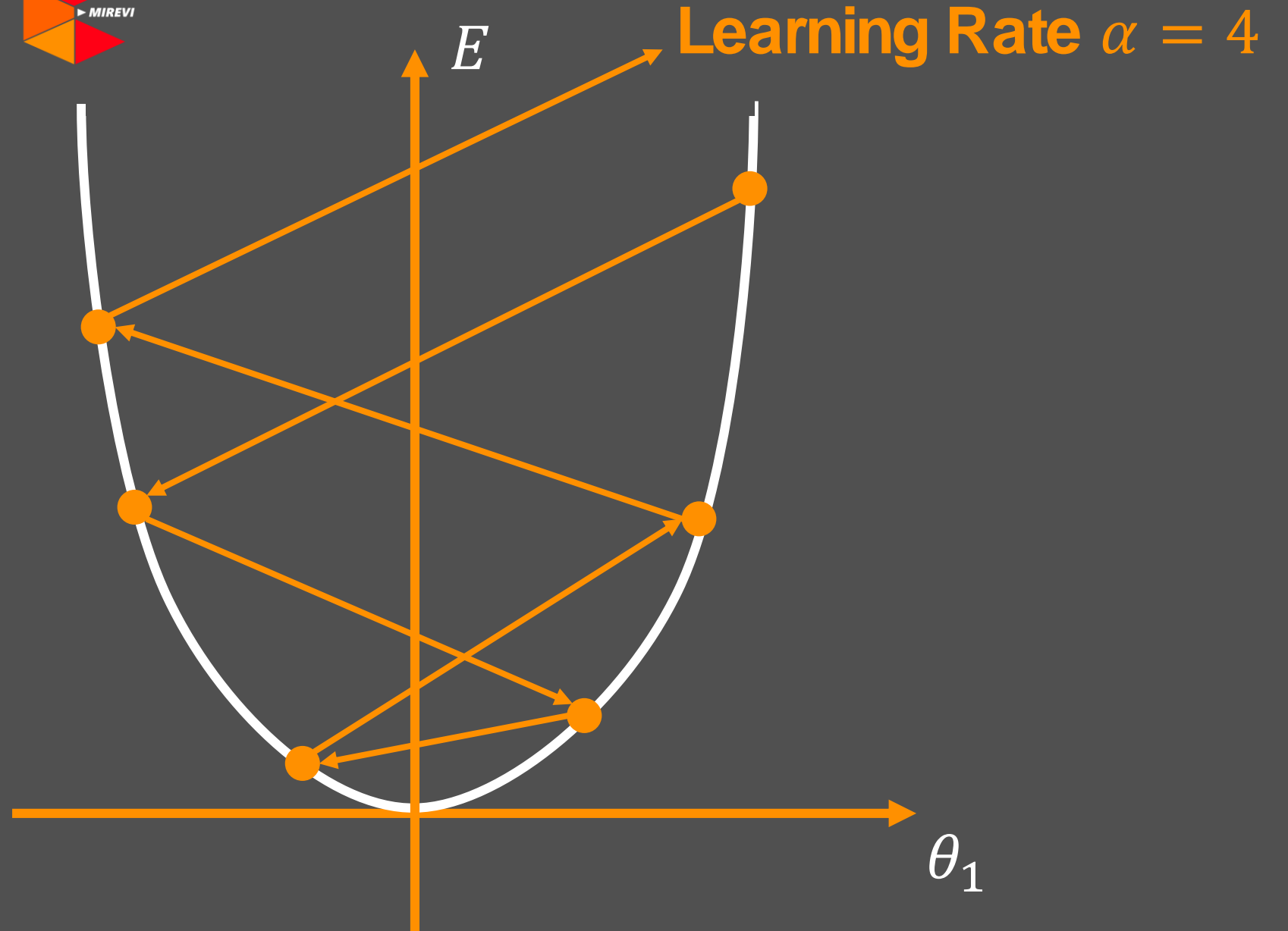




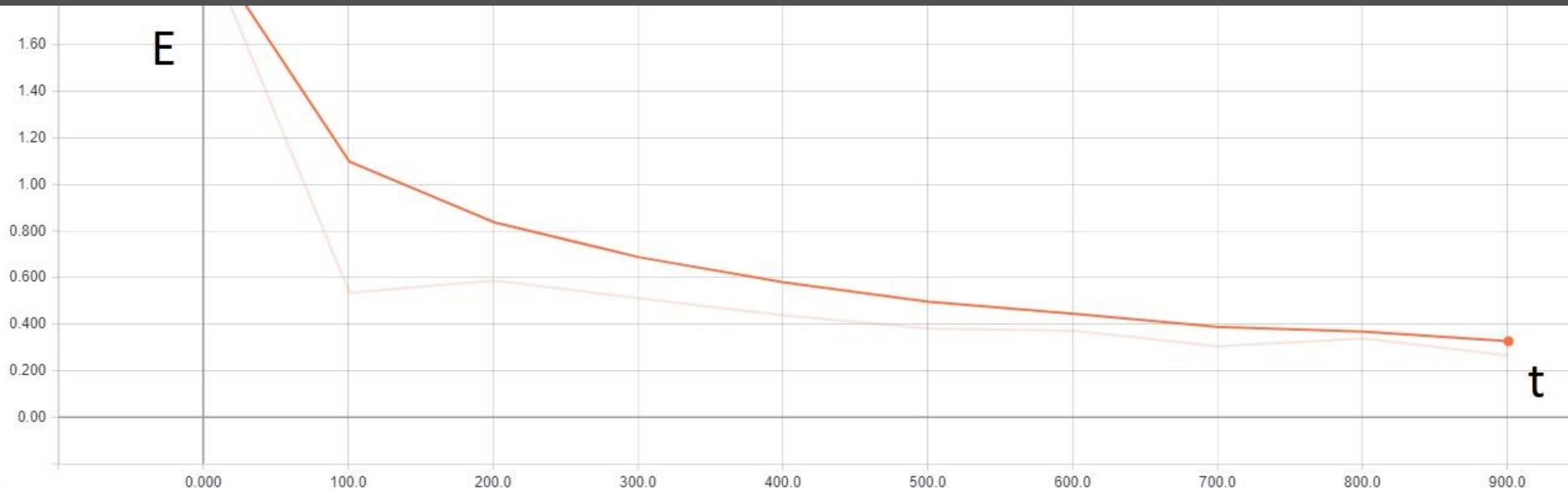


Learning Rate $\alpha = 4$

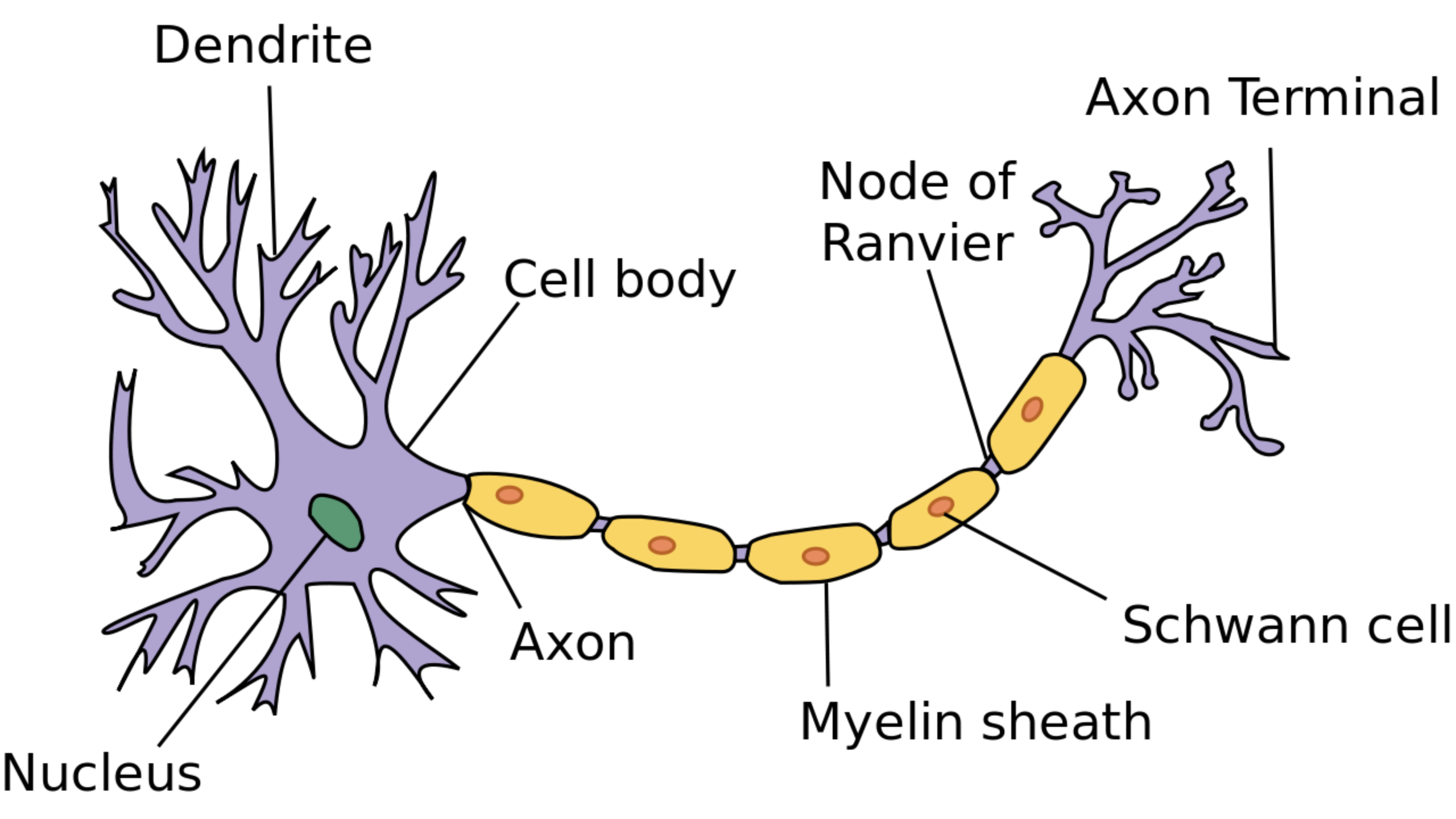


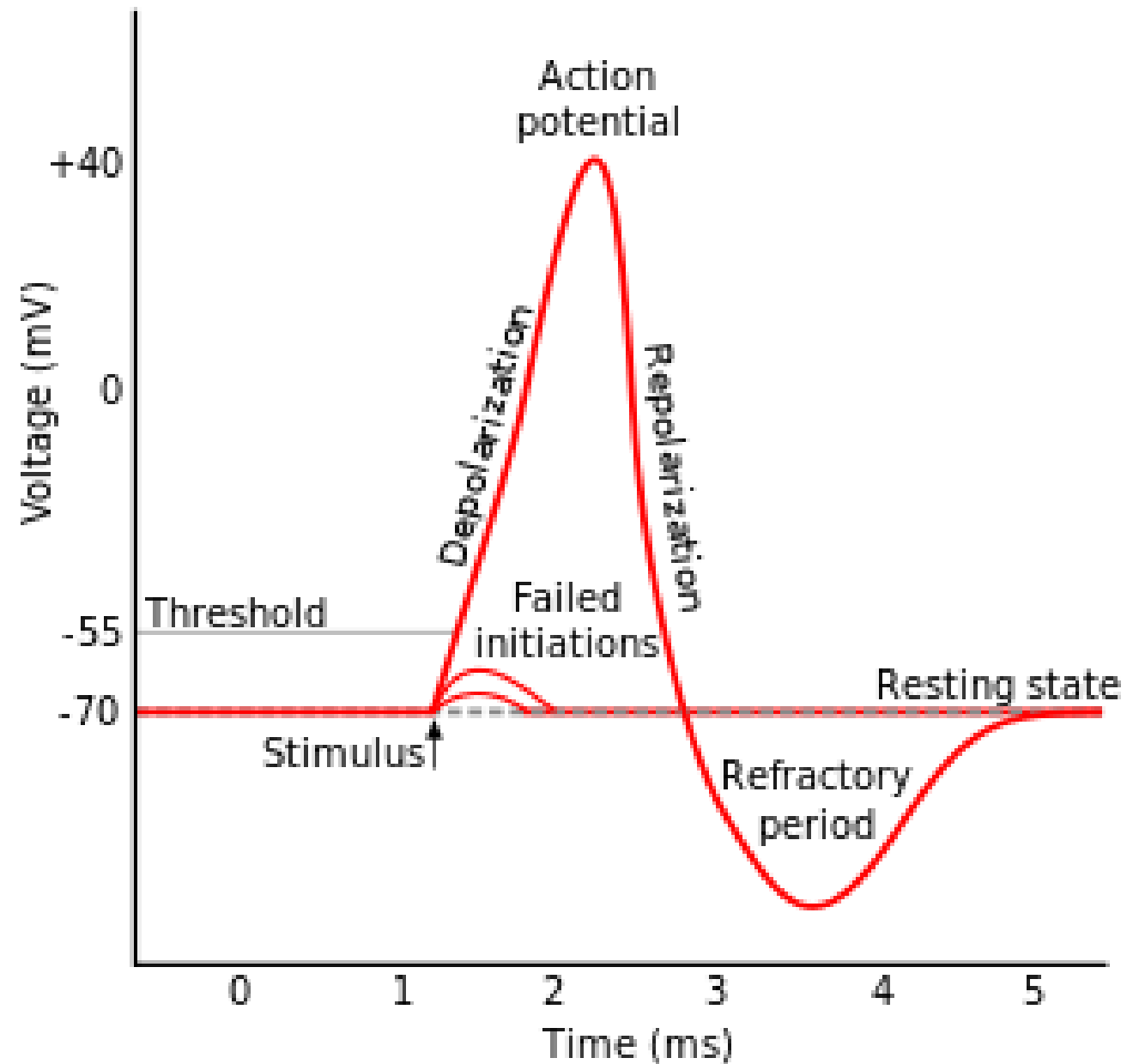


- Choose good learning rate α by testing
- e.g. $\alpha = i, i \in \{0.01, 0.03, 0.09, 0.1, 0.3, 0.9, 1, 3, 9\}$



Towards Neural Nets





Perceptron

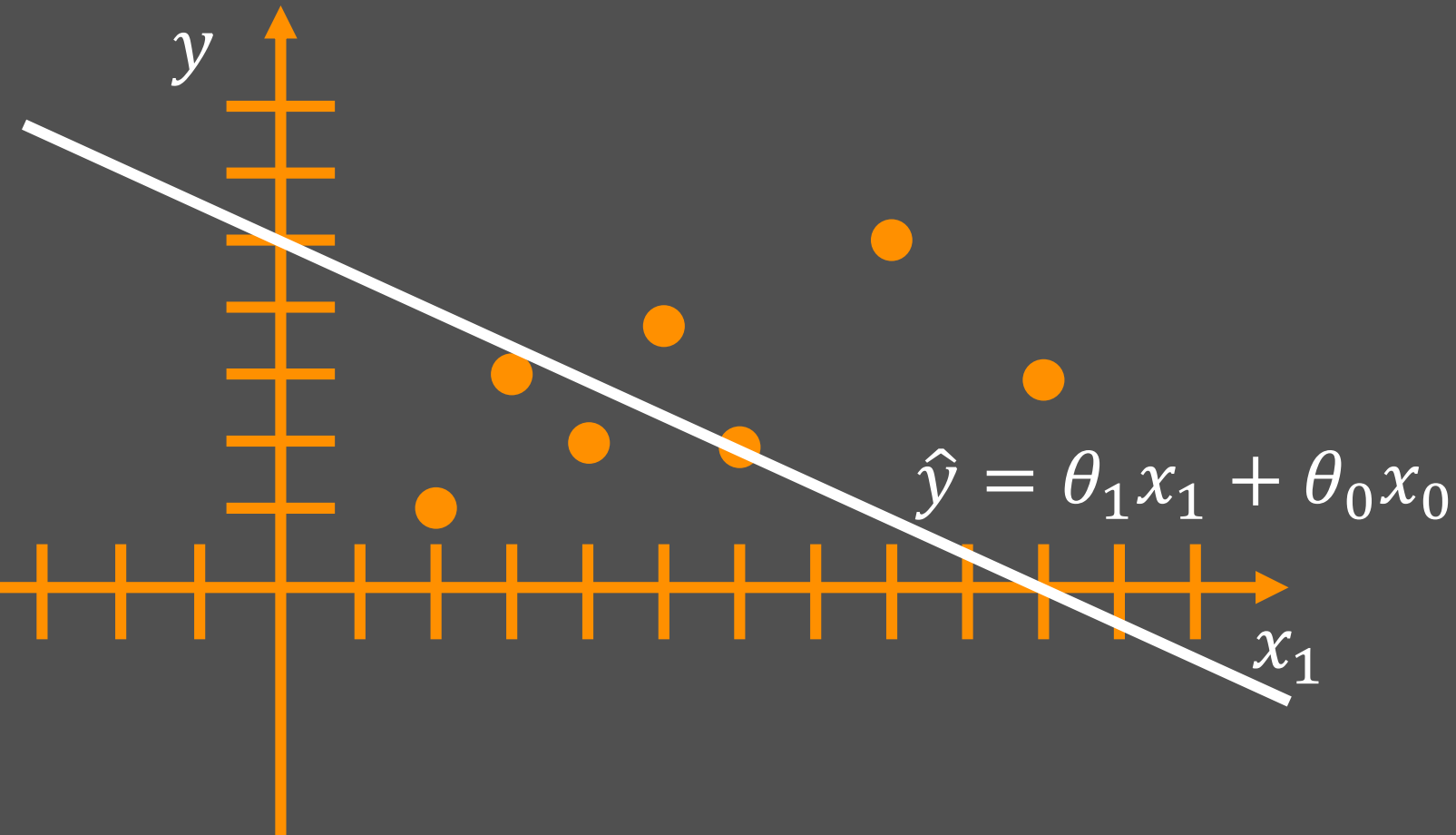
- **Binary Classification**
- **Element of Neural Net**
- **Weighted Sum**
- **Activation Function**
- **Decision Boundary**

$$y=f(x)$$

Elements of Perceptron

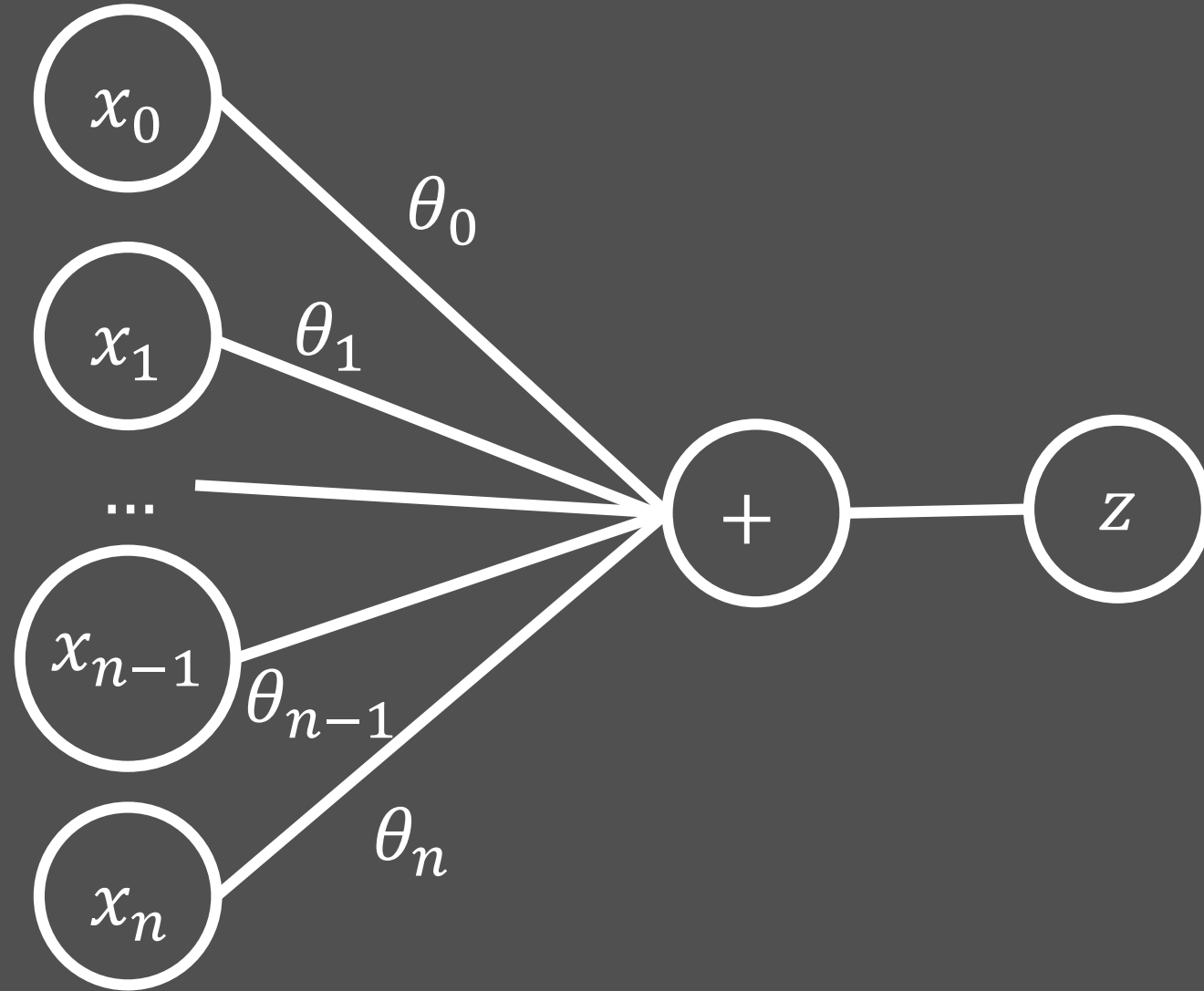
- **Activation Function** $y = a(z)$
- **Weighted Sum** $z = \theta x$

Weighted Sum:



Weighted Sum $z = \theta x$

- $z = \theta x = \theta_n * x_n + \theta_{n-1} * x_{n-1} + \dots + \theta_1 * x_1 + \theta_0 * x_0$
- θ : Weights
- x : Data
- $x_0 = 1$

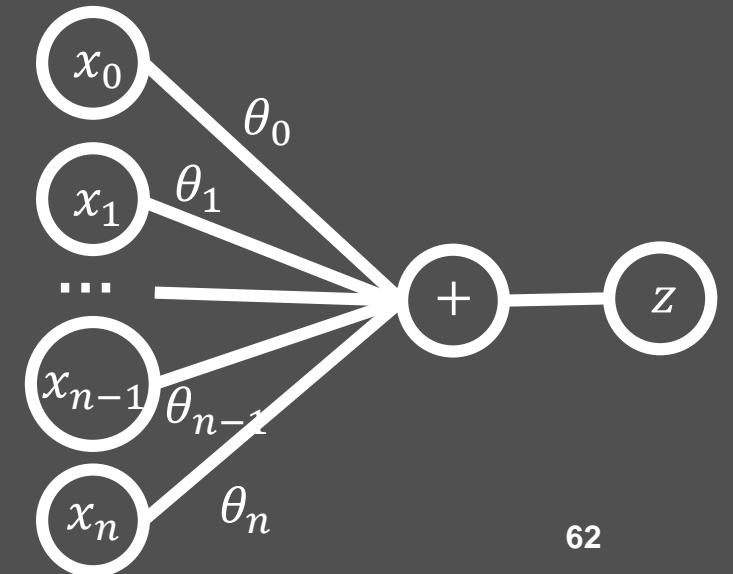
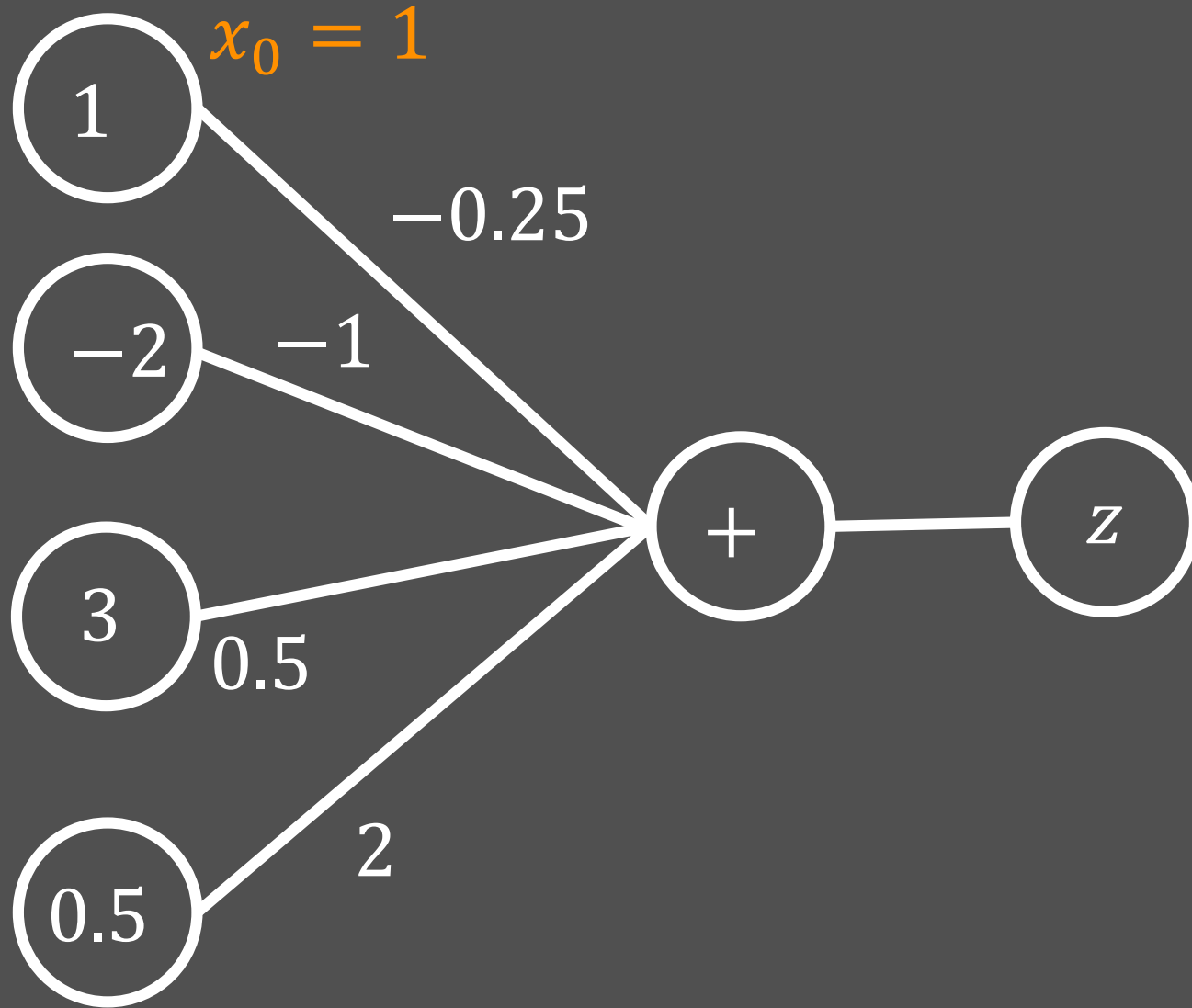


Weighted Sum

- $z = \theta x$
- θ : **Weights**
- x : **Data**
- $x_0 = 1$

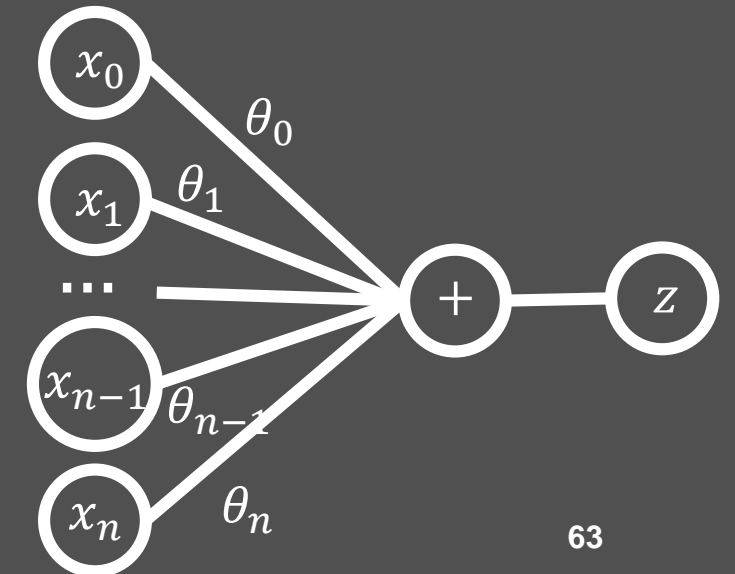
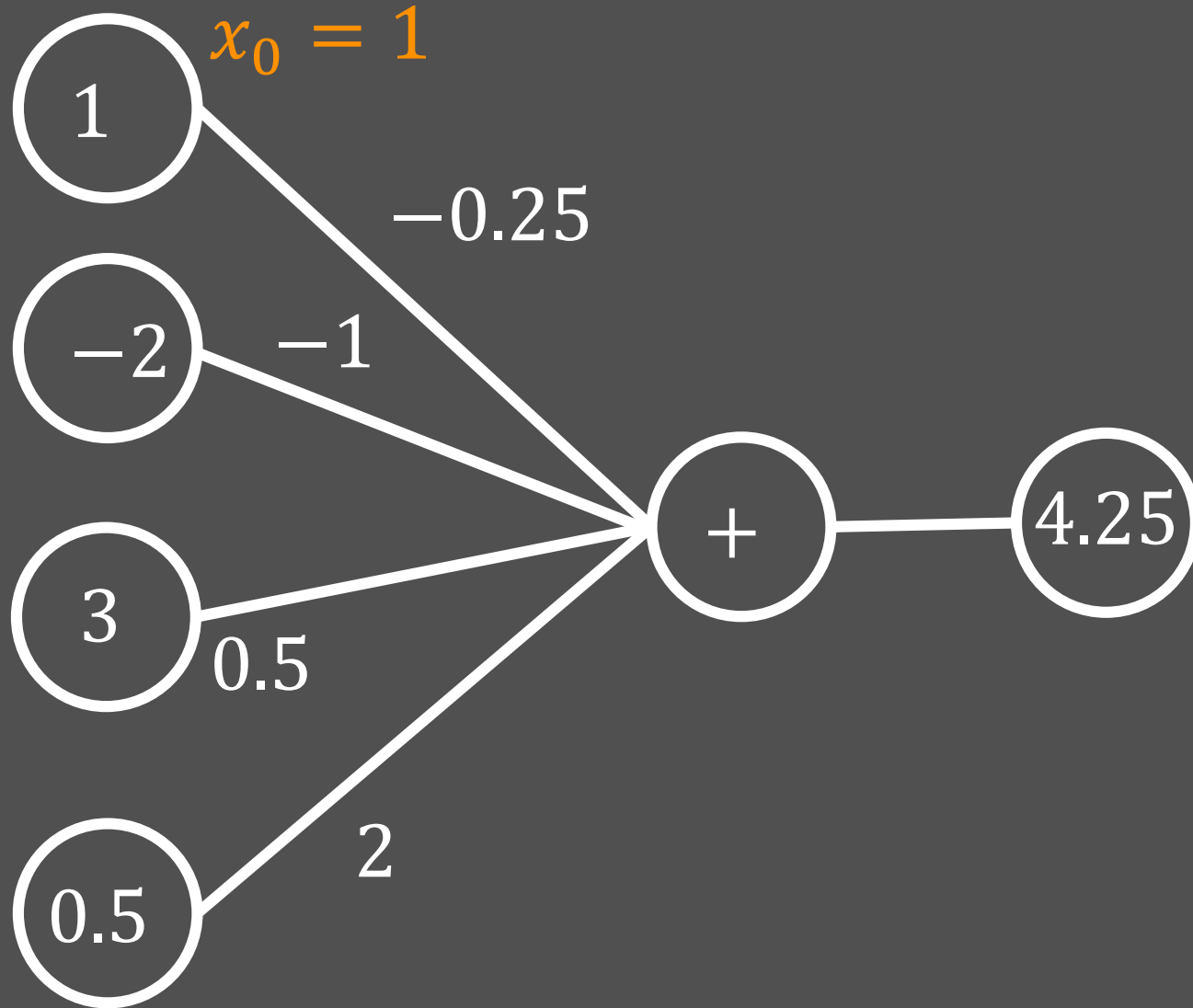
Weighted Sum

- $z = \theta x$
- θ : **Weights**
- x : **Data**
- $x_0 = 1$



Weighted Sum

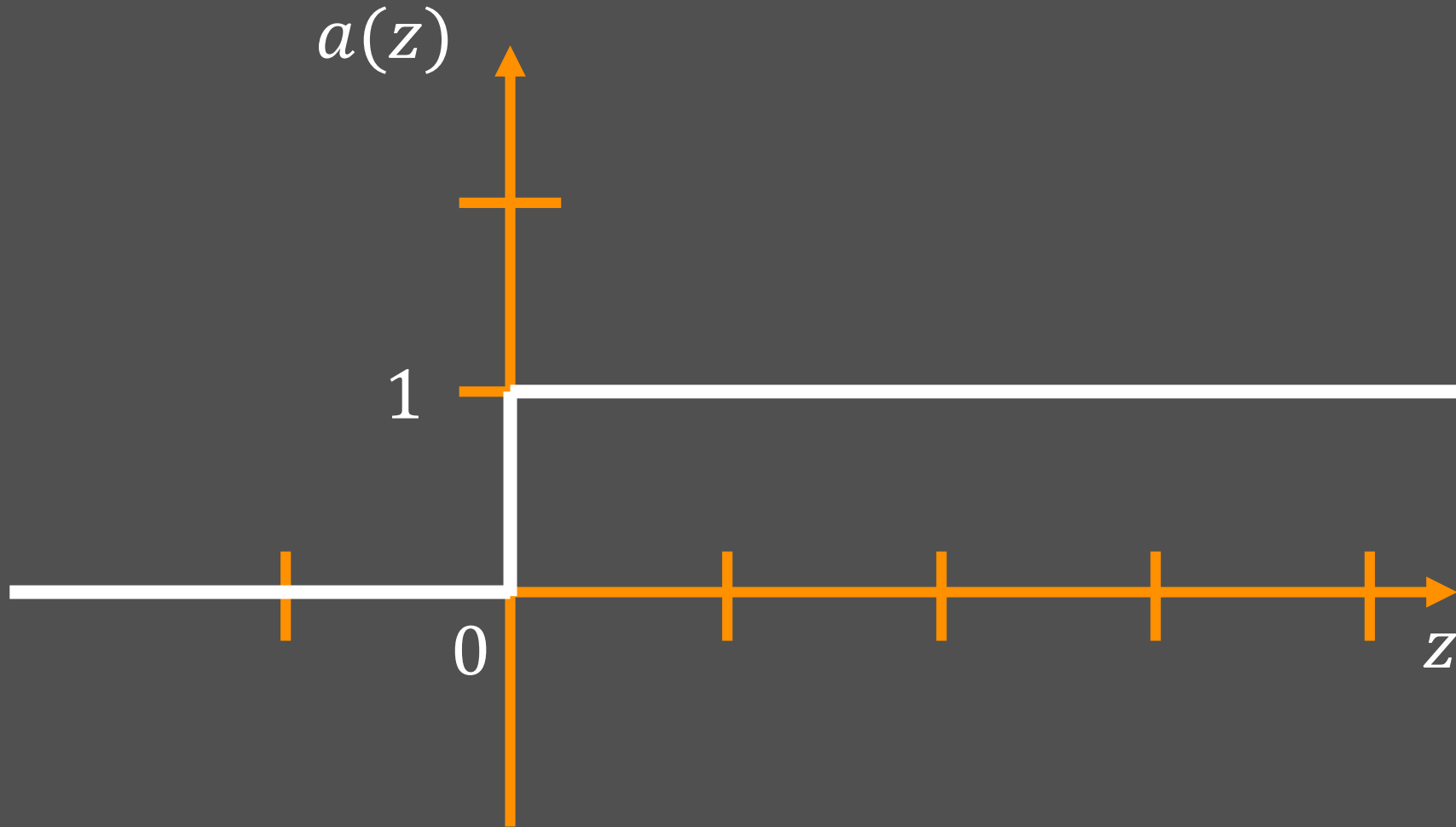
- $z = \theta x$
- θ : **Weights**
- x : **Data**
- $x_0 = 1$



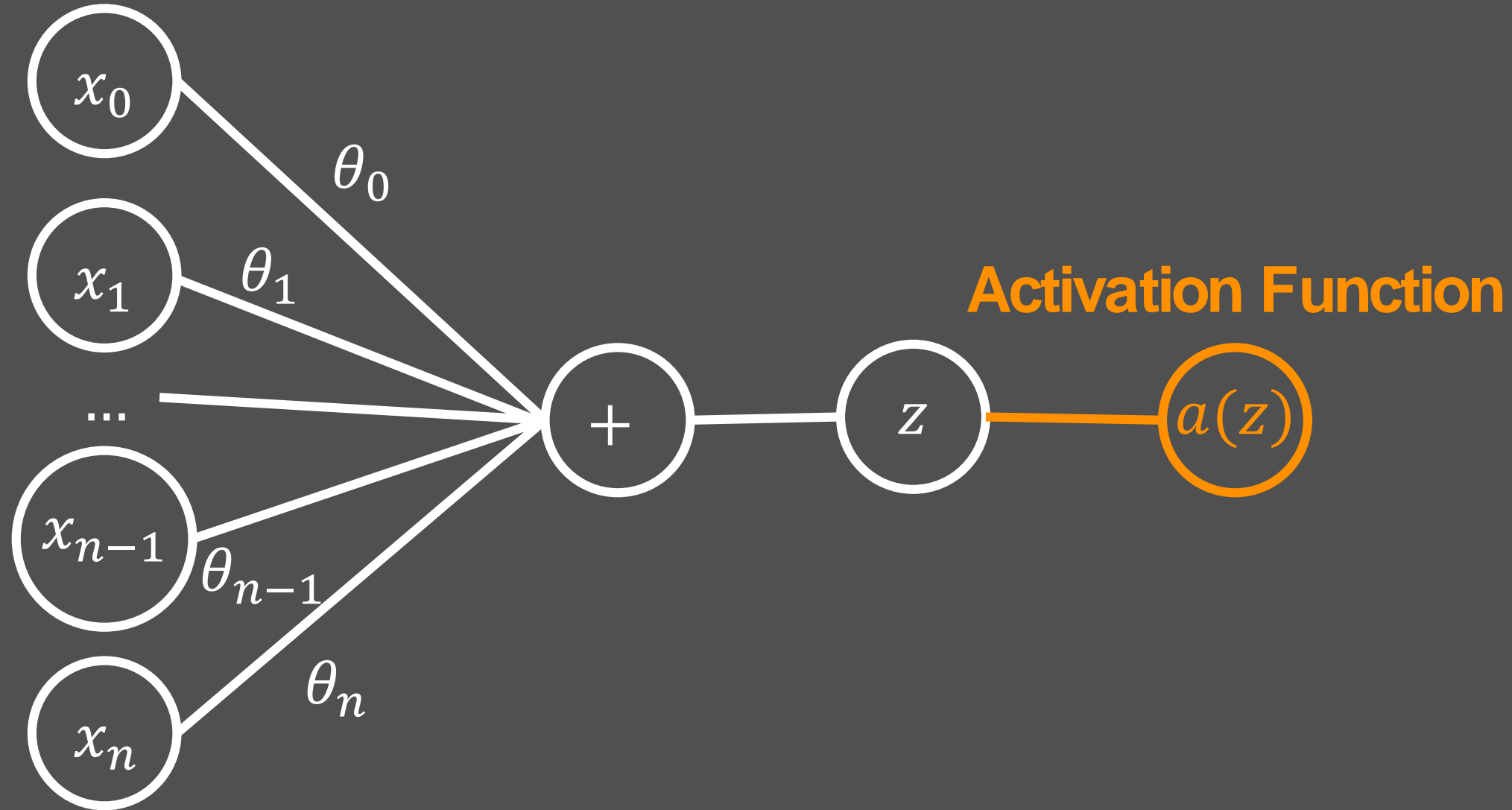
Activation Function $y = a(z)$

Binary Threshold Unit: $a(z) = \begin{cases} 0, & z < 0 \\ 1, & z \geq 0 \end{cases}$

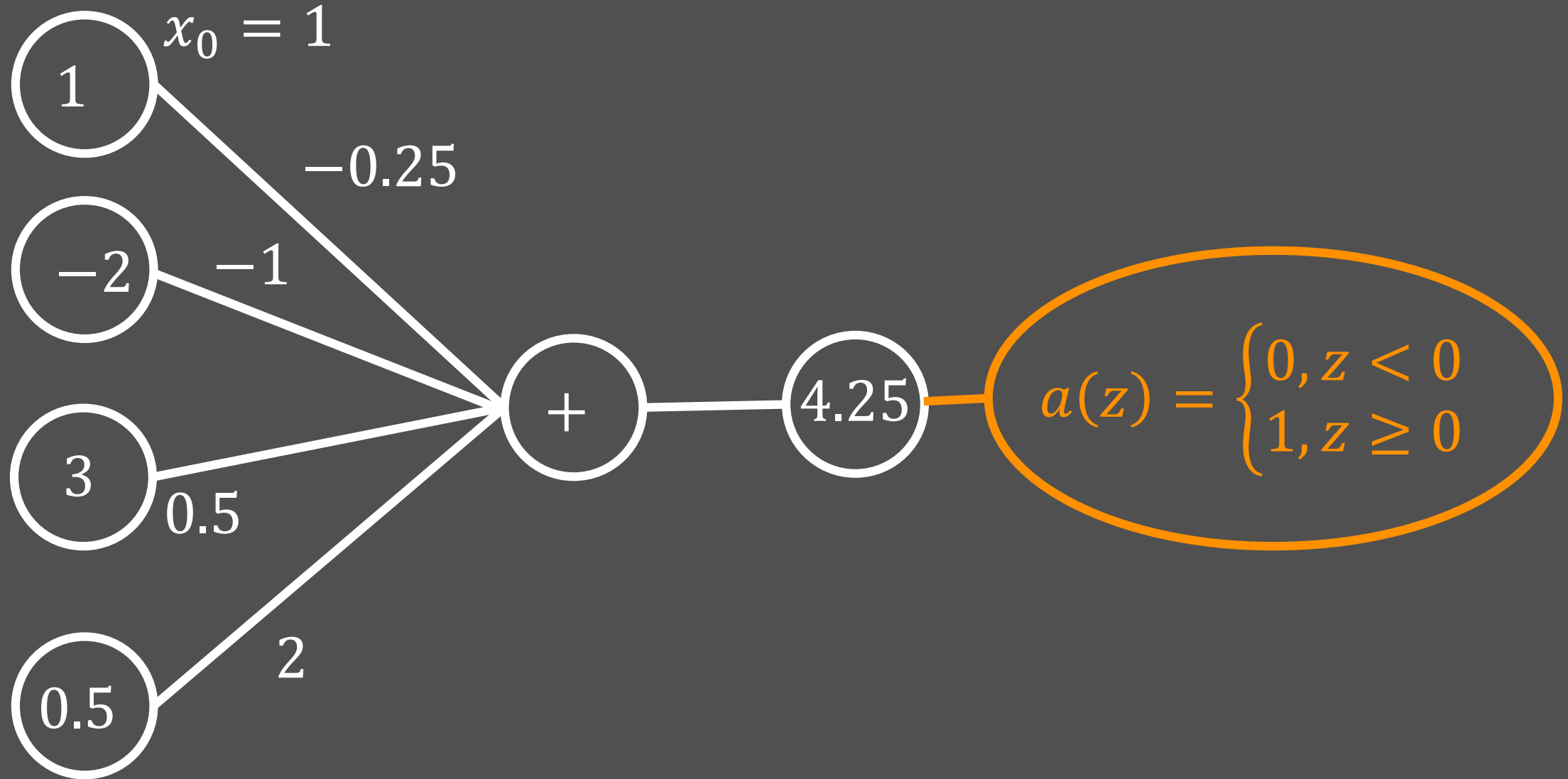
Perceptron



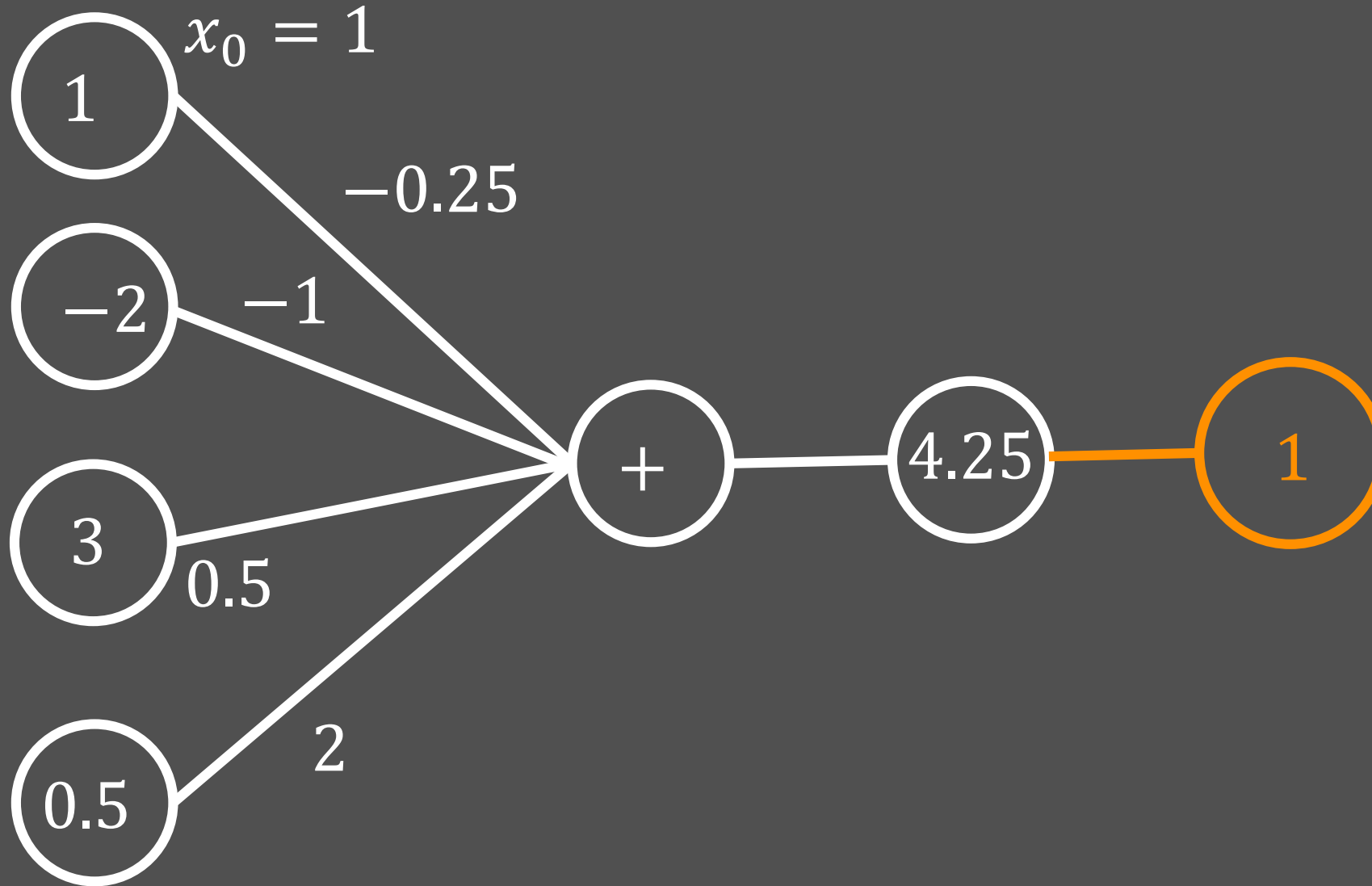
Perceptron



Perceptron

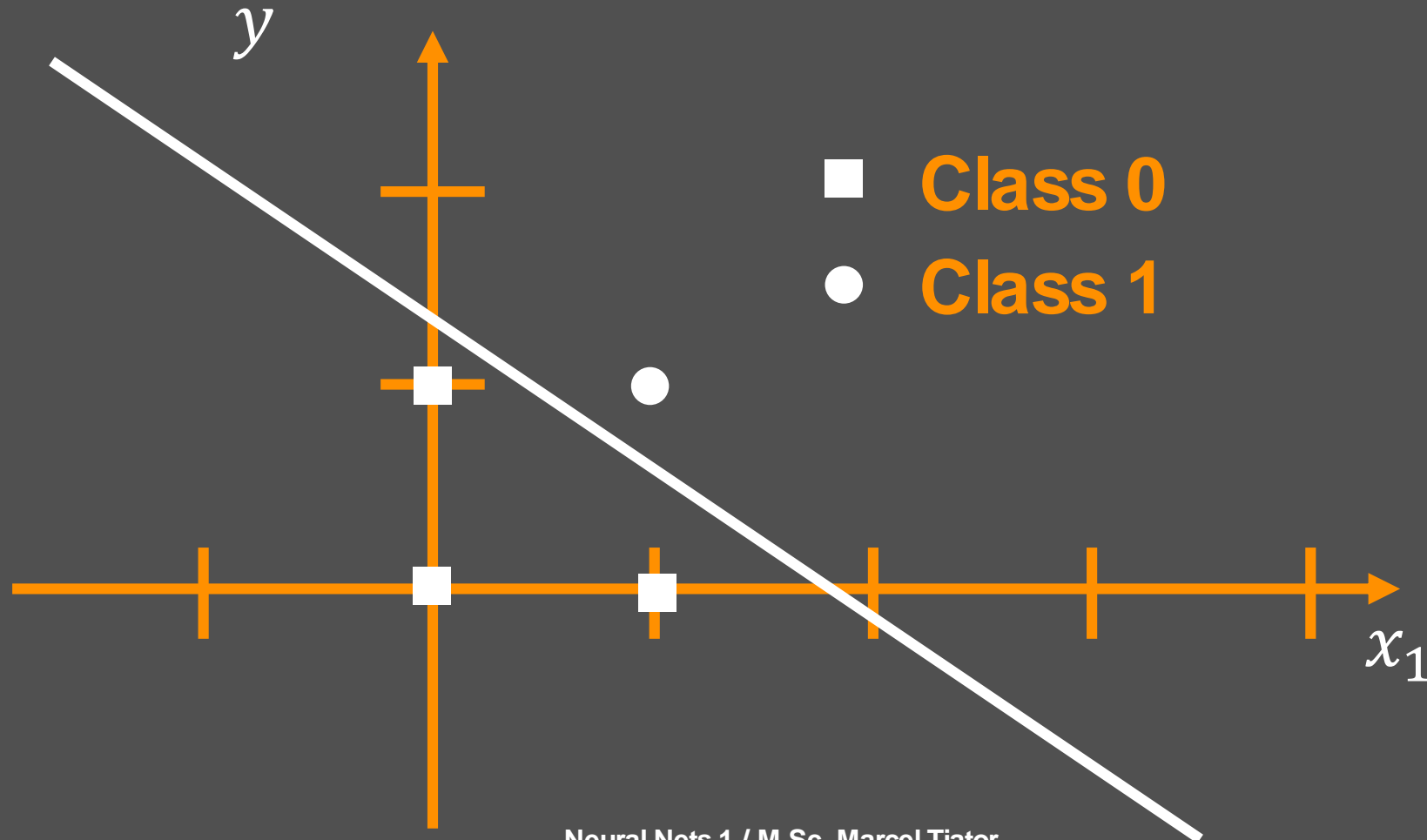


Perceptron

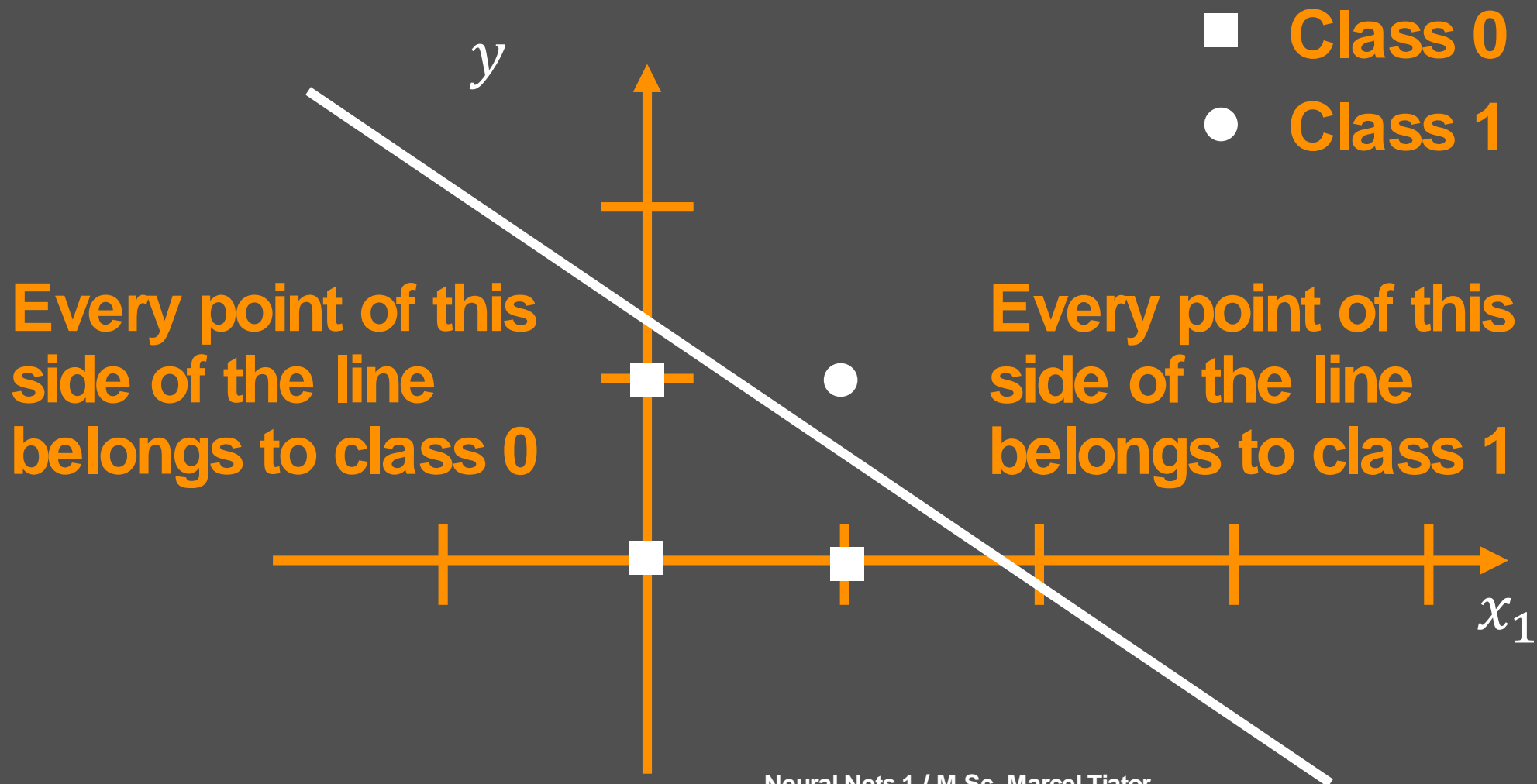


Linear Classifier

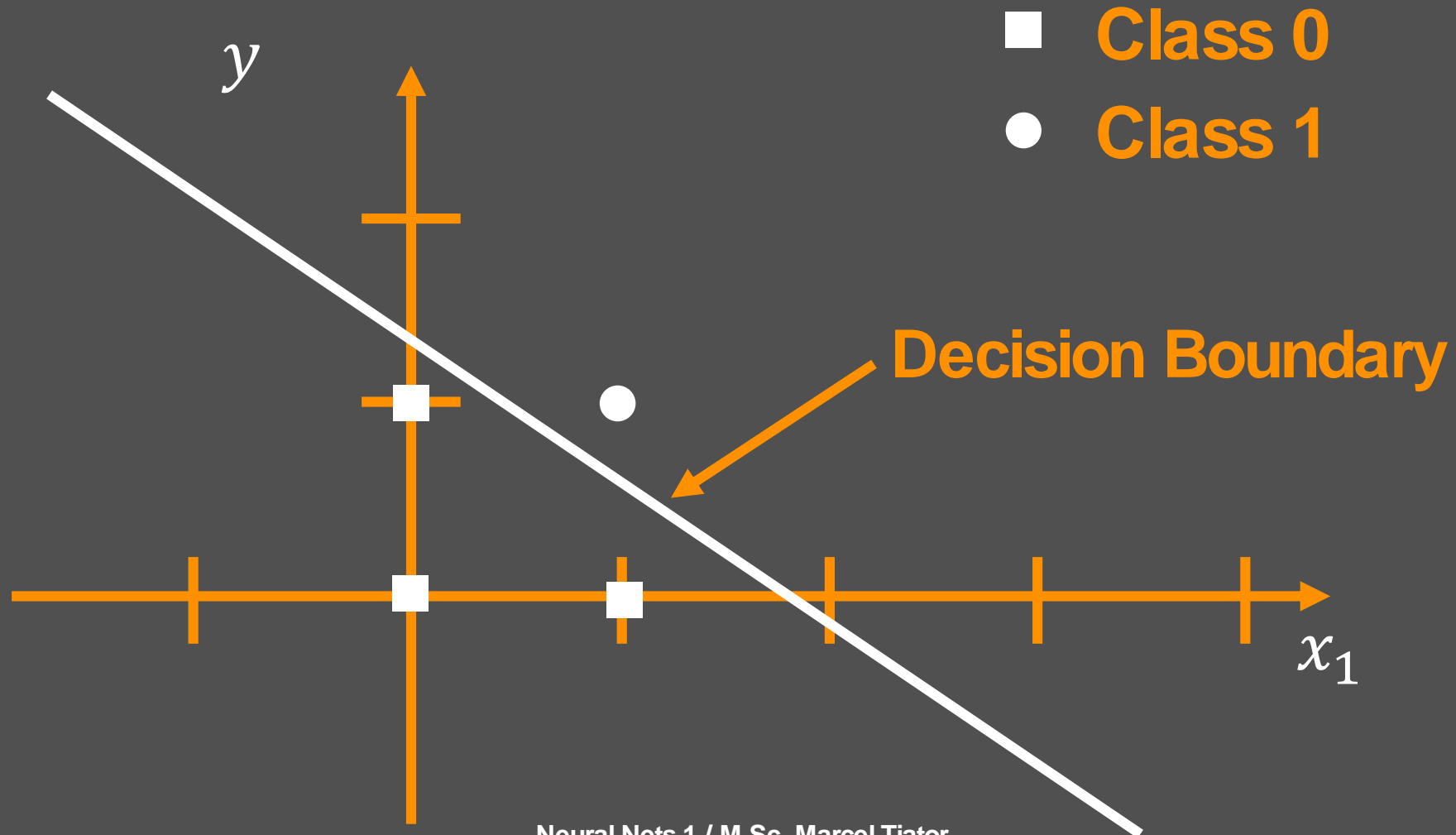
Binary Classification



Binary Classification



Binary Classification



Perceptron

Training:

Initialize weights randomly

do{

 error=false

$d=0$

 foreach x in trainingsset{

$\hat{y}=\text{Prediction}(x)$, $\hat{y} \in \{0,1\}$

$d = y - \hat{y}$, $y \in \{0,1\}$

 if($d == 1$) $\theta = \theta + x$

 elif ($d == -1$) $\theta = \theta - x$

 error =error | $d \neq 0$

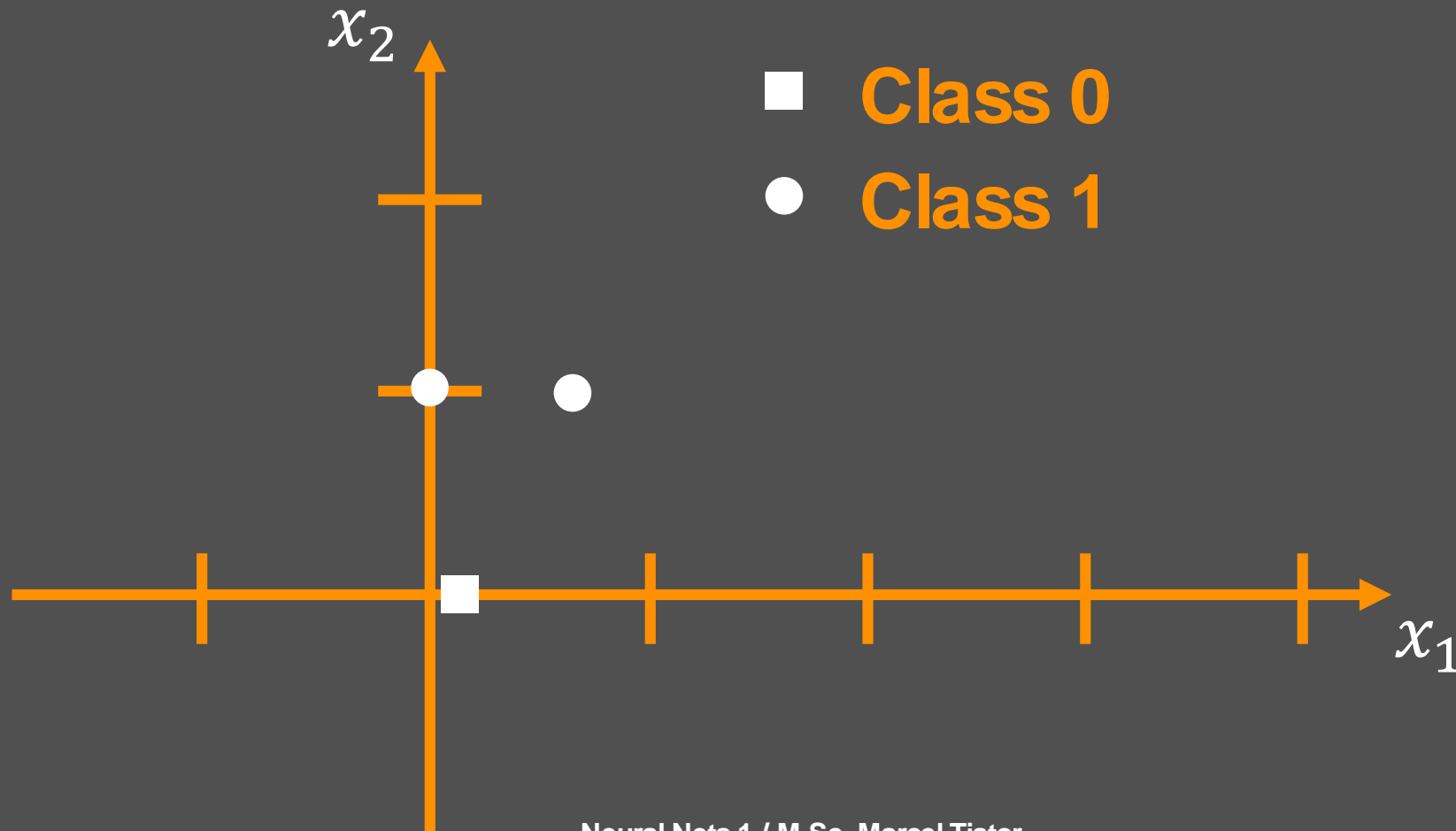
 }

} while(error)

Perceptron

x_1	x_2	y
0.1	0	0
0.6	1	1
0	1	1

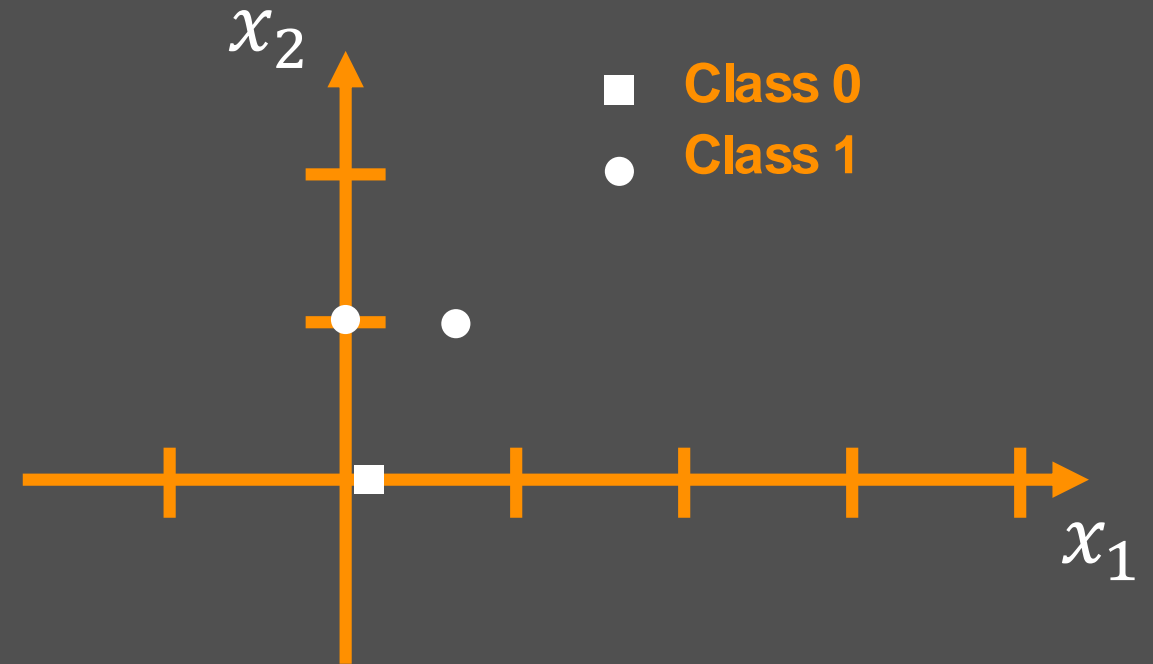
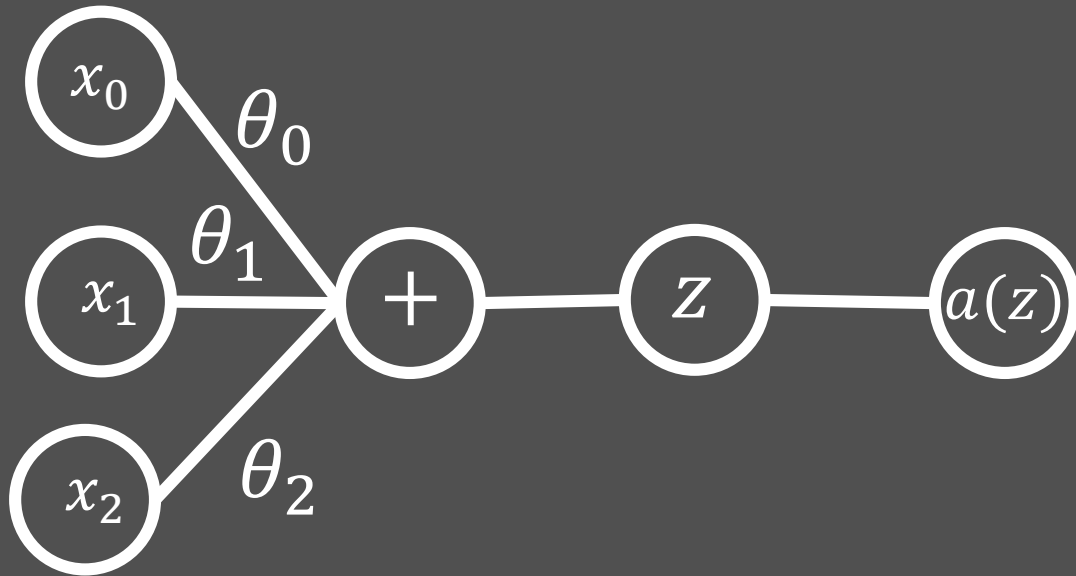
Visualization of data



Append x_0

x_0	x_1	y
1	0.1	0
1	0.6	1
1	0	1

Perceptron-Model



Perceptron

Initialize weights randomly

foreach x **in** **trainingsset**{

$\hat{y} = \text{Prediction}(x)$, $\hat{y} \in \{0, 1\}$

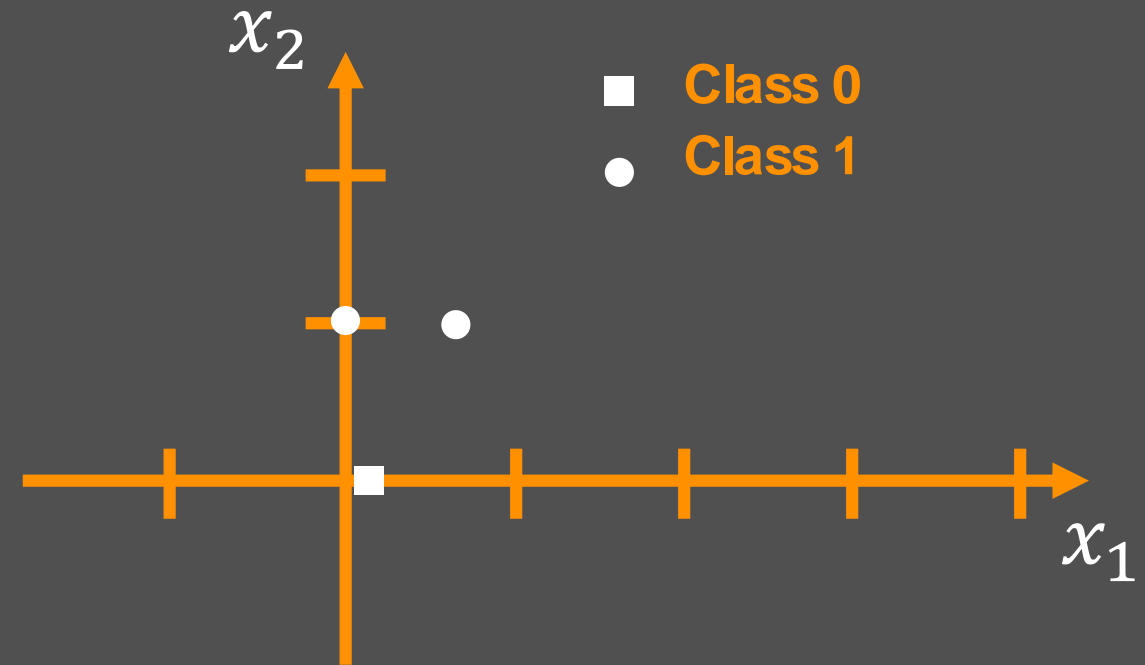
$d = y - \hat{y}$, $y \in \{0, 1\}$

if ($d == 1$) $\theta = \theta + x$

elif ($d == -1$) $\theta = \theta - x$

}

$$\theta_0 = -0.5, \theta_1 = 1, \theta_2 = -1$$



Perceptron

Initialize weights randomly
foreach x in trainingsset{

$\hat{y} = \text{Prediction}(x), \hat{y} \in \{0, 1\}$

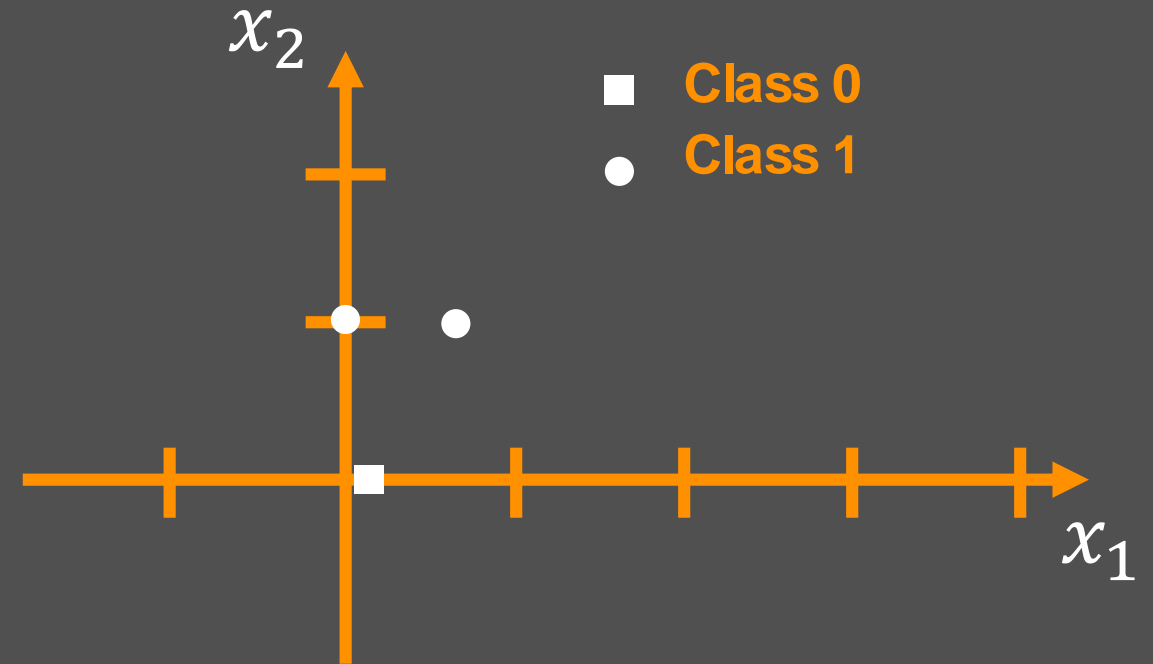
$d = y - \hat{y}, y \in \{0, 1\}$

if ($d == 1$) $\theta = \theta + x$

elif ($d == -1$) $\theta = \theta - x$

}

$$\theta_0 = -0.5, \theta_1 = 1, \theta_2 = -1$$

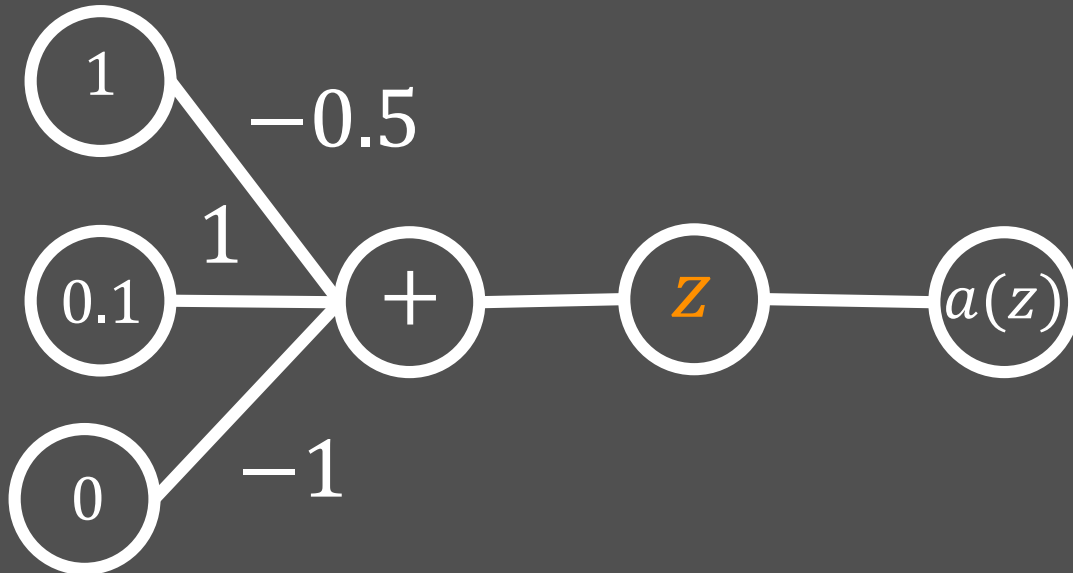


Perceptron

Perceptron-Model

$$z = \theta_1 * x_1 + \theta_0 * x_0$$

$$\theta_0 = -0.5, \theta_1 = 1, \theta_2 = -1$$



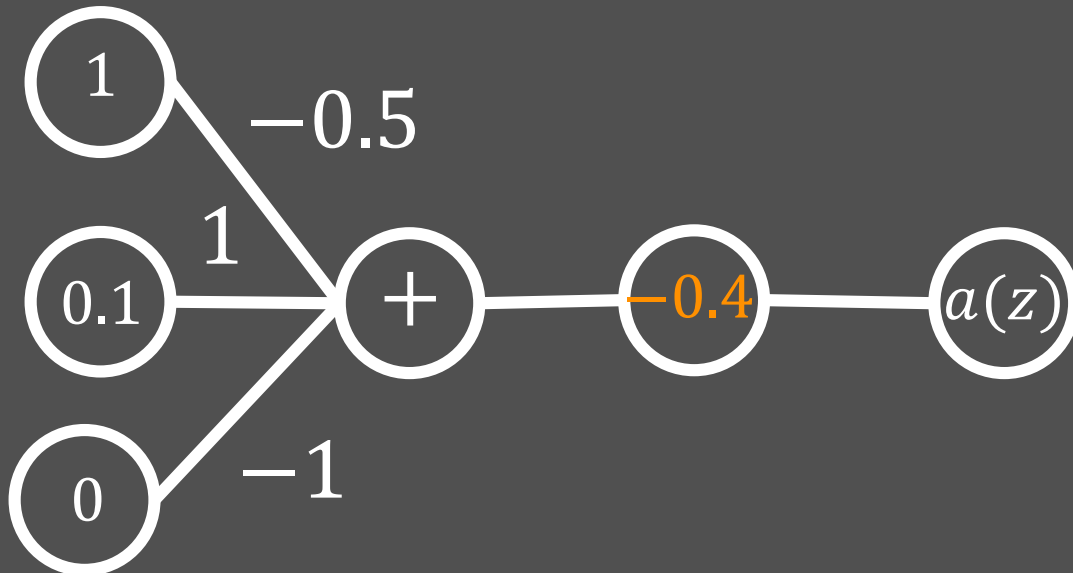
x_0	x_1	x_2	y
1	0.1	0	0
1	0.6	1	1
1	0	1	1

Perceptron

Perceptron-Model

$$z = \theta_1 * x_1 + \theta_0 * x_0$$

$$\theta_0 = -0.5, \theta_1 = 1, \theta_2 = -1$$

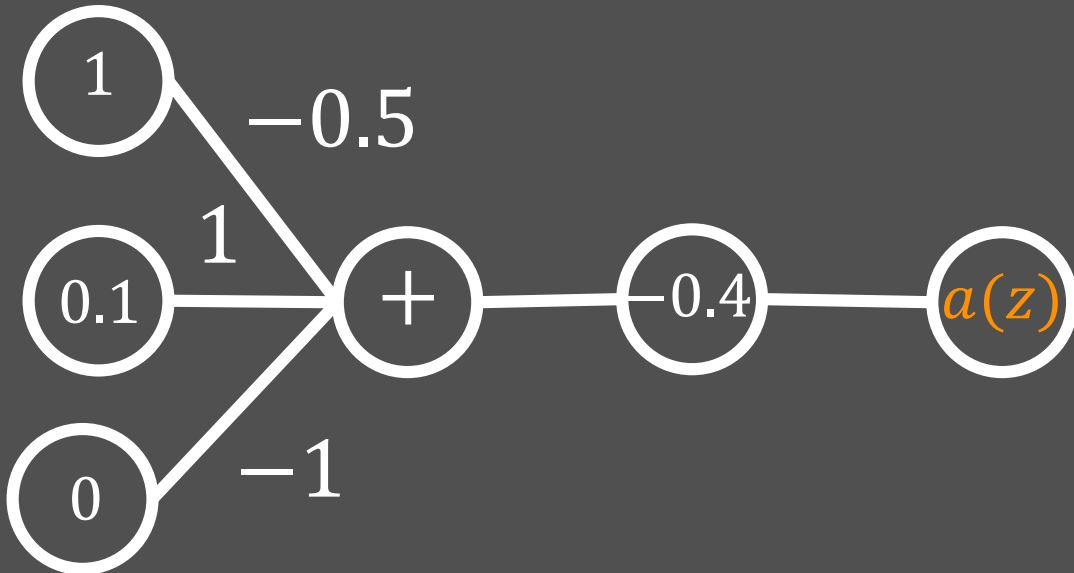


x_0	x_1	x_2	y
1	0.1	0	0
1	0.6	1	1
1	0	1	1

Perceptron

Perceptron-Model

$$a(z) = \begin{cases} 0, & z < 0 \\ 1, & z \geq 0 \end{cases}$$



$$\theta_0 = -0.5, \theta_1 = 1, \theta_2 = -1$$

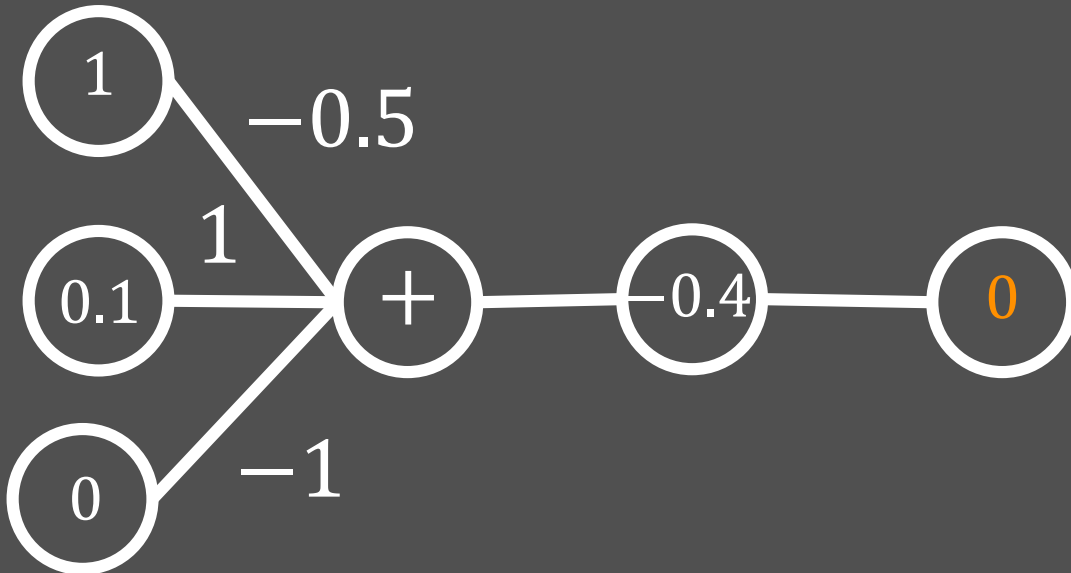


x_0	x_1	x_2	y
1	0.1	0	0
1	0.6	1	1
1	0	1	1

Perceptron

Perceptron-Model

$$\hat{y} = a(z) = \begin{cases} 0, & z < 0 \\ 1, & z \geq 0 \end{cases}$$



$$\theta_0 = -0.5, \theta_1 = 1, \theta_2 = -1$$

x_0	x_1	x_2	y
1	0.1	0	0
1	0.6	1	1
1	0	1	1

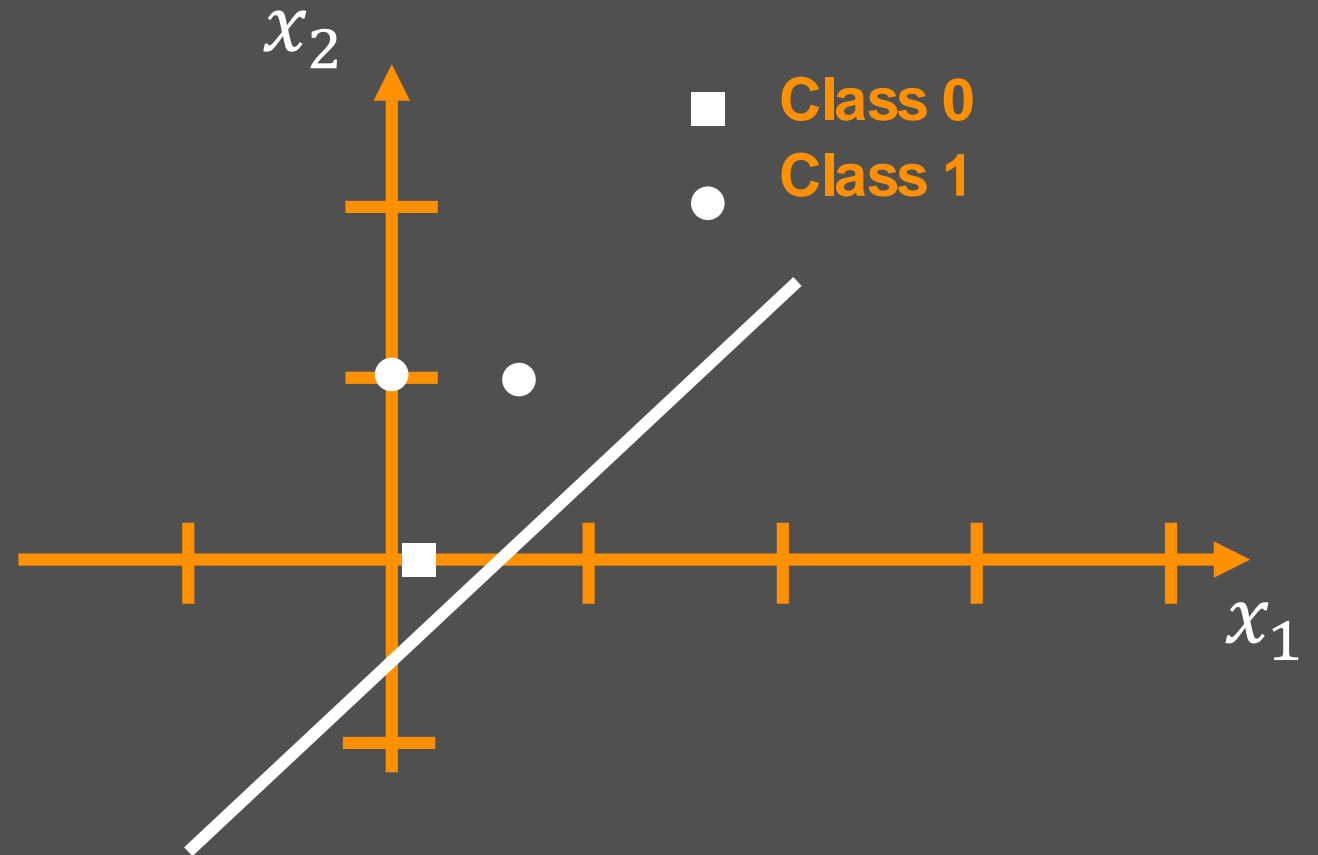
$$x_2\theta_2 + x_1\theta_1 + x_0\theta_0 = 0$$

$$x_2 = \frac{-(x_1\theta_1 + x_0\theta_0)}{\theta_2}$$

$$x_2 = \frac{-(1x_1 - 0.5)}{-1}$$

$$x_2 = x_1 - 0.5$$

$$\theta_0 = -0.5, \theta_1 = 1, \theta_2 = -1$$



Perceptron

Initialize weights randomly

foreach x in trainingsset{

$\hat{y} = \text{Prediction}(x)$, $\hat{y} \in \{0, 1\}$

$d = y - \hat{y}$, $y \in \{0, 1\}$

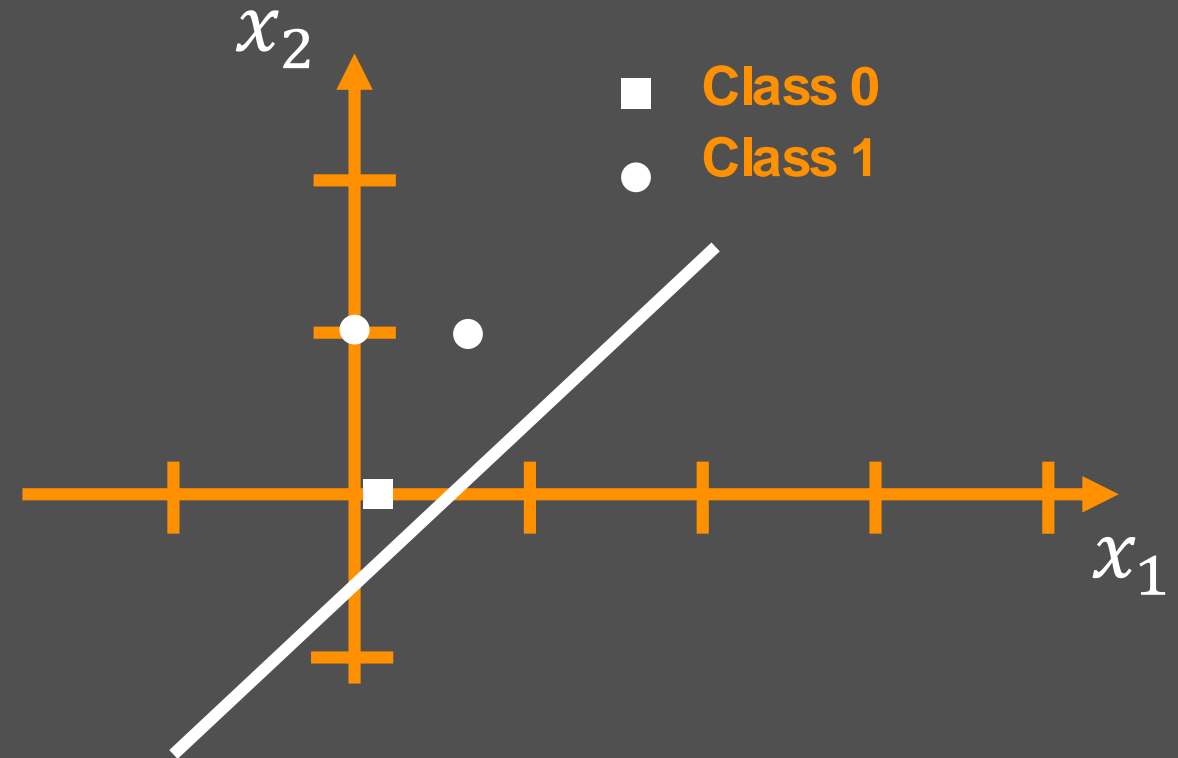
if ($d == 1$) $\theta = \theta + x$

elif ($d == -1$) $\theta = \theta - x$

}

$\theta_0 = -0.5, \theta_1 = 1, \theta_2 = -1$,

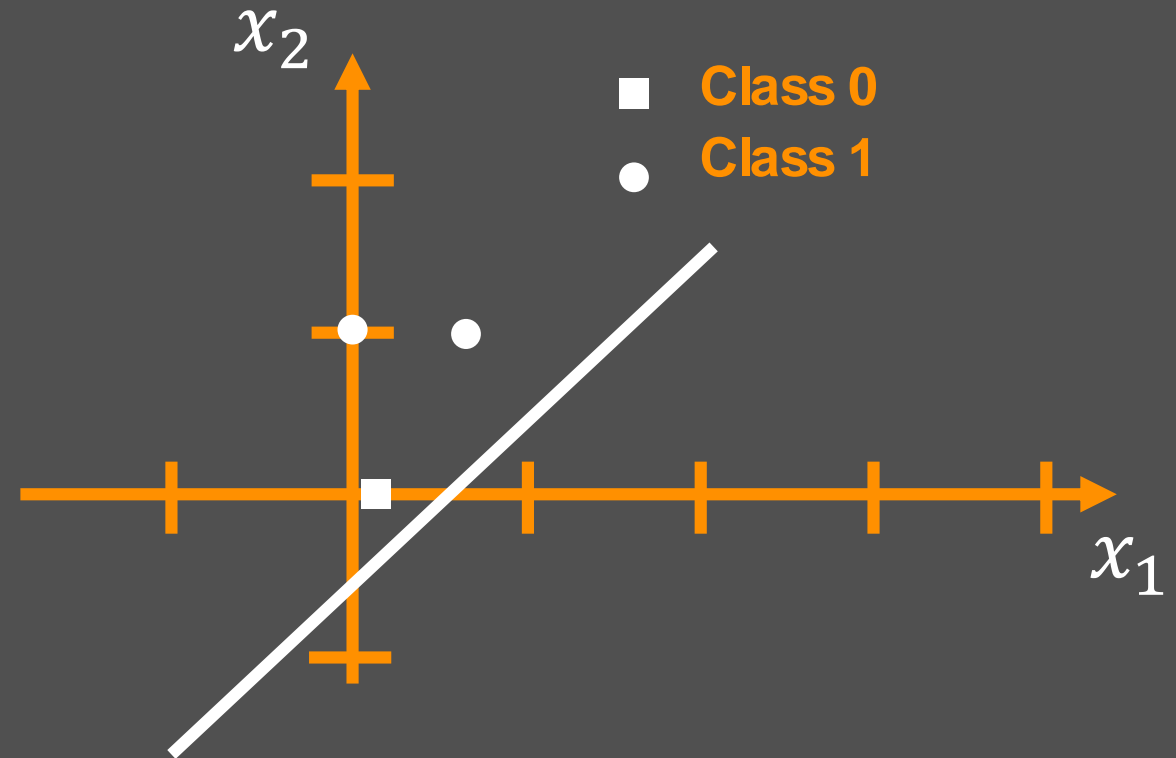
$\hat{y} = 0$



Perceptron

```
Initialize weights randomly
foreach  $x$  in trainingsset{
     $\hat{y} = \text{Prediction}(x)$ ,  $\hat{y} \in \{0, 1\}$ 
     $d = y - \hat{y}$ ,  $y \in \{0, 1\}$ 
    if ( $d == 1$ )  $\theta = \theta + x$ 
    elif ( $d == -1$ )  $\theta = \theta - x$ 
}
```

$$\theta_0 = -0.5, \theta_1 = 1, \theta_2 = -1, \\ \hat{y} = 0$$



Perceptron

Initialize weights randomly

foreach x in trainingsset{

$\hat{y} = \text{Prediction}(x)$, $\hat{y} \in \{0, 1\}$

$d = y - \hat{y}$, $y \in \{0, 1\}$

if ($d == 1$) $\theta = \theta + x$

elif ($d == -1$) $\theta = \theta - x$

}

$$\theta_0 = -0.5, \theta_1 = 1, \theta_2 = -1,$$

$$\hat{y} = 0, d = 0$$



x_0	x_1	x_2	y
1	0.1	0	0
1	0.6	1	1
1	0	1	1

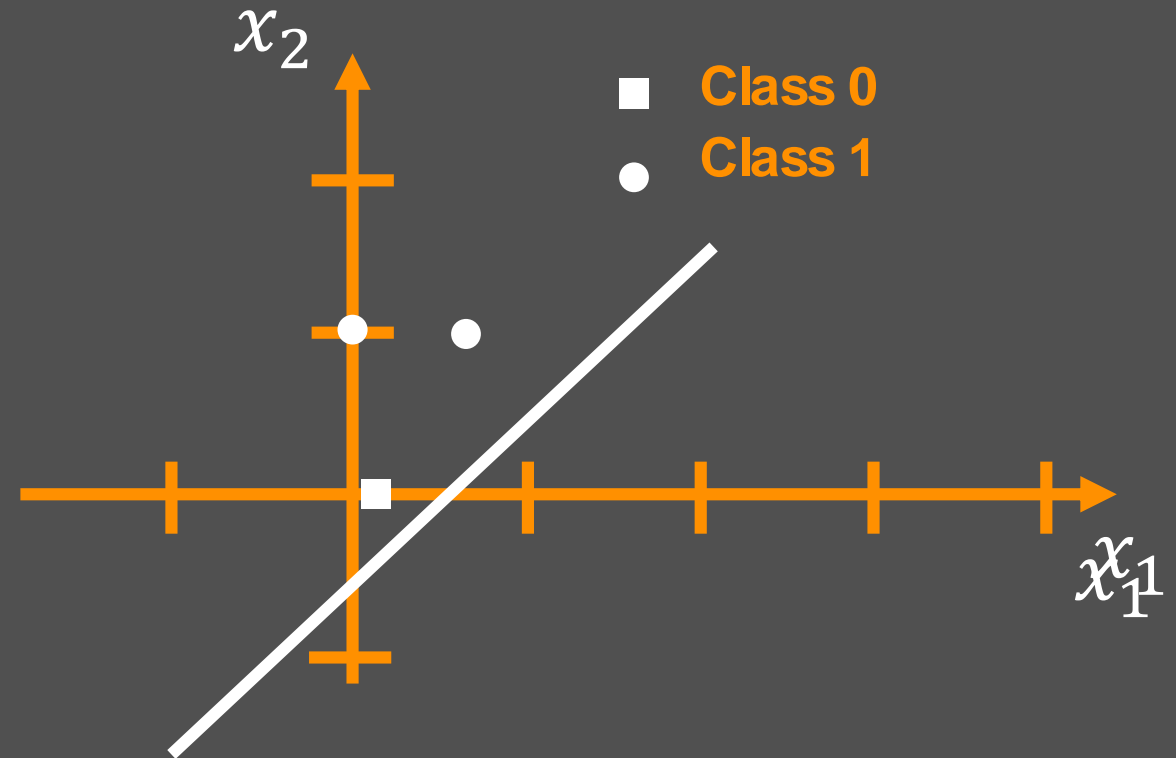
Perceptron

```

Initialize weights randomly
foreach  $x$  in trainingsset{
     $\hat{y} = \text{Prediction}(x)$ ,  $\hat{y} \in \{0, 1\}$ 
     $d = y - \hat{y}$ ,  $y \in \{0, 1\}$ 
    if ( $d == 1$ )  $\theta = \theta + x$ 
    elif ( $d == -1$ )  $\theta = \theta - x$ 
}
    
```

$$\theta_0 = -0.5, \theta_1 = 1, \theta_2 = -1,$$

$$\hat{y} = 0, d = 0$$



Perceptron

Initialize weights randomly

foreach x in trainingsset{

$\hat{y} = \text{Prediction}(x)$, $\hat{y} \in \{0, 1\}$

$d = y - \hat{y}$, $y \in \{0, 1\}$

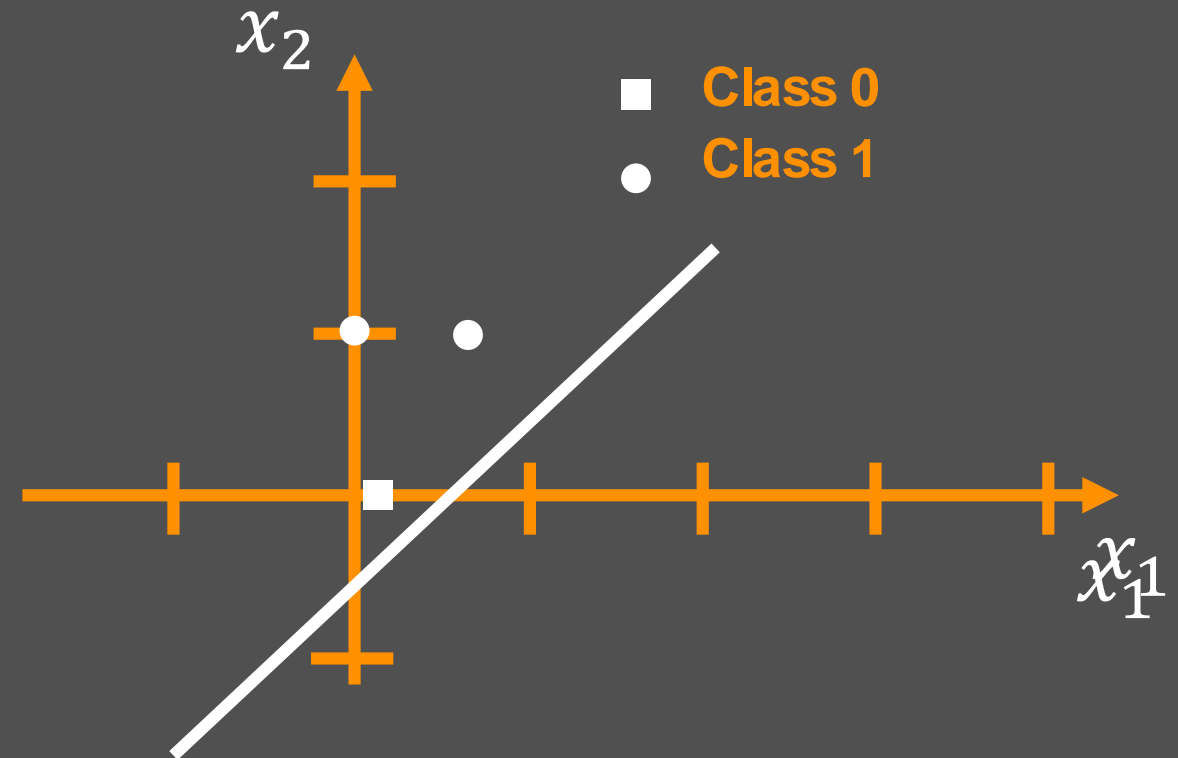
if($d == 1$) $\theta = \theta + x$

elif ($d == -1$) $\theta = \theta - x$

}

$$\theta_0 = -0.5, \theta_1 = 1, \theta_2 = -1,$$

$$\hat{y} = 0, d = 0$$



Perceptron

Initialize weights randomly

foreach x in trainingsset{

$\hat{y} = \text{Prediction}(x)$, $\hat{y} \in \{0, 1\}$

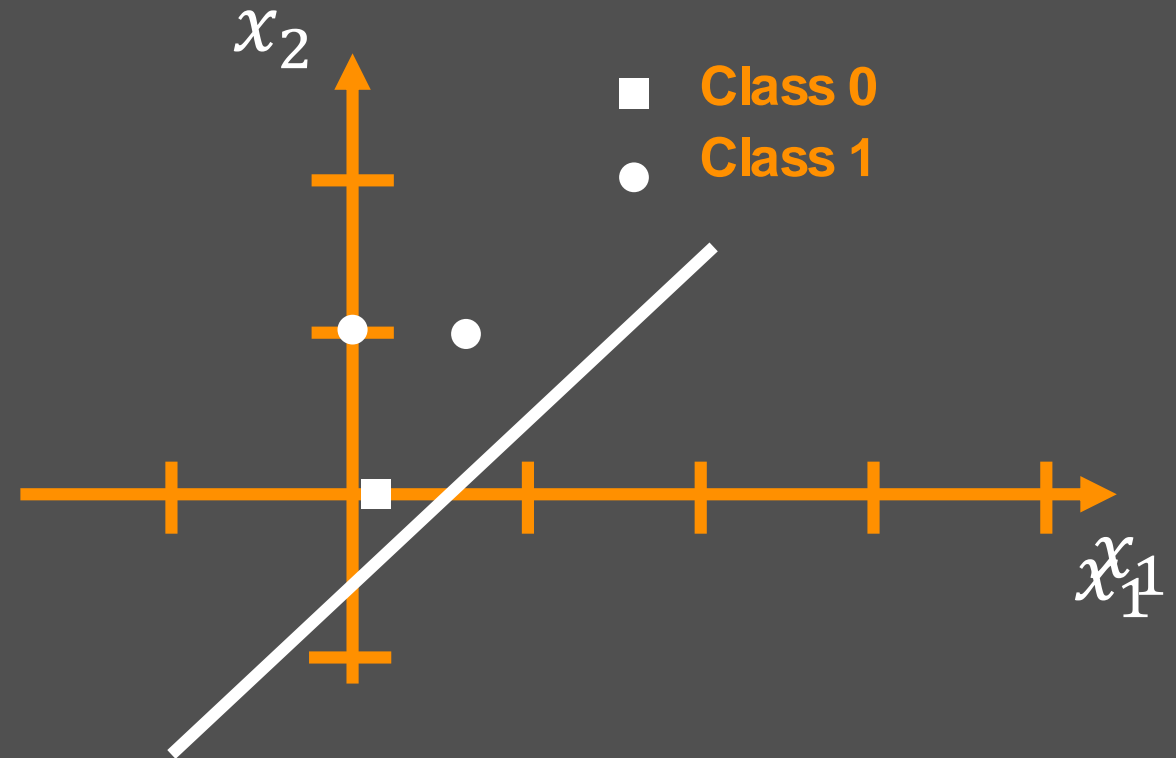
$d = y - \hat{y}$, $y \in \{0, 1\}$

if ($d == 1$) $\theta = \theta + x$

elif ($d == -1$) $\theta = \theta - x$

}

$$\theta_0 = -0.5, \theta_1 = 1, \theta_2 = -1$$

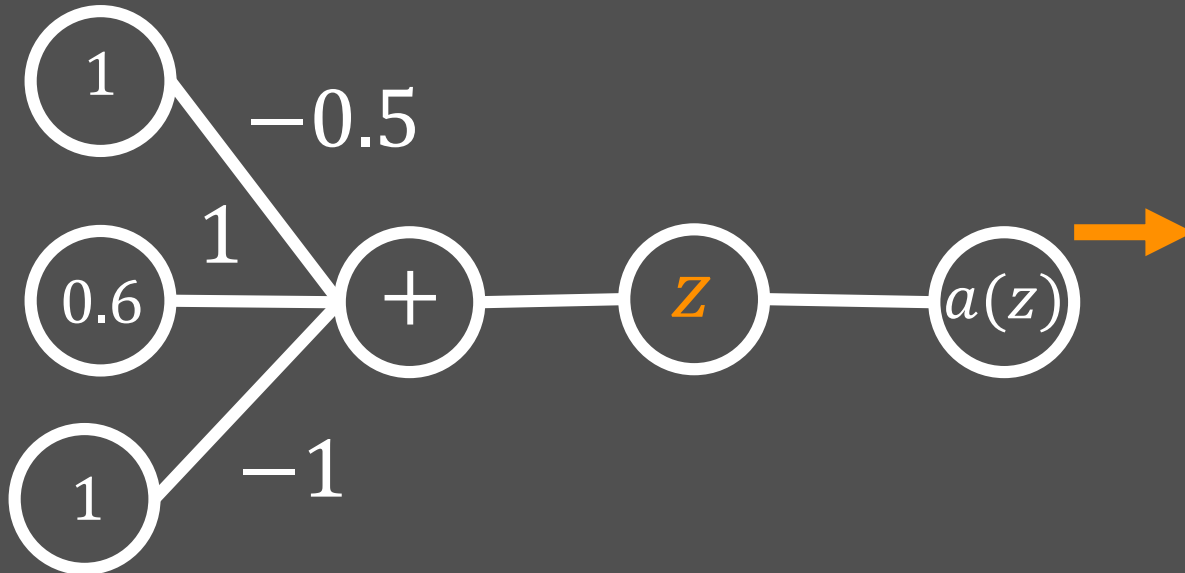


Perceptron

Perceptron-Model

$$z = \theta_1 * x_1 + \theta_0 * x_0$$

$$\theta_0 = -0.5, \theta_1 = 1, \theta_2 = -1$$



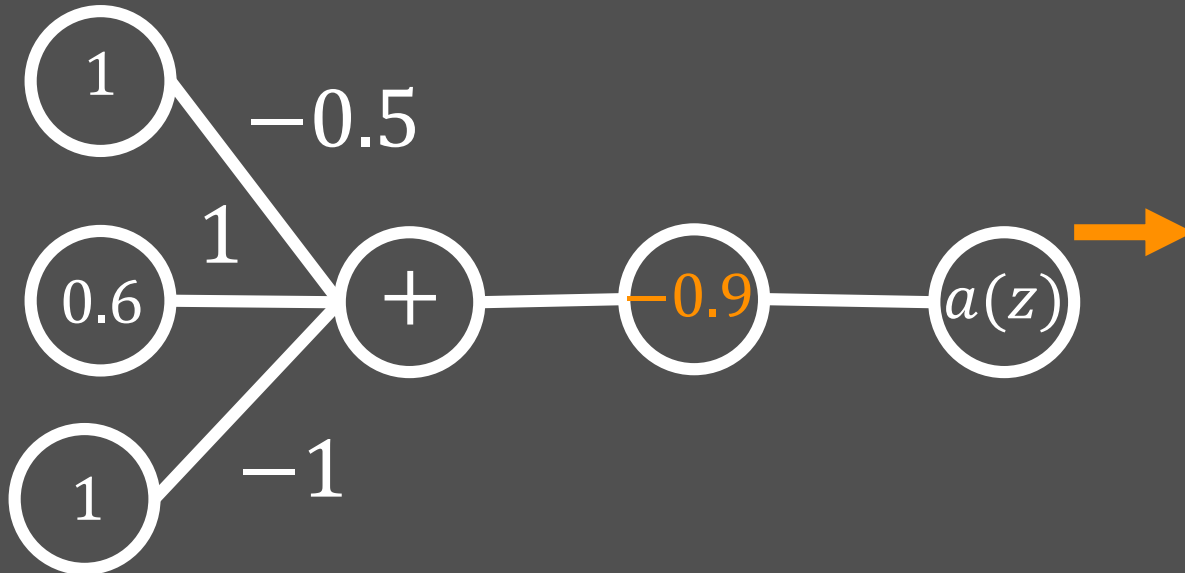
x_0	x_1	x_2	y
1	0.1	0	0
1	0.6	1	1
1	0	1	1

Perceptron

Perceptron-Model

$$z = \theta_1 * x_1 + \theta_0 * x_0$$

$$\theta_0 = -0.5, \theta_1 = 1, \theta_2 = -1$$

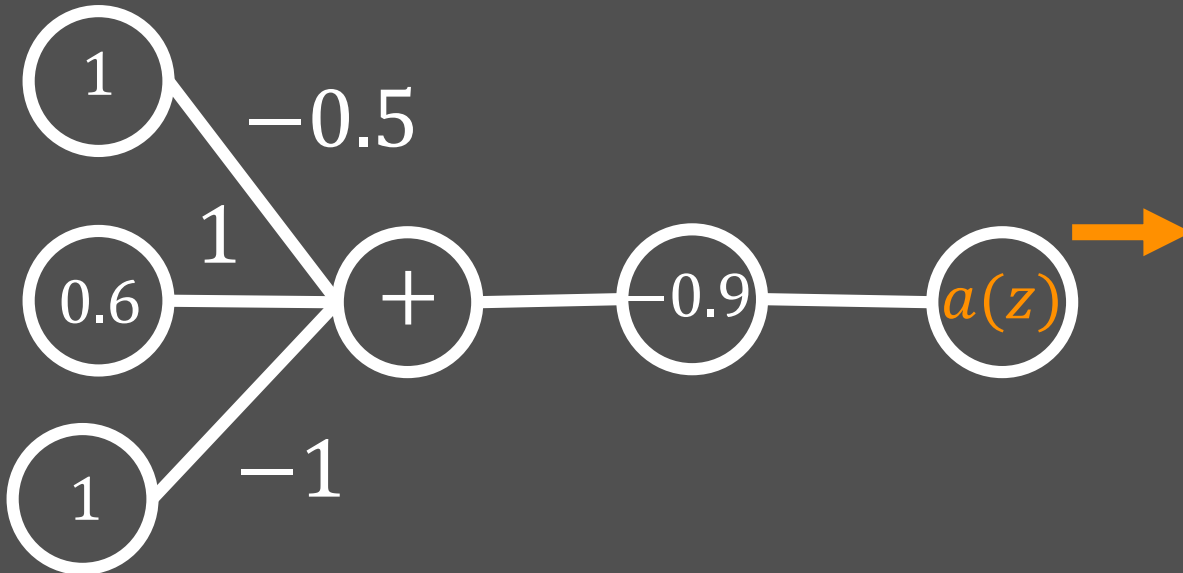


x_0	x_1	x_2	y
1	0.1	0	0
1	0.6	1	1
1	0	1	1

Perceptron

Perceptron-Model

$$a(z) = \begin{cases} 0, & z < 0 \\ 1, & z \geq 0 \end{cases}$$



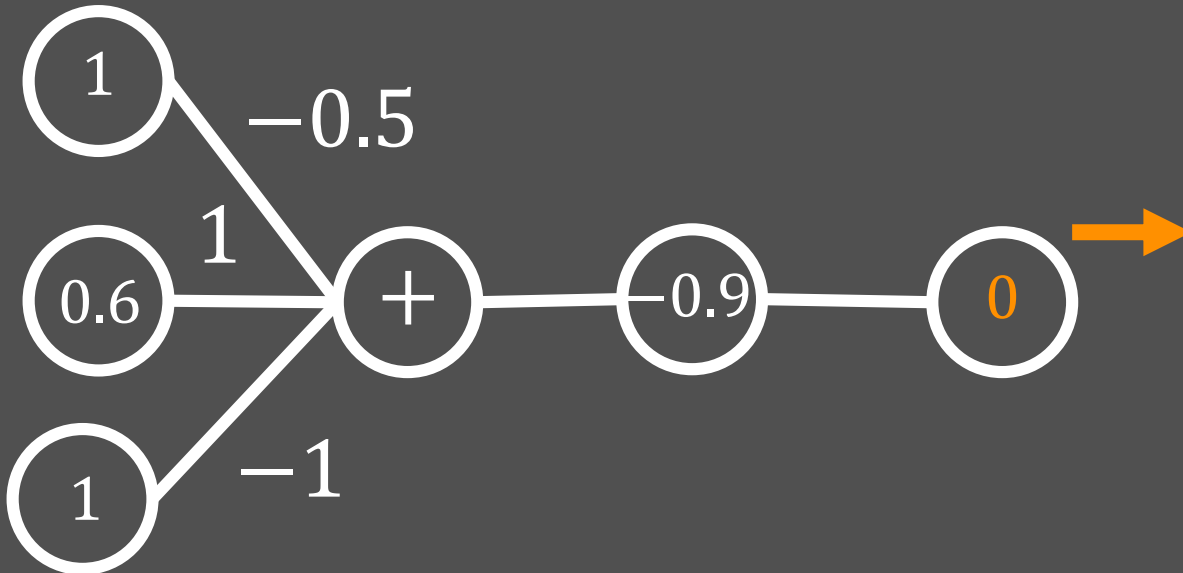
$$\theta_0 = -0.5, \theta_1 = 1, \theta_2 = -1$$

x_0	x_1	x_2	y
1	0.1	0	0
1	0.6	1	1
1	0	1	1

Perceptron

Perceptron-Model

$$\hat{y} = a(z) = \begin{cases} 0, & z < 0 \\ 1, & z \geq 0 \end{cases}$$



$$\theta_0 = -0.5, \theta_1 = 1, \theta_2 = -1$$

x_0	x_1	x_2	y
1	0.1	0	0
1	0.6	1	1
1	0	1	1

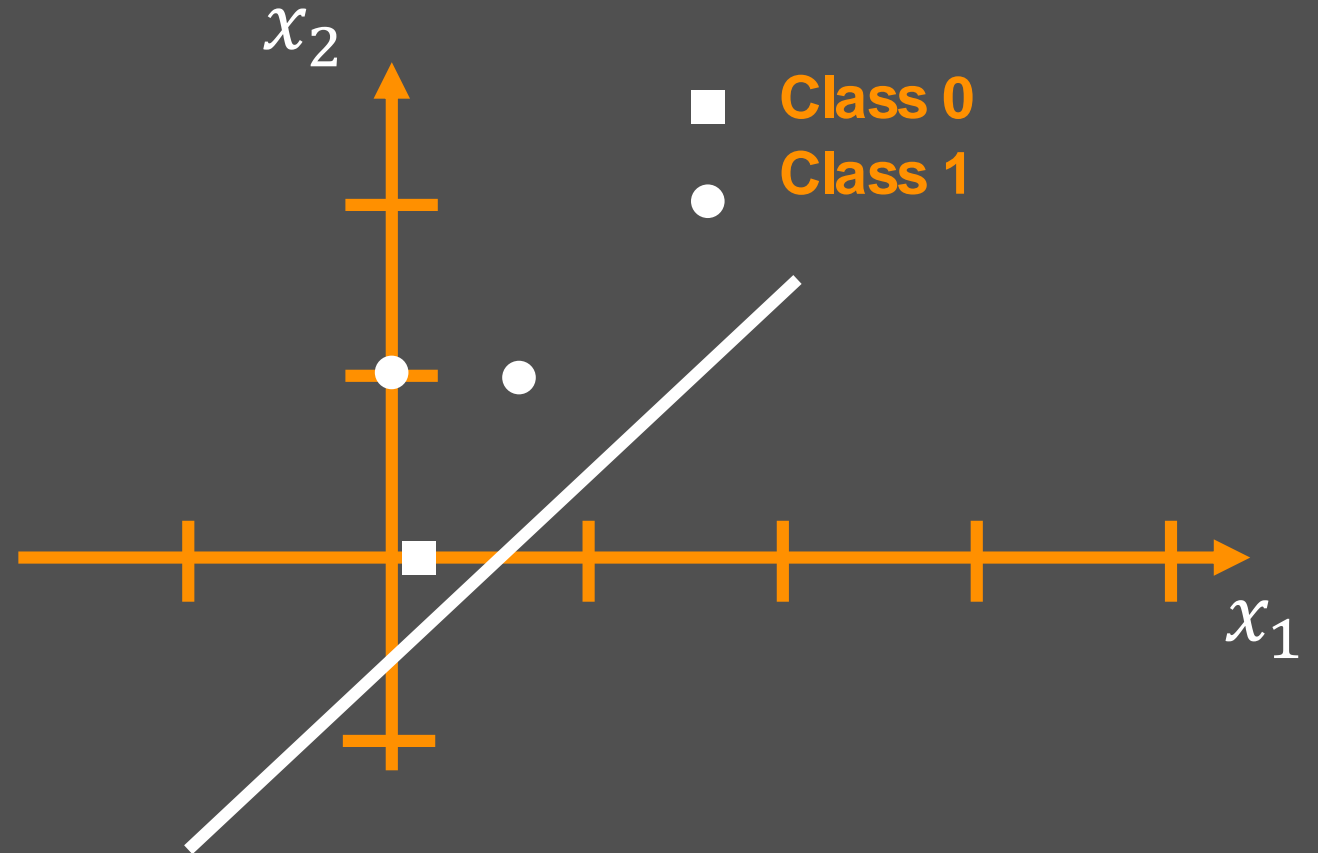
$$x_2\theta_2 + x_1\theta_1 + x_0\theta_0 = 0$$

$$x_2 = \frac{-(x_1\theta_1 + x_0\theta_0)}{\theta_2}$$

$$x_2 = \frac{-(1x_1 - 0.5)}{-1}$$

$$x_2 = x_1 - 0.5$$

$$\theta_0 = -0.5, \theta_1 = 1, \theta_2 = -1$$



Perceptron

Initialize weights randomly

foreach x in trainingsset{

$\hat{y} = \text{Prediction}(x)$, $\hat{y} \in \{0, 1\}$

$d = y - \hat{y}$, $y \in \{0, 1\}$

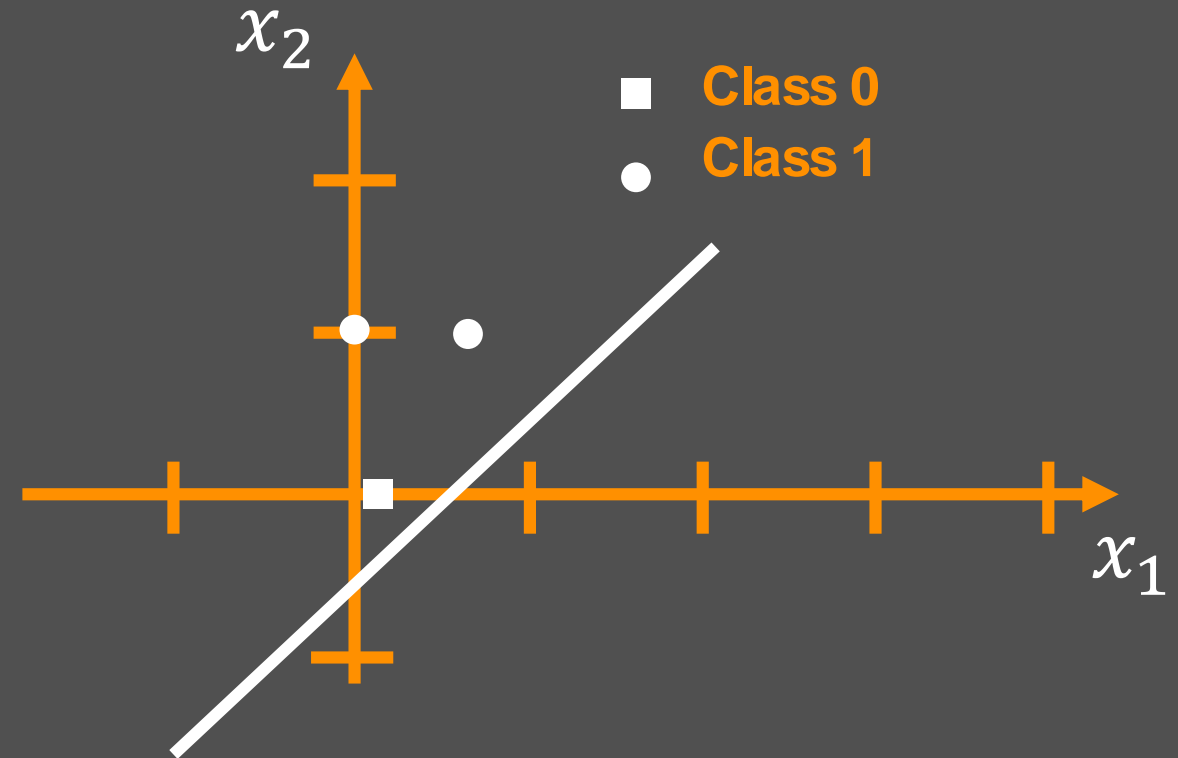
if ($d == 1$) $\theta = \theta + x$

elif ($d == -1$) $\theta = \theta - x$

}

$\theta_0 = -0.5, \theta_1 = 1, \theta_2 = -1,$

$\hat{y} = 0$



Perceptron

Initialize weights randomly

foreach x in trainingsset{

$\hat{y} = \text{Prediction}(x)$, $\hat{y} \in \{0, 1\}$

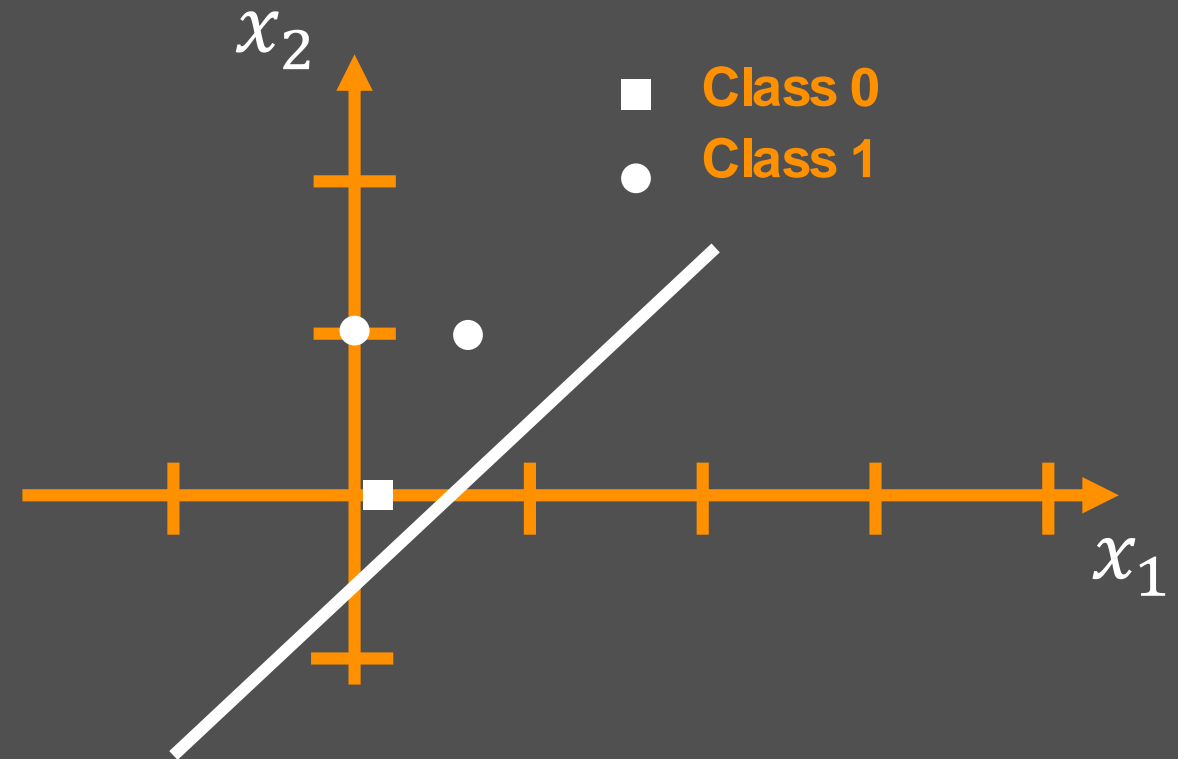
$d = y - \hat{y}$, $y \in \{0, 1\}$

if ($d == 1$) $\theta = \theta + x$

elif ($d == -1$) $\theta = \theta - x$

}

$$\theta_0 = -0.5, \theta_1 = 1, \theta_2 = -1, \\ \hat{y} = 0$$



Perceptron

Initialize weights randomly

foreach x in trainingsset{

$\hat{y} = \text{Prediction}(x)$, $\hat{y} \in \{0, 1\}$

$d = y - \hat{y}$, $y \in \{0, 1\}$

if($d == 1$) $\theta = \theta + x$

elif ($d == -1$) $\theta = \theta - x$ →

}

$$\theta_0 = -0.5, \theta_1 = 1, \theta_2 = -1, \\ \hat{y} = 0, d = 1$$

x_0	x_1	x_2	y
1	0.1	0	0
1	0.6	1	1
1	0	1	1

Perceptron

Initialize weights randomly

foreach x in trainingsset{

$\hat{y} = \text{Prediction}(x)$, $\hat{y} \in \{0, 1\}$

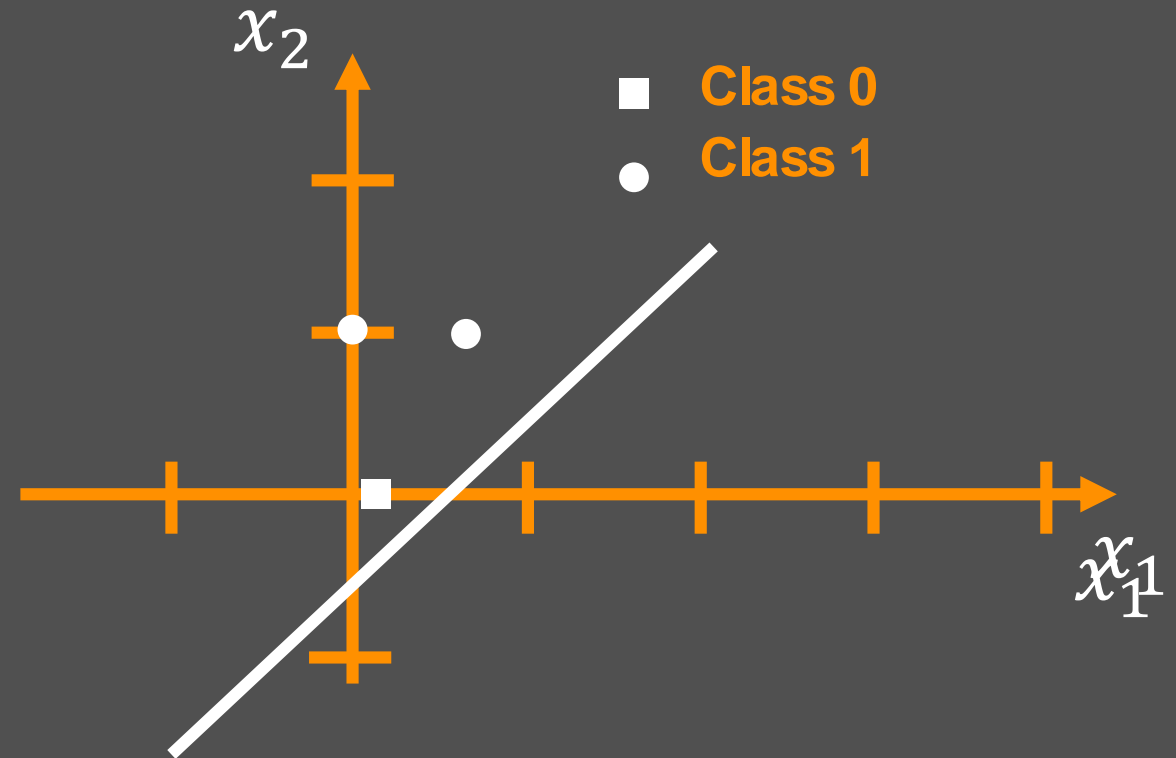
$d = y - \hat{y}$, $y \in \{0, 1\}$

if ($d == 1$) $\theta = \theta + x$

elif ($d == -1$) $\theta = \theta - x$

}

$$\theta_0 = -0.5, \theta_1 = 1, \theta_2 = -1, \\ \hat{y} = 0, d = 1$$



Perceptron

Initialize weights randomly

foreach x in trainingsset{

$\hat{y} = \text{Prediction}(x)$, $\hat{y} \in \{0, 1\}$

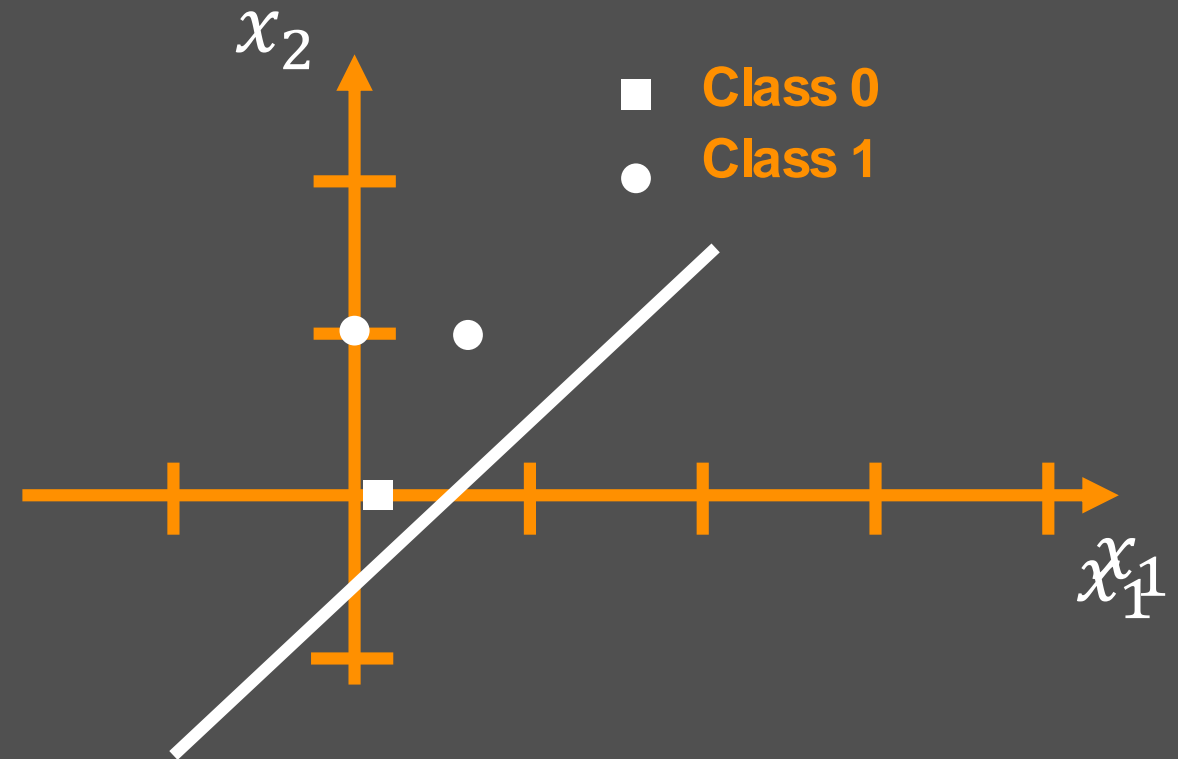
$d = y - \hat{y}$, $y \in \{0, 1\}$

if ($d == 1$) $\theta = \theta + x$

elif ($d == -1$) $\theta = \theta - x$

}

$$\theta_0 = -0.5, \theta_1 = 1, \theta_2 = -1, \\ \hat{y} = 0, d = 1$$



if($d==1$) $\theta = \theta + x$

$$\theta = \begin{pmatrix} -0.5 \\ 1 \\ -1 \end{pmatrix} + \begin{pmatrix} 1 \\ 0.6 \\ 1 \end{pmatrix}$$

$$\theta = \begin{pmatrix} 0.5 \\ 1.6 \\ 0 \end{pmatrix}$$

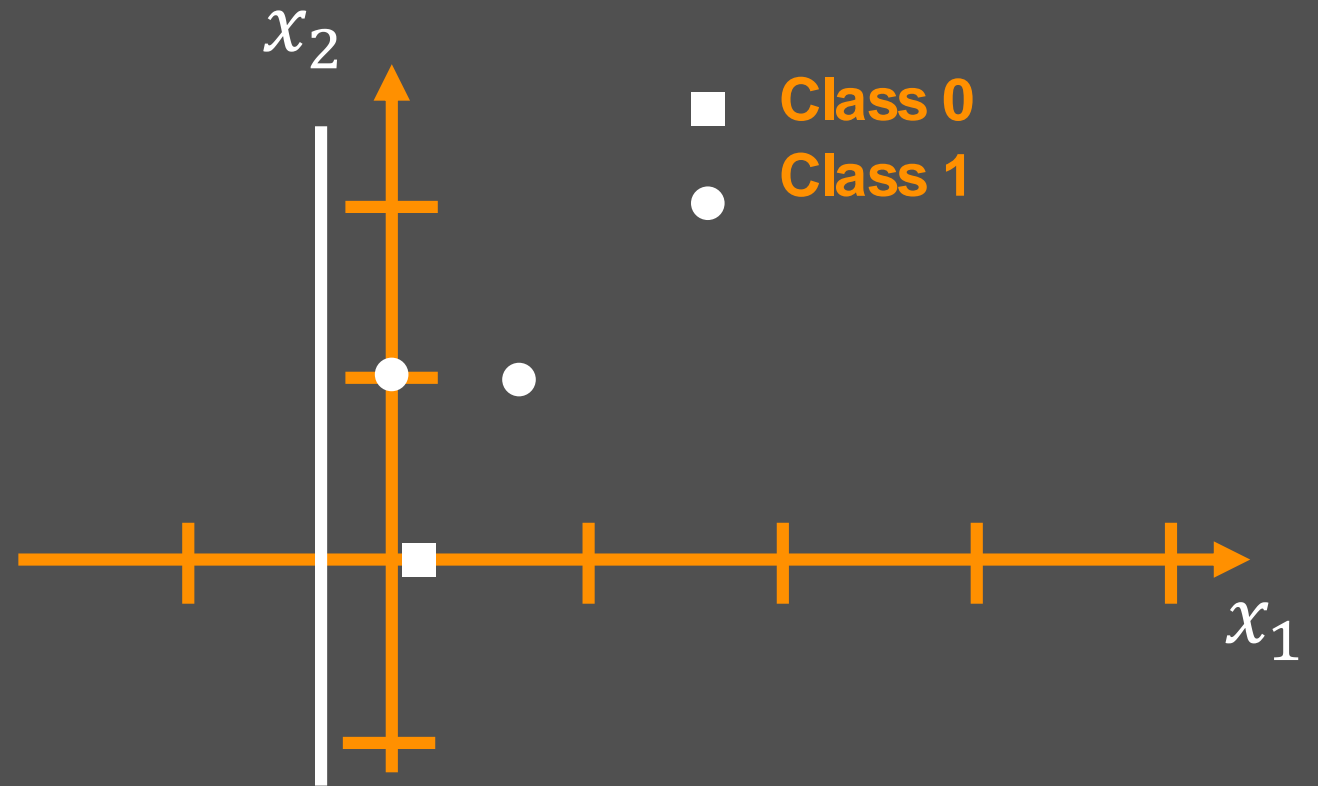
$$\theta_0 = -0.5, \theta_1 = 1, \theta_2 = -1, \\ \hat{y} = 0, d = 1$$



x_0	x_1	x_2	y
1	0.1	0	0
1	0.6	1	1
1	0	1	1

$$\begin{aligned}x_2\theta_2 + x_1\theta_1 + x_0\theta_0 &= 0 \\1.6x_1 + 0.5 &= 0 \\x_1 &= -\frac{0.5}{1.6} = -0.3125\end{aligned}$$

$$\theta_0 = 0.5, \theta_1 = 1.6, \theta_2 = 0$$



Perceptron

Initialize weights randomly

foreach x in trainingsset{

$\hat{y} = \text{Prediction}(x)$, $\hat{y} \in \{0, 1\}$

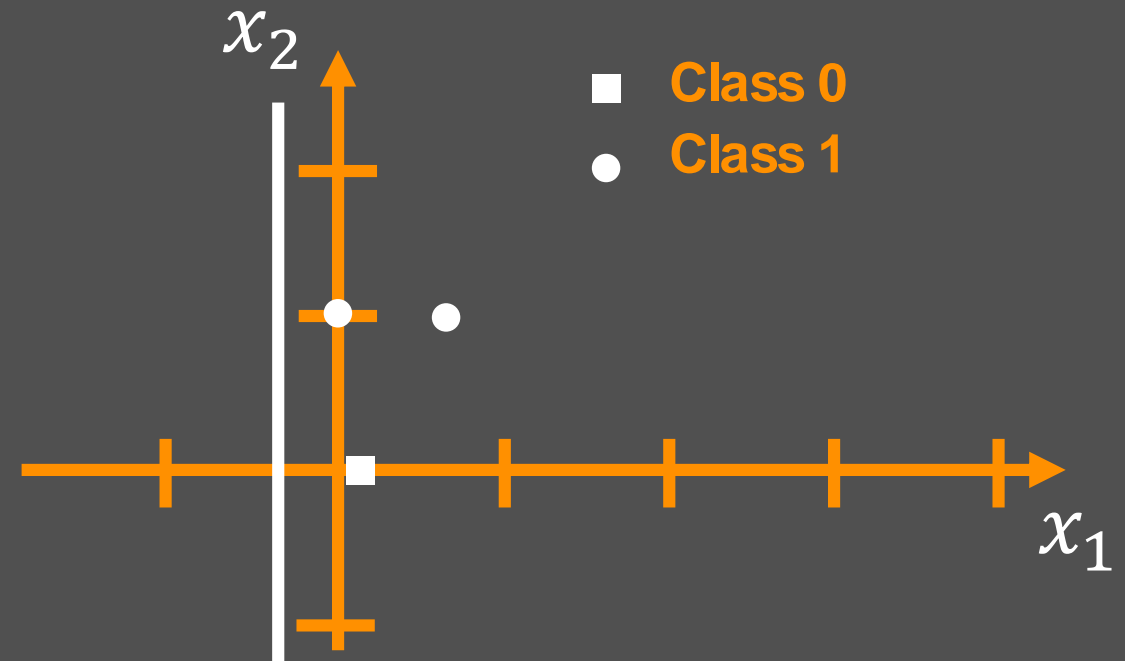
$d = y - \hat{y}$, $y \in \{0, 1\}$

if ($d == 1$) $\theta = \theta + x$

elif ($d == -1$) $\theta = \theta - x$

}

$$\theta_0 = 0.5, \theta_1 = 1.6, \theta_2 = 0$$

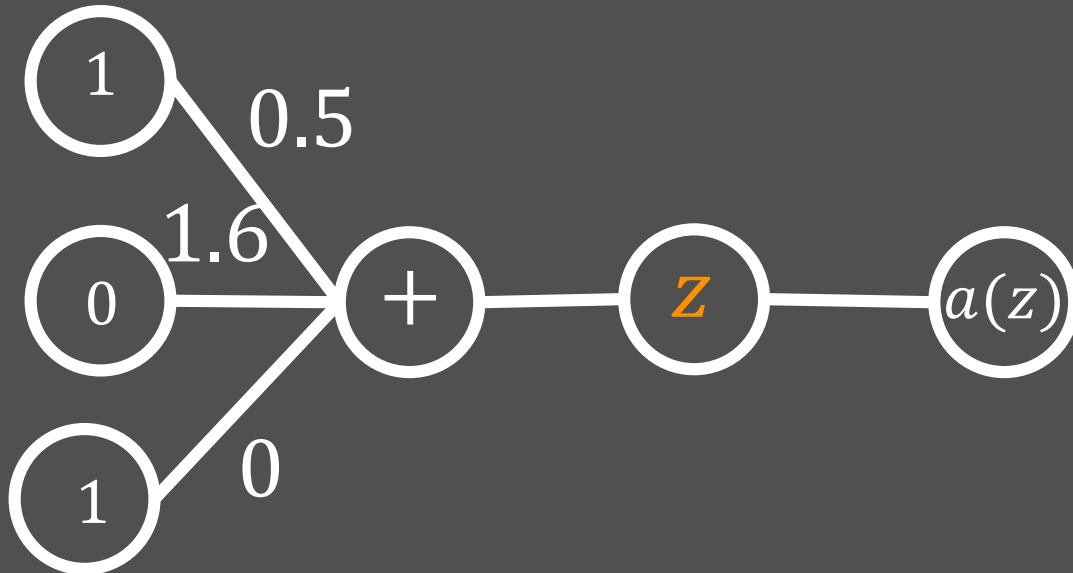


Perceptron

Perceptron-Model

$$z = \theta_1 * x_1 + \theta_0 * x_0$$

$$\theta_0 = 0.5, \theta_1 = 1.6, \theta_2 = 0$$



x_0	x_1	x_2	y
1	0.1	0	0
1	0.6	1	1
1	0	1	1

Perceptron

Perceptron-Model

$$z = \theta_1 * x_1 + \theta_0 * x_0$$

$$\theta_0 = 0.5, \theta_1 = 1.6, \theta_2 = 0$$

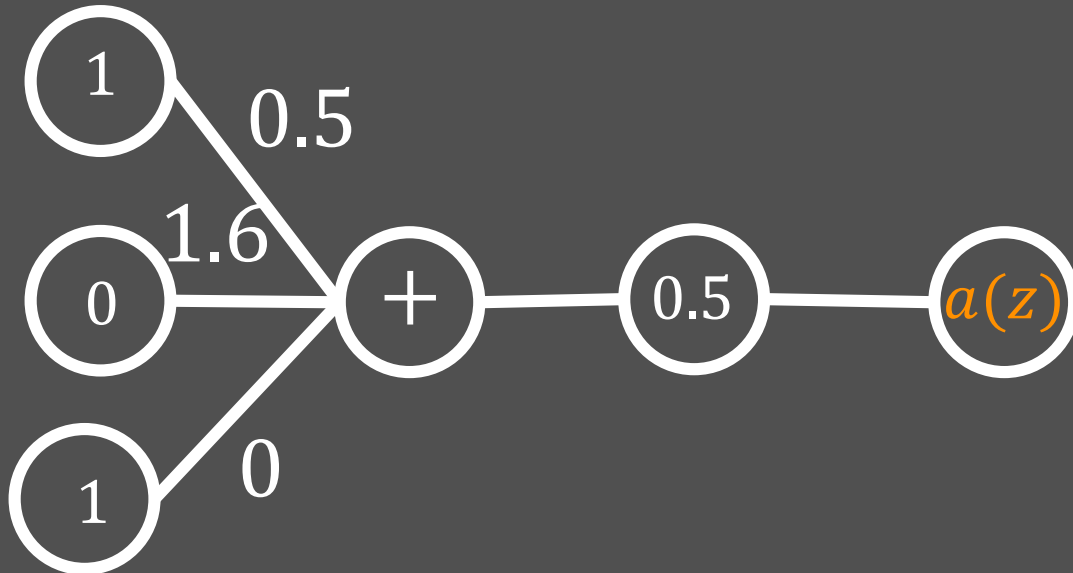


x_0	x_1	x_2	y
1	0.1	0	0
1	0.6	1	1
1	0	1	1

Perceptron

Perceptron-Model

$$a(z) = \begin{cases} 0, & z < 0 \\ 1, & z \geq 0 \end{cases}$$



$$\theta_0 = 0.5, \theta_1 = 1.6, \theta_2 = 0$$

x_0	x_1	x_2	y
1	0.1	0	0
1	0.6	1	1
1	0	1	1

Perceptron

Perceptron-Model

$$\hat{y} = a(z) = \begin{cases} 0, & z < 0 \\ 1, & z \geq 0 \end{cases}$$



$$\theta_0 = 0.5, \theta_1 = 1.6, \theta_2 = 0$$

x_0	x_1	x_2	y
1	0.1	0	0
1	0.6	1	1
1	0	1	1

Perceptron

Initialize weights randomly
foreach x in trainingsset{

$\hat{y} = \text{Prediction}(x), \hat{y} \in \{0, 1\}$

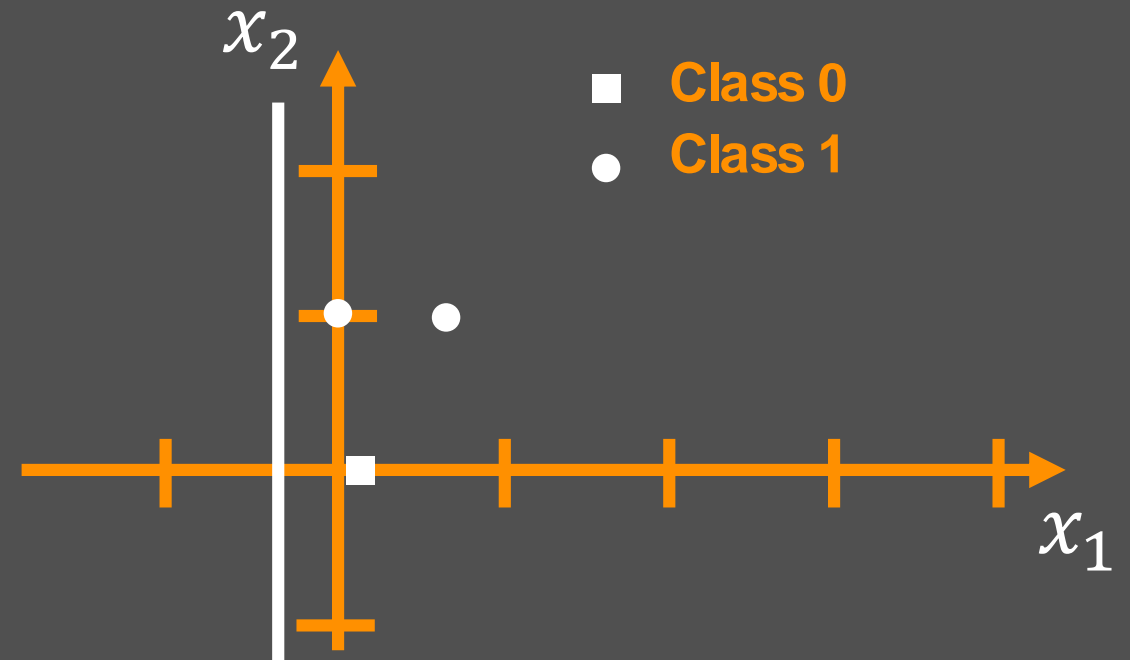
$d = y - \hat{y}, y \in \{0, 1\}$

if ($d == 1$) $\theta = \theta + x$

elif ($d == -1$) $\theta = \theta - x$

}

$\theta_0 = 0.5, \theta_1 = 1.6, \theta_2 = 0,$
 $\hat{y} = 1$



Perceptron

Initialize weights randomly

foreach x in trainingsset{

$\hat{y} = \text{Prediction}(x)$, $\hat{y} \in \{0, 1\}$

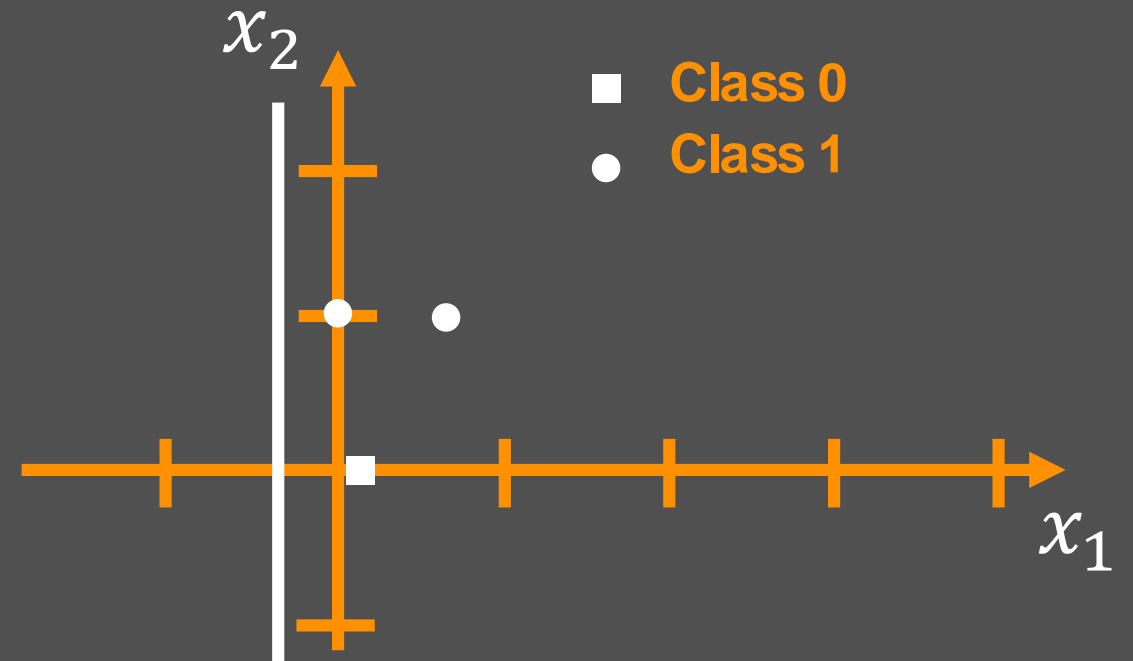
$d = y - \hat{y}$, $y \in \{0, 1\}$

if ($d == 1$) $\theta = \theta + x$

elif ($d == -1$) $\theta = \theta - x$

}

$$\theta_0 = 0.5, \theta_1 = 1.6, \theta_2 = 0, \\ \hat{y} = 1$$



Perceptron

Initialize weights randomly

foreach x in trainingsset{

$\hat{y} = \text{Prediction}(x)$, $\hat{y} \in \{0, 1\}$

$d = y - \hat{y}$, $y \in \{0, 1\}$

if ($d == 1$) $\theta = \theta + x$

elif ($d == -1$) $\theta = \theta - x$

}

$$\theta_0 = 0.5, \theta_1 = 1.6, \theta_2 = 0,$$

$$\hat{y} = 1, d = 1$$

x_0	x_1	x_2	y
1	0.1	0	0
1	0.6	1	1
1	0	1	1



Perceptron

Initialize weights randomly
foreach x in trainingsset{

$\hat{y} = \text{Prediction}(x)$, $\hat{y} \in \{0, 1\}$

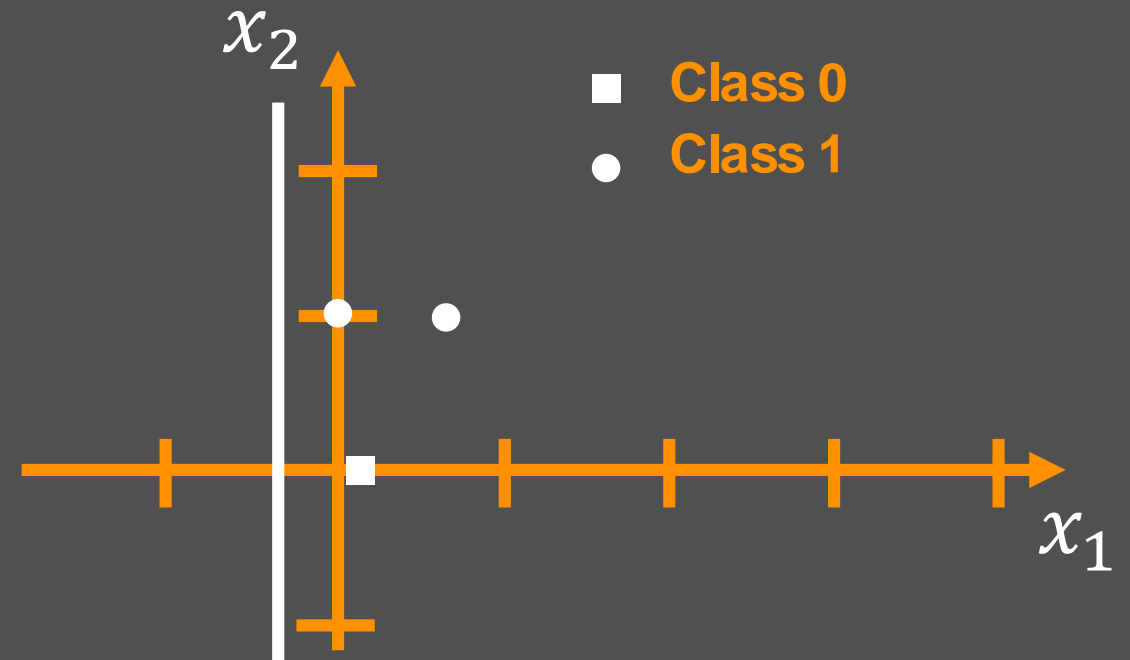
$d = y - \hat{y}$, $y \in \{0, 1\}$

if ($d == 1$) $\theta = \theta + x$

elif ($d == -1$) $\theta = \theta - x$

}

$\theta_0 = 0.5, \theta_1 = 1.6, \theta_2 = 0$,
 $\hat{y} = 1$, $d = 0$

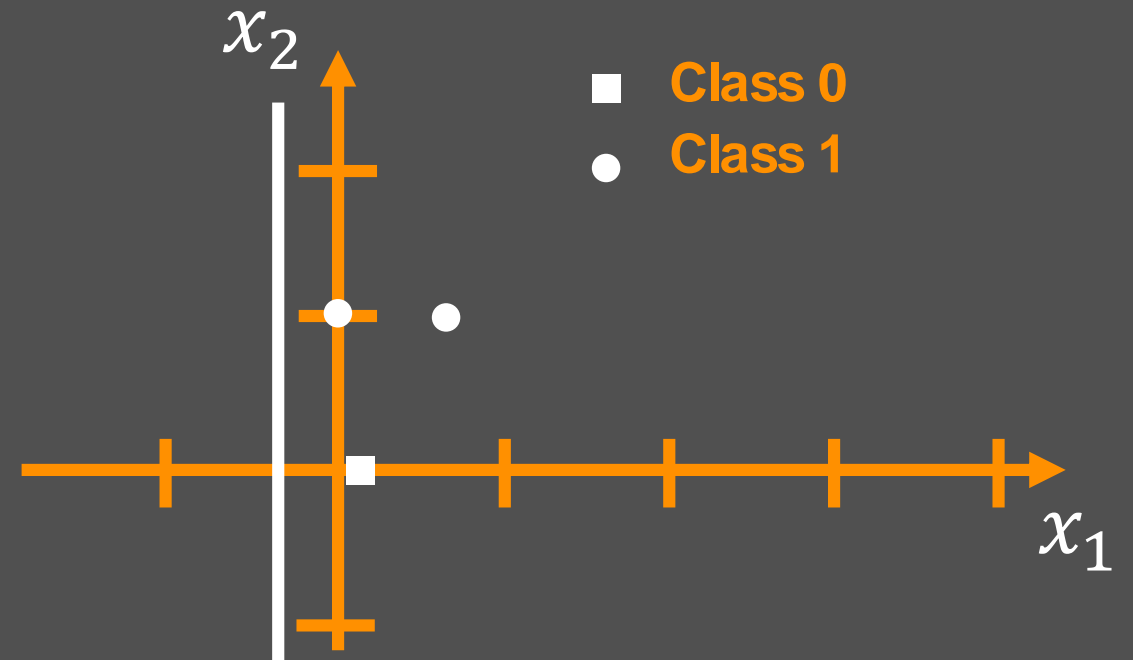


Perceptron

```
Initialize weights randomly
foreach  $x$  in trainingsset{
     $\hat{y} = \text{Prediction}(x)$ ,  $\hat{y} \in \{0, 1\}$ 
     $d = y - \hat{y}$ ,  $y \in \{0, 1\}$ 
    if ( $d == 1$ )  $\theta = \theta + x$ 
    elif ( $d == -1$ )  $\theta = \theta - x$ 
}
```

$$\theta_0 = 0.5, \theta_1 = 1.6, \theta_2 = 0,$$

$$\hat{y} = 1, d = 0$$



Perceptron

Initialize weights randomly

foreach x in trainingsset{

$\hat{y} = \text{Prediction}(x)$, $\hat{y} \in \{0, 1\}$

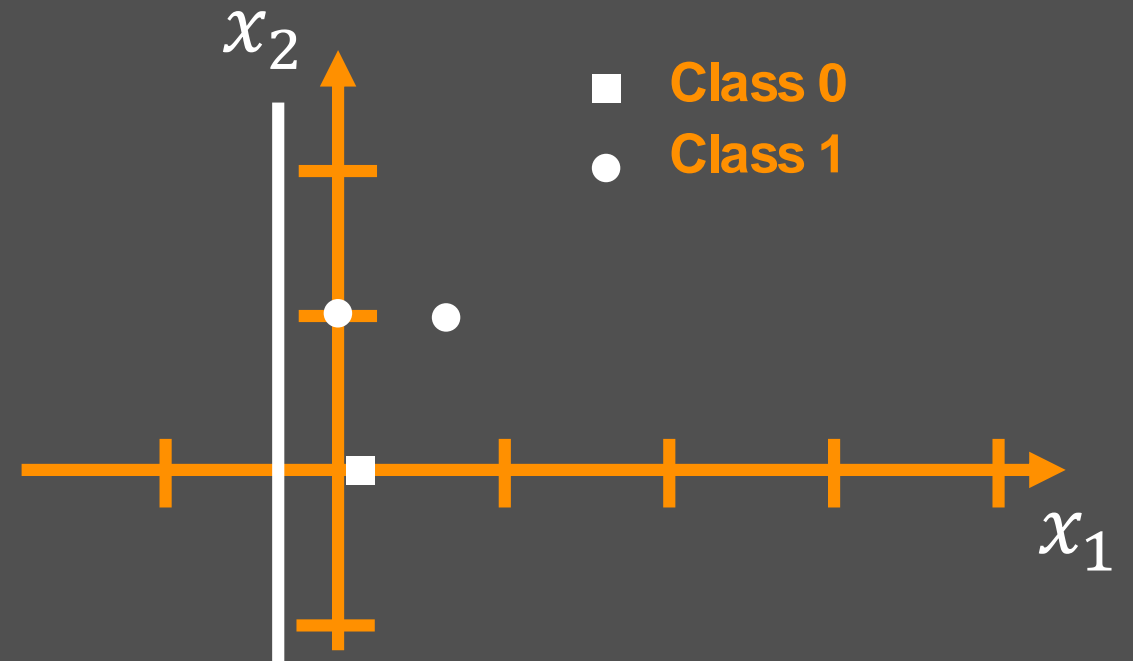
$d = y - \hat{y}$, $y \in \{0, 1\}$

if ($d == 1$) $\theta = \theta + x$

elif ($d == -1$) $\theta = \theta - x$

}

$$\theta_0 = 0.5, \theta_1 = 1.6, \theta_2 = 0$$

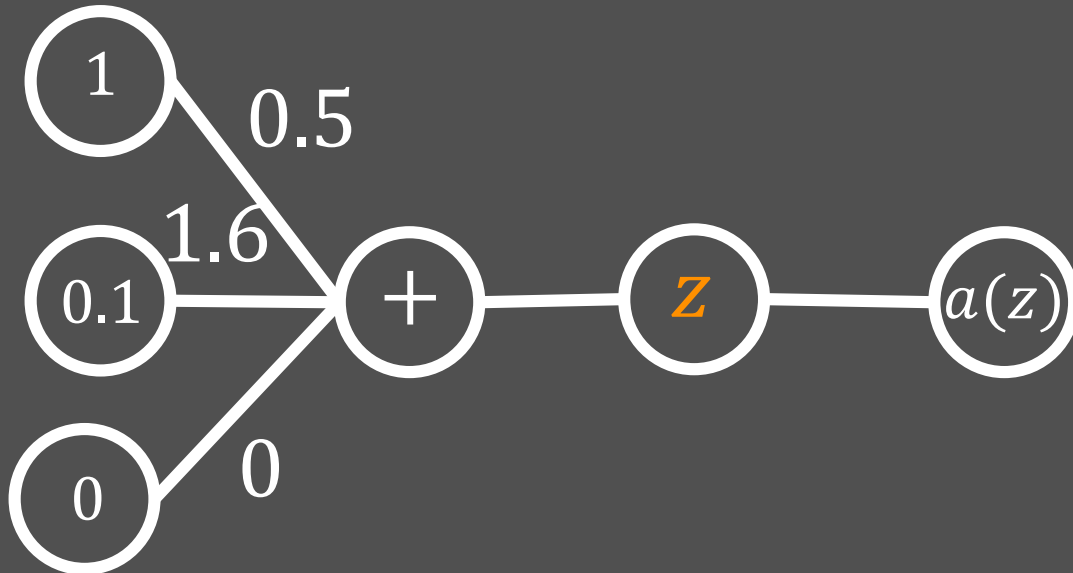


Perceptron

Perceptron-Model

$$z = \theta_1 * x_1 + \theta_0 * x_0$$

$$\theta_0 = 0.5, \theta_1 = 1.6, \theta_2 = 0$$



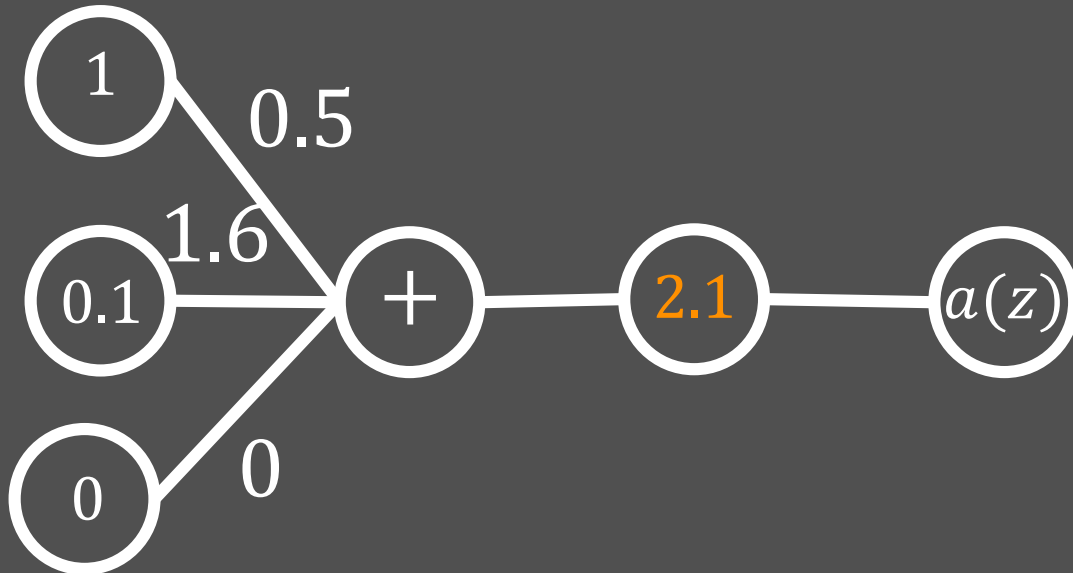
x_0	x_1	x_2	y
1	0.1	0	0
1	0.6	1	1
1	0	1	1

Perceptron

Perceptron-Model

$$z = \theta_1 * x_1 + \theta_0 * x_0$$

$$\theta_0 = 0.5, \theta_1 = 1.6, \theta_2 = 0$$

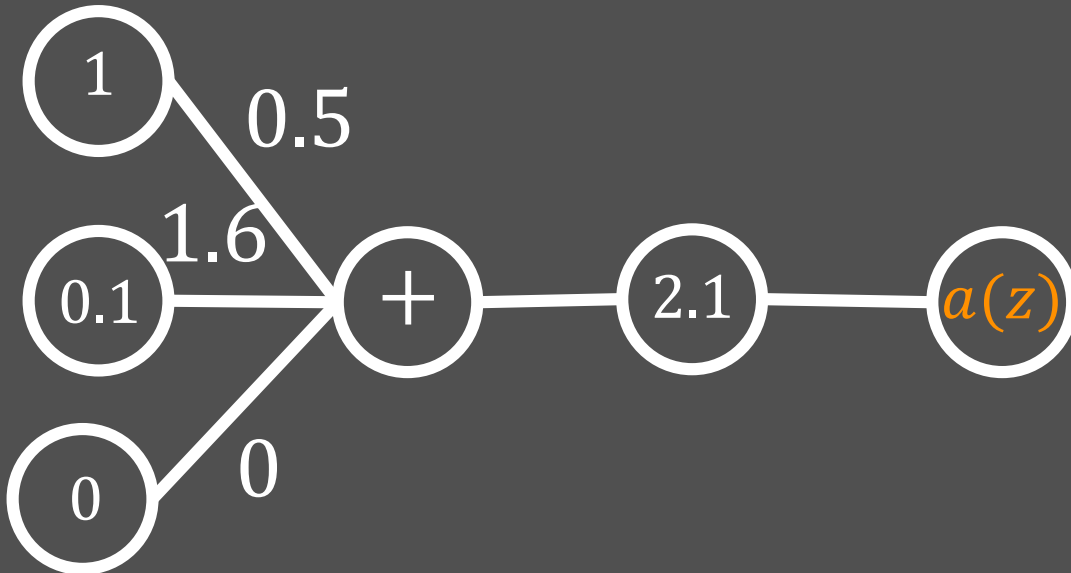


x_0	x_1	x_2	y
1	0.1	0	0
1	0.6	1	1
1	0	1	1

Perceptron

Perceptron-Model

$$a(z) = \begin{cases} 0, & z < 0 \\ 1, & z \geq 0 \end{cases}$$



$$\theta_0 = 0.5, \theta_1 = 1.6, \theta_2 = 0$$

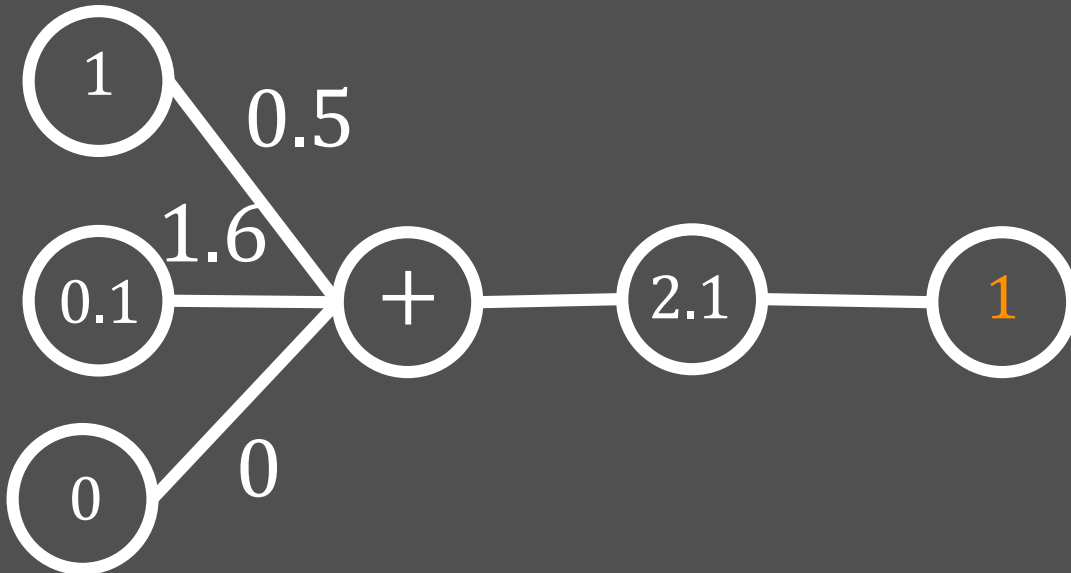


x_0	x_1	x_2	y
1	0.1	0	0
1	0.6	1	1
1	0	1	1

Perceptron

Perceptron-Model

$$\hat{y} = a(z) = \begin{cases} 0, & z < 0 \\ 1, & z \geq 0 \end{cases}$$



$$\theta_0 = 0.5, \theta_1 = 1.6, \theta_2 = 0$$

x_0	x_1	x_2	y
1	0.1	0	0
1	0.6	1	1
1	0	1	1

Perceptron

Initialize weights randomly
foreach x in trainingsset{

$\hat{y} = \text{Prediction}(x), \hat{y} \in \{0, 1\}$

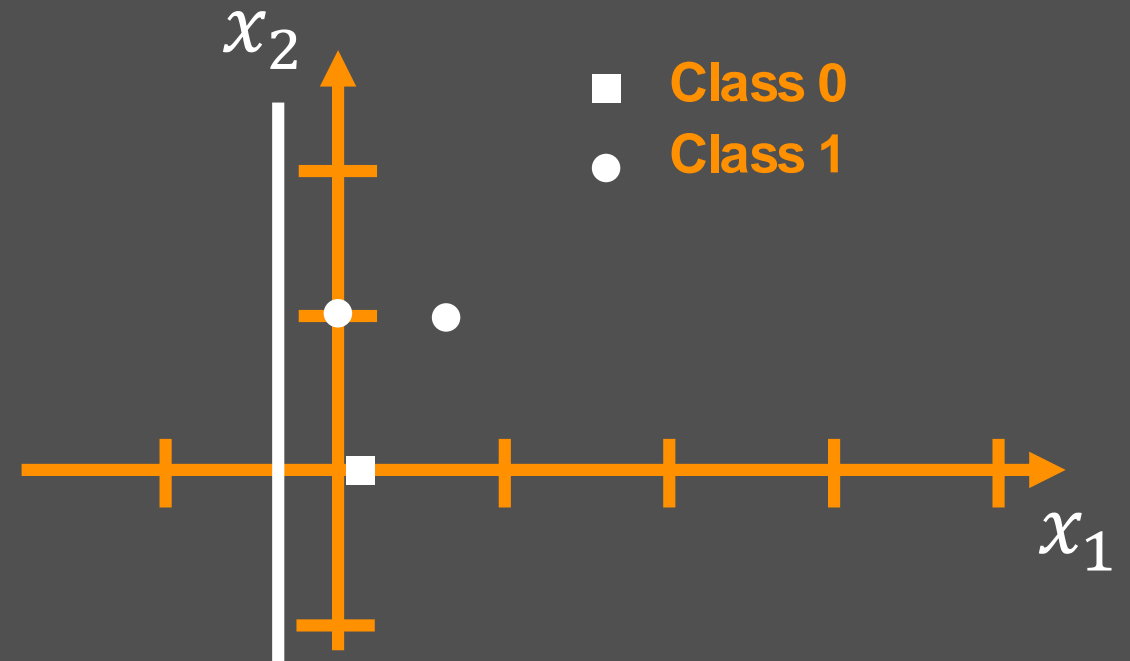
$d = y - \hat{y}, y \in \{0, 1\}$

if ($d == 1$) $\theta = \theta + x$

elif ($d == -1$) $\theta = \theta - x$

}

$\theta_0 = 0.5, \theta_1 = 1.6, \theta_2 = 0,$
 $\hat{y} = 1$



Perceptron

Initialize weights randomly

foreach x in trainingsset{

$\hat{y} = \text{Prediction}(x)$, $\hat{y} \in \{0, 1\}$

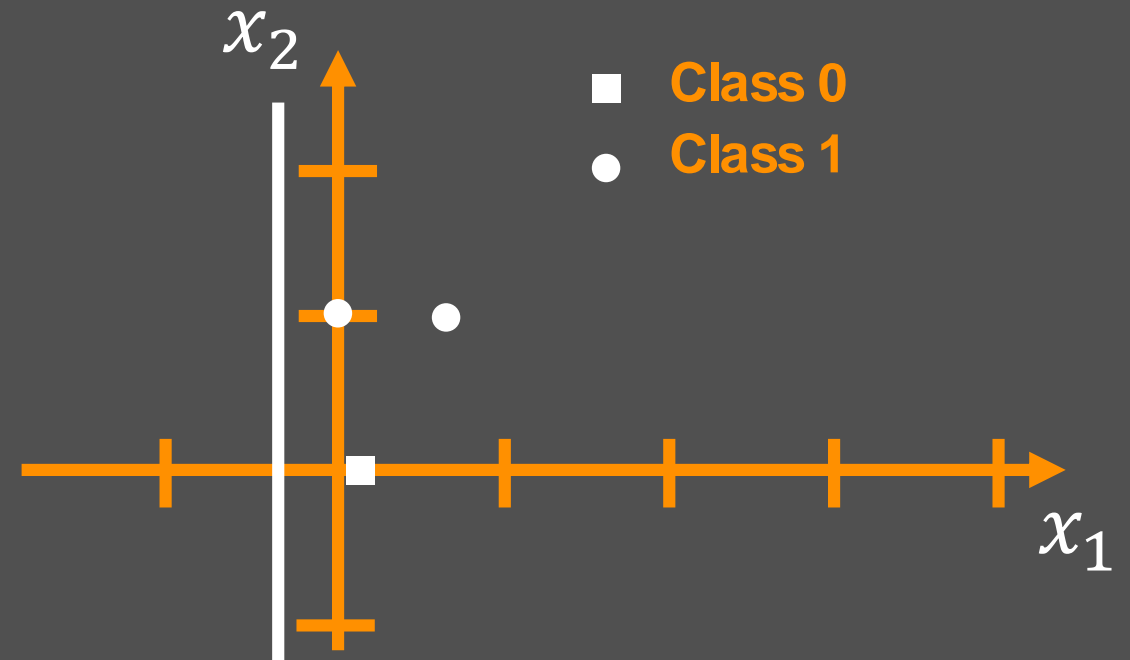
$d = y - \hat{y}$, $y \in \{0, 1\}$

if ($d == 1$) $\theta = \theta + x$

elif ($d == -1$) $\theta = \theta - x$

}

$$\theta_0 = 0.5, \theta_1 = 1.6, \theta_2 = 0, \\ \hat{y} = 1$$



Perceptron

Initialize weights randomly

foreach x in trainingsset{

$\hat{y} = \text{Prediction}(x)$, $\hat{y} \in \{0, 1\}$

$d = y - \hat{y}$, $y \in \{0, 1\}$

if ($d == 1$) $\theta = \theta + x$

elif ($d == -1$) $\theta = \theta - x$

}

$$\theta_0 = 0.5, \theta_1 = 1.6, \theta_2 = 0, \\ \hat{y} = 1, d = -1$$



x_0	x_1	x_2	y
1	0.1	0	0
1	0.6	1	1
1	0	1	1

Perceptron

Initialize weights randomly
foreach x in trainingsset{

$\hat{y} = \text{Prediction}(x)$, $\hat{y} \in \{0, 1\}$

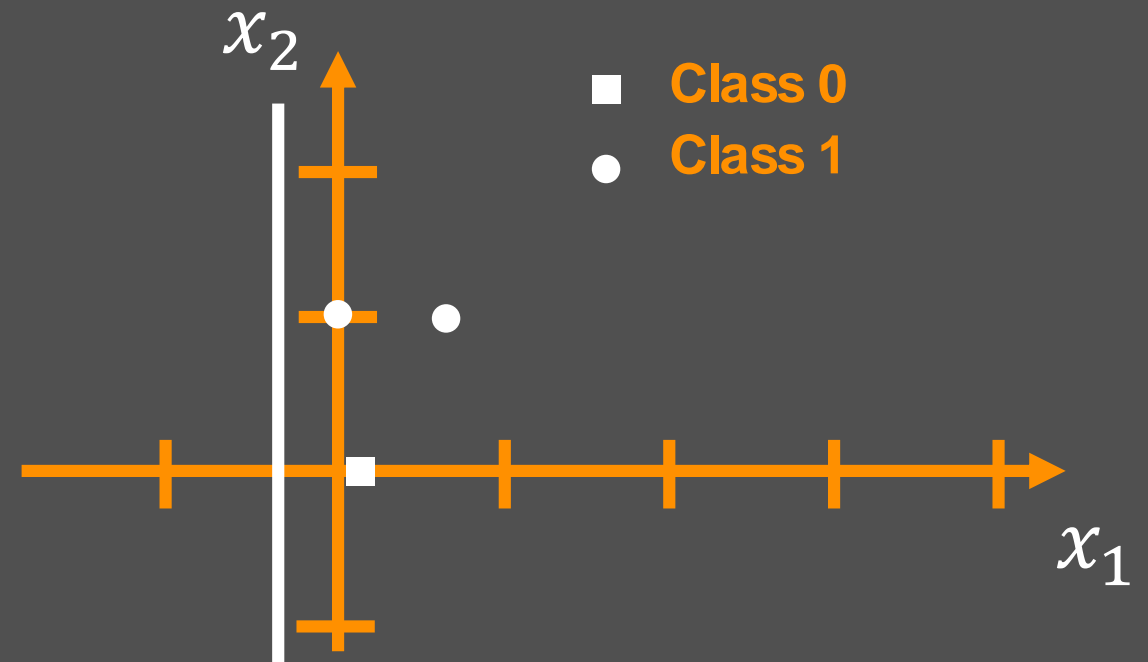
$d = y - \hat{y}$, $y \in \{0, 1\}$

if ($d == 1$) $\theta = \theta + x$

elif ($d == -1$) $\theta = \theta - x$

}

$\theta_0 = 0.5, \theta_1 = 1.6, \theta_2 = 0$,
 $\hat{y} = 1$, $d = -1$



Perceptron

Initialize weights randomly

foreach x in trainingsset{

$\hat{y} = \text{Prediction}(x)$, $\hat{y} \in \{0, 1\}$

$d = y - \hat{y}$, $y \in \{0, 1\}$

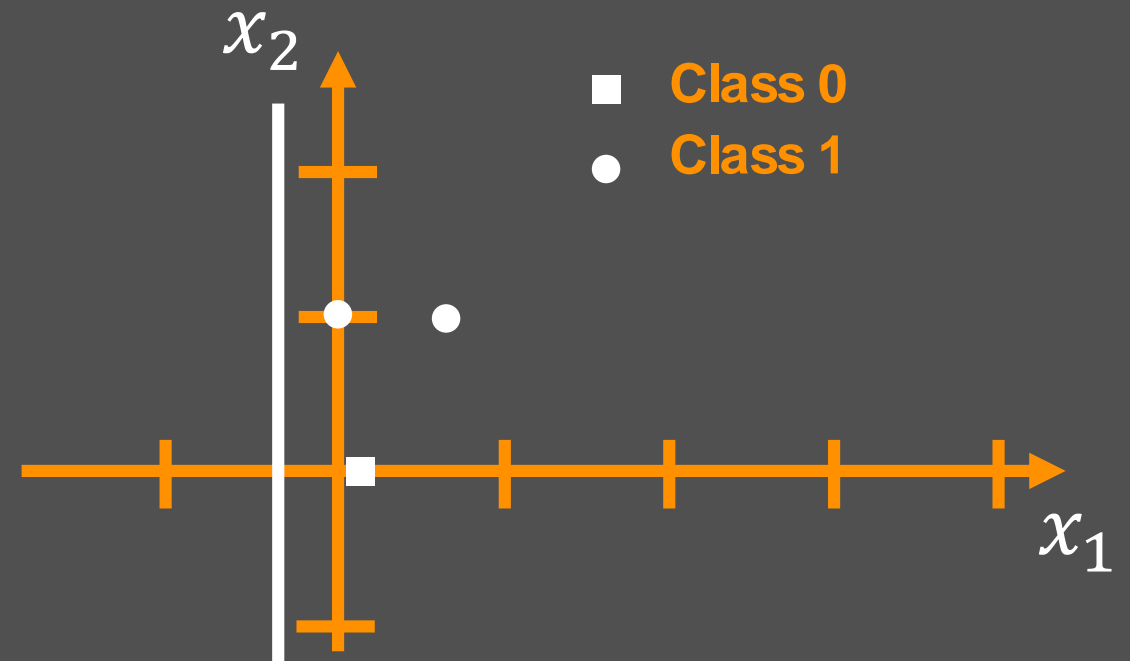
if($d == 1$) $\theta = \theta + x$

elif ($d == -1$) $\theta = \theta - x$

}

$$\theta_0 = 0.5, \theta_1 = 1.6, \theta_2 = 0,$$

$$\hat{y} = 1, d = -1$$



Perceptron

Initialize weights randomly
foreach x in trainingsset{

$\hat{y} = \text{Prediction}(x)$, $\hat{y} \in \{0, 1\}$

$d = y - \hat{y}$, $y \in \{0, 1\}$

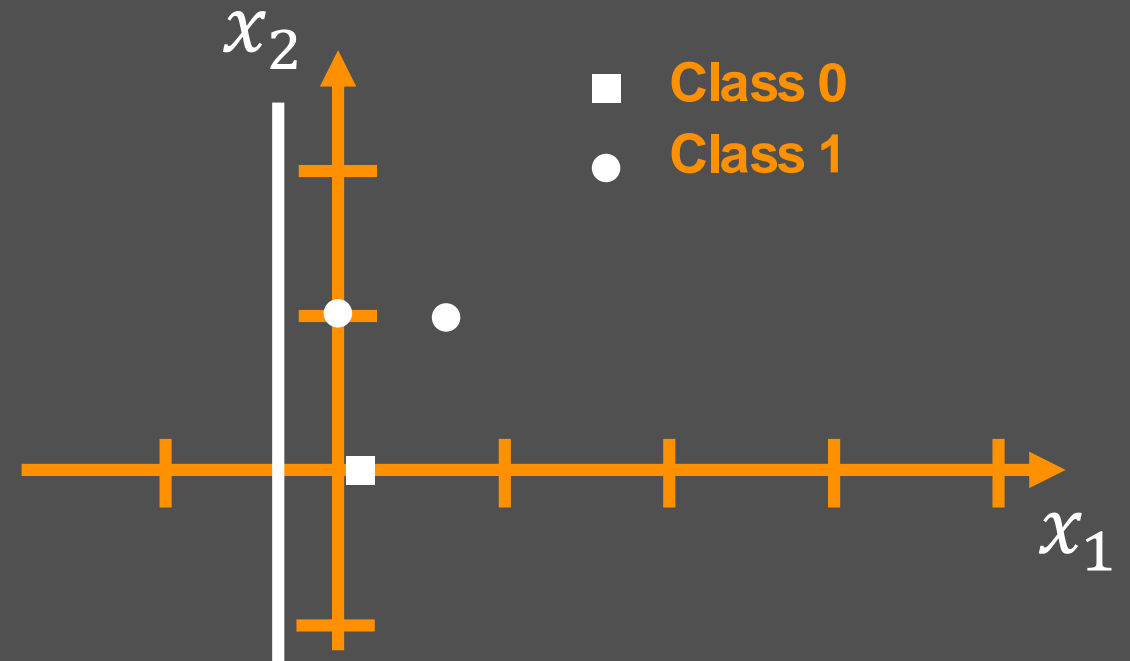
if ($d == 1$) $\theta = \theta + x$

elif ($d == -1$) $\theta = \theta - x$

}

$$\theta_0 = 0.5, \theta_1 = 1.6, \theta_2 = 0,$$

$$\hat{y} = 1, d = -1$$



if($d == -1$) $\theta = \theta - x$

$$\theta = \begin{pmatrix} 0.5 \\ 1.6 \\ 0 \end{pmatrix} - \begin{pmatrix} 1 \\ 0.1 \\ 0 \end{pmatrix}$$

$$\theta = \begin{pmatrix} -0.5 \\ 1.5 \\ 0 \end{pmatrix}$$

$$\theta_0 = 0.5, \theta_1 = 1.6, \theta_2 = 0, \\ \hat{y} = 0, d = -1$$



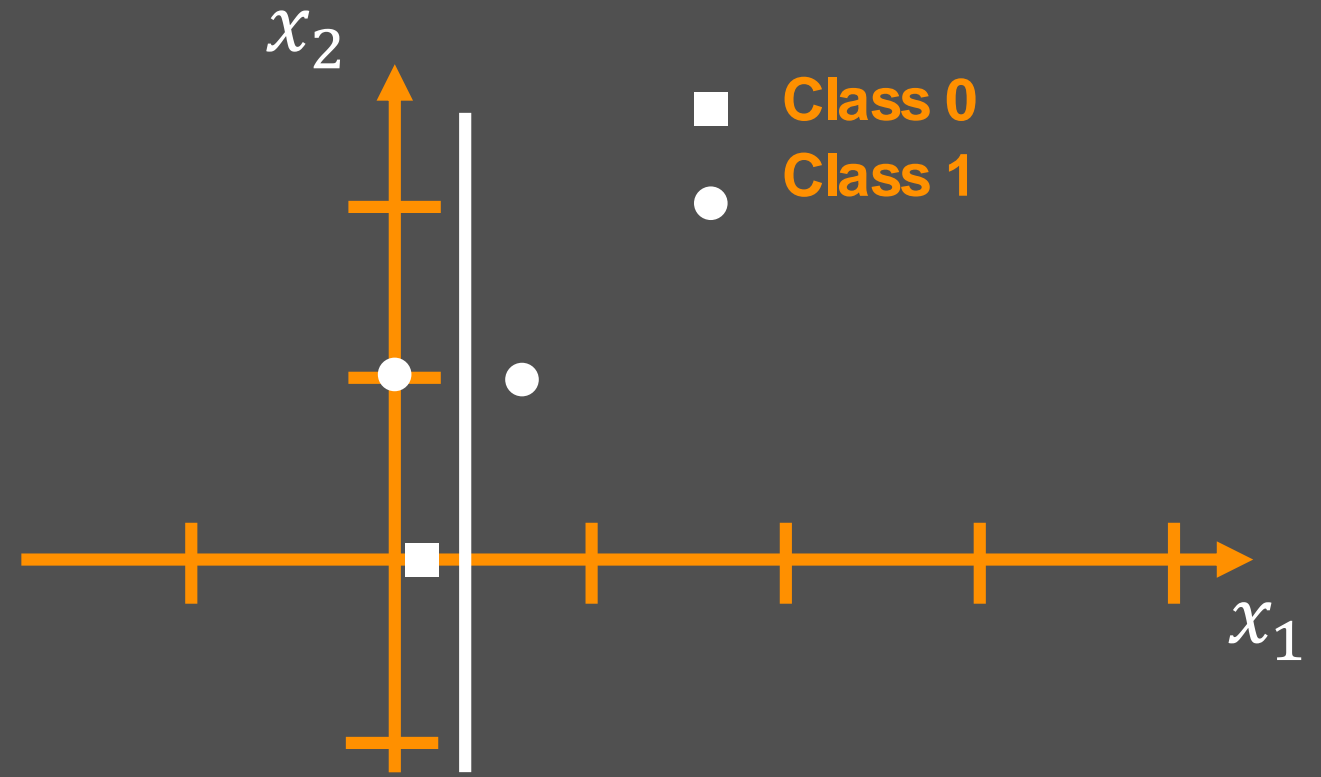
x_0	x_1	x_2	y
1	0.1	0	0
1	0.6	1	1
1	0	1	1

$$x_2\theta_2 + x_1\theta_1 + x_0\theta_0 = 0$$

$$1.5x_1 - 0.5 = 0$$

$$x_1 = \frac{0.5}{1.5} = \frac{1}{3}$$

$$\theta_0 = -0.5, \theta_1 = 1.5, \theta_2 = 0$$



Two iterations later...

Perceptron

Initialize weights randomly

foreach x in trainingsset{

$\hat{y} = \text{Prediction}(x)$, $\hat{y} \in \{0, 1\}$

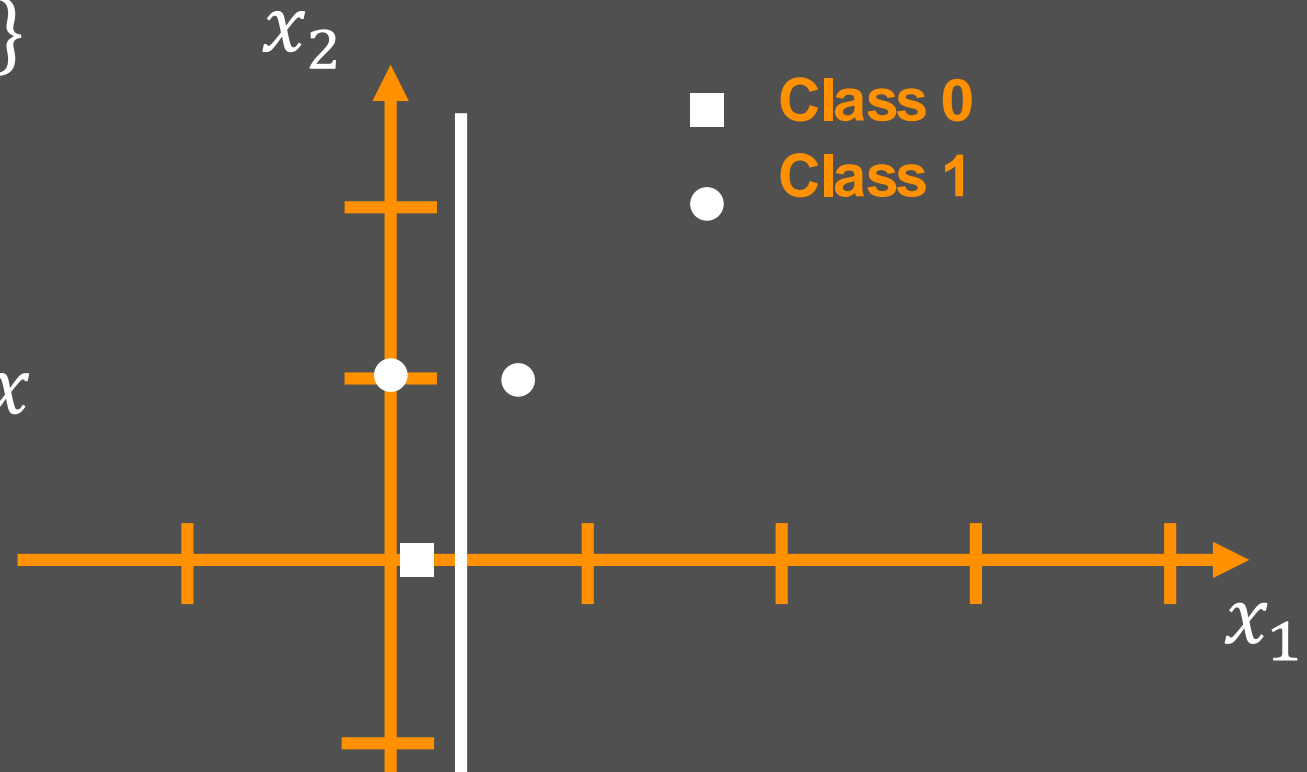
$d = y - \hat{y}$, $y \in \{0, 1\}$

if ($d == 1$) $\theta = \theta + x$

elif ($d == -1$) $\theta = \theta - x$

}

$$\theta_0 = -0.5, \theta_1 = 1.5, \theta_2 = 0, \\ \hat{y} = 0, d = 1$$



if($d==1$) $\theta = \theta + x$

$$\theta = \begin{pmatrix} 0.5 \\ 1.5 \\ 0 \end{pmatrix} + \begin{pmatrix} 1 \\ 0 \\ 1 \end{pmatrix}$$

$$\theta = \begin{pmatrix} 1.5 \\ 1.5 \\ 1 \end{pmatrix}$$

$$\theta_0 = -0.5, \theta_1 = 1.5, \theta_2 = 0, \\ \hat{y} = 0, d = 1$$



x_0	x_1	x_2	y
1	0.1	0	0
1	0.6	1	1
1	0	1	1

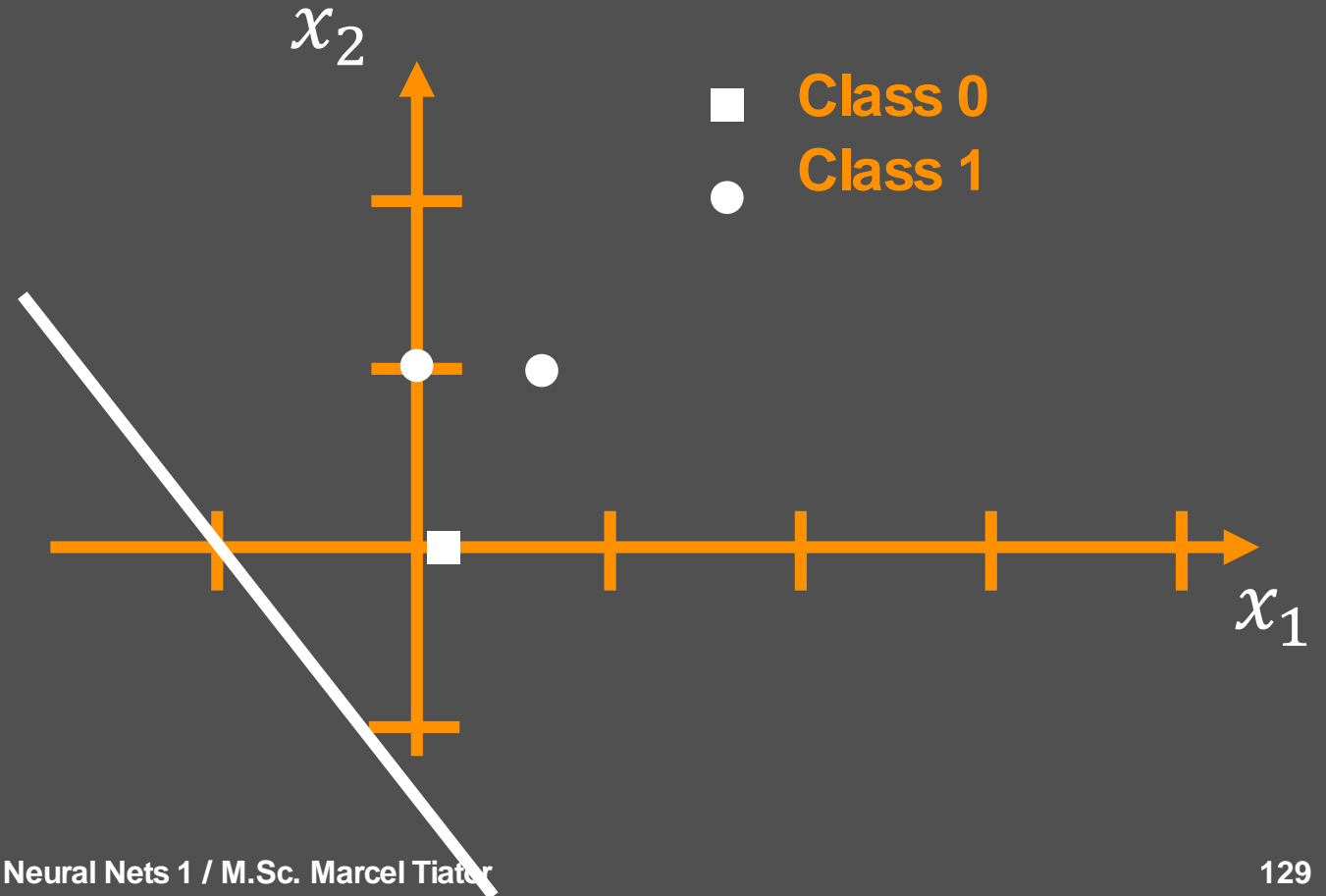
$$x_2\theta_2 + x_1\theta_1 + x_0\theta_0 = 0$$

$$x_2 = \frac{-(x_1\theta_1 + x_0\theta_0)}{\theta_2}$$

$$x_2 = \frac{-(1.5x_1 + 1.5)}{1}$$

$$x_2 = -1.5(x_1 + 1)$$

$$\theta_0 = 1.5, \theta_1 = 1.5, \theta_2 = 1$$



One iteration later...

Perceptron

Initialize weights randomly

foreach x in trainingsset{

$\hat{y} = \text{Prediction}(x)$, $\hat{y} \in \{0, 1\}$

$d = y - \hat{y}$, $y \in \{0, 1\}$

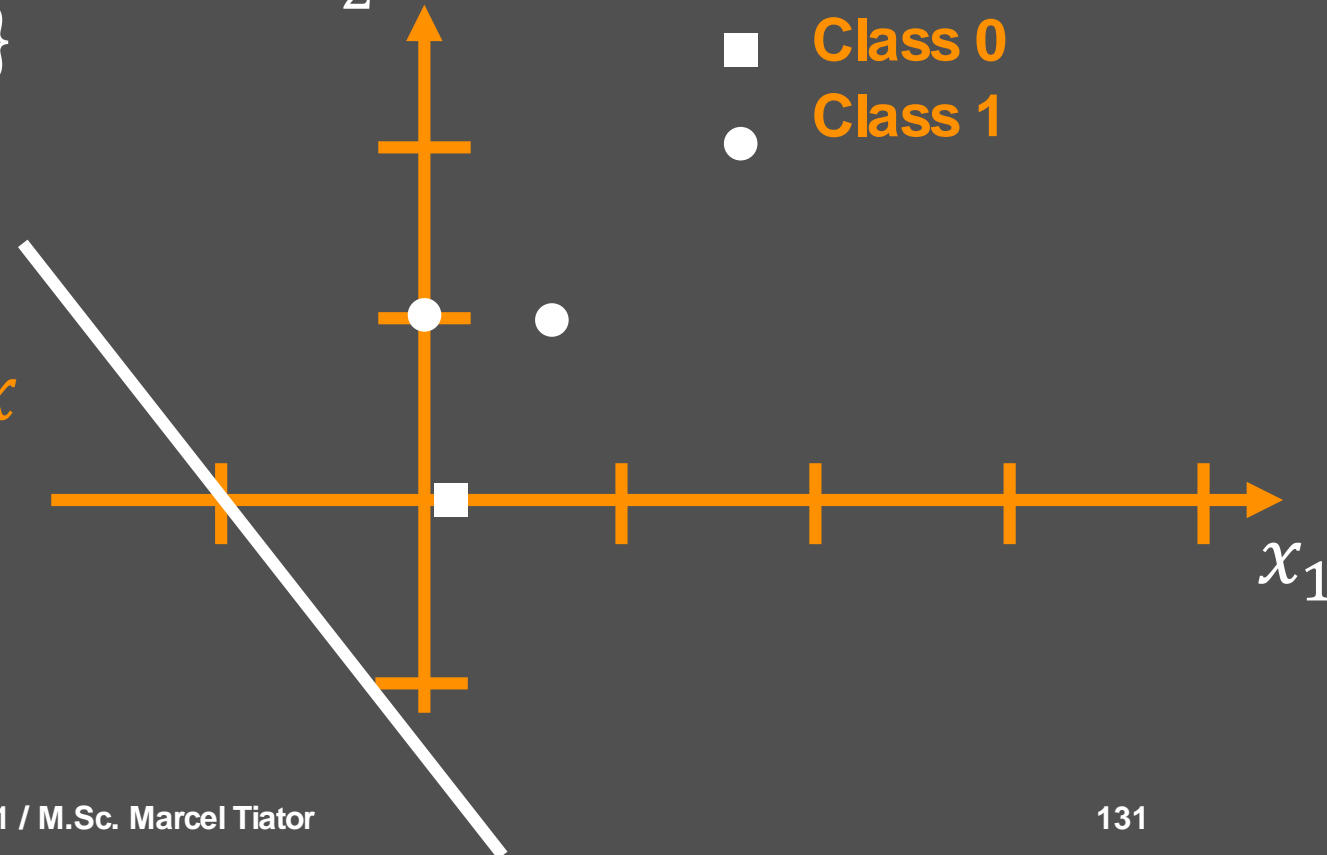
if ($d == 1$) $\theta = \theta + x$

elif ($d == -1$) $\theta = \theta - x$

}

$$\theta_0 = 1.5, \theta_1 = 1.5, \theta_2 = 1,$$

$$\hat{y} = 1, d = -1$$



if($d == -1$) $\theta = \theta - x$

$$\theta = \begin{pmatrix} 1.5 \\ 1.5 \\ 1 \end{pmatrix} - \begin{pmatrix} 1 \\ 0.1 \\ 0 \end{pmatrix}$$

$$\theta = \begin{pmatrix} 0.5 \\ 1.4 \\ 1 \end{pmatrix}$$

$$\theta_0 = 1.5, \theta_1 = 1.5, \theta_2 = 1, \\ \hat{y} = 1, d = -1$$



x_0	x_1	x_2	y
1	0.1	0	0
1	0.6	1	1
1	0	1	1

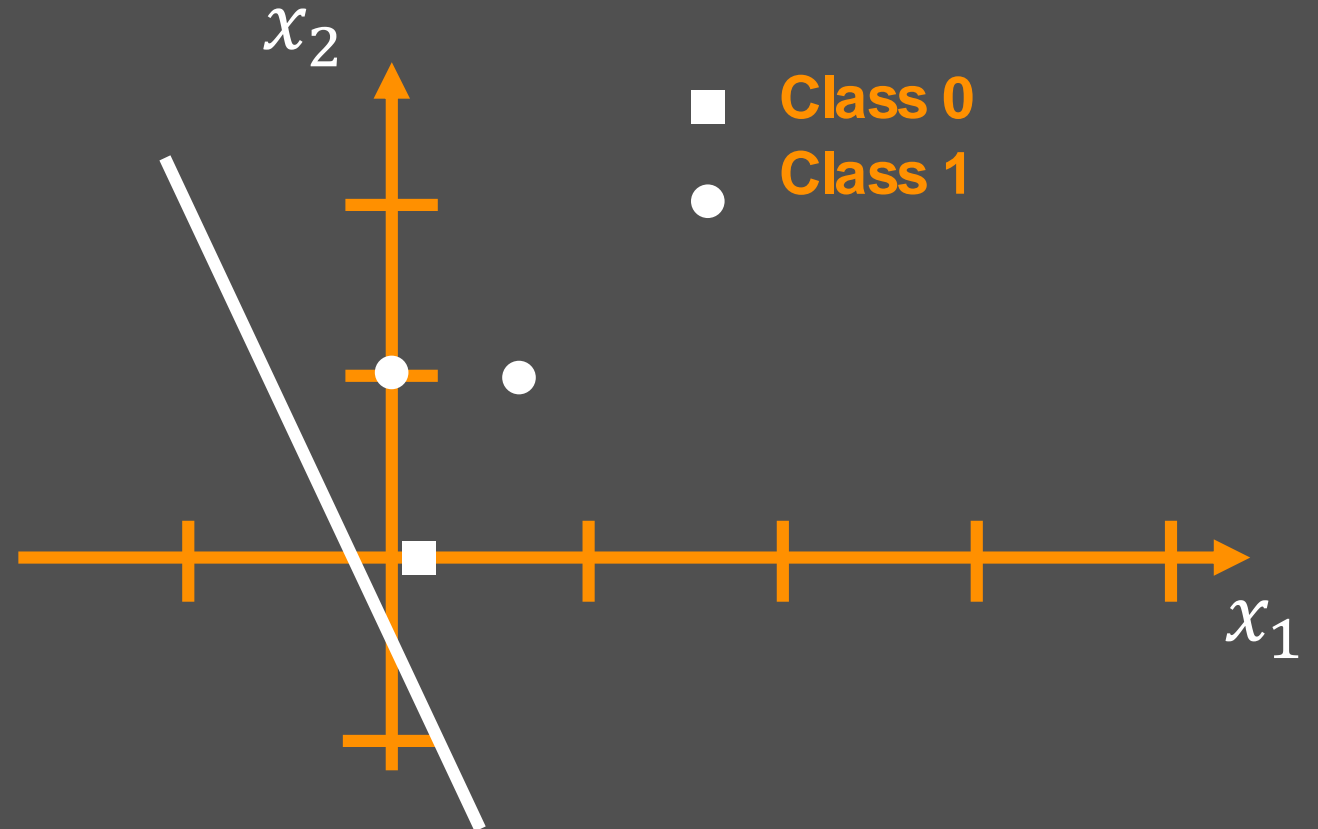
$$x_2\theta_2 + x_1\theta_1 + x_0\theta_0 = 0$$

$$x_2 = \frac{-(x_1\theta_1 + x_0\theta_0)}{\theta_2}$$

$$x_2 = \frac{-(1.4x_1 + 0.5)}{1}$$

$$x_2 = -1.4(x_1 + 0.357)$$

$$\theta_0 = 0.5, \theta_1 = 1.4, \theta_2 = 1$$



Two iterations later...

Perceptron

Initialize weights randomly

foreach x in trainingsset{

$\hat{y} = \text{Prediction}(x)$, $\hat{y} \in \{0, 1\}$

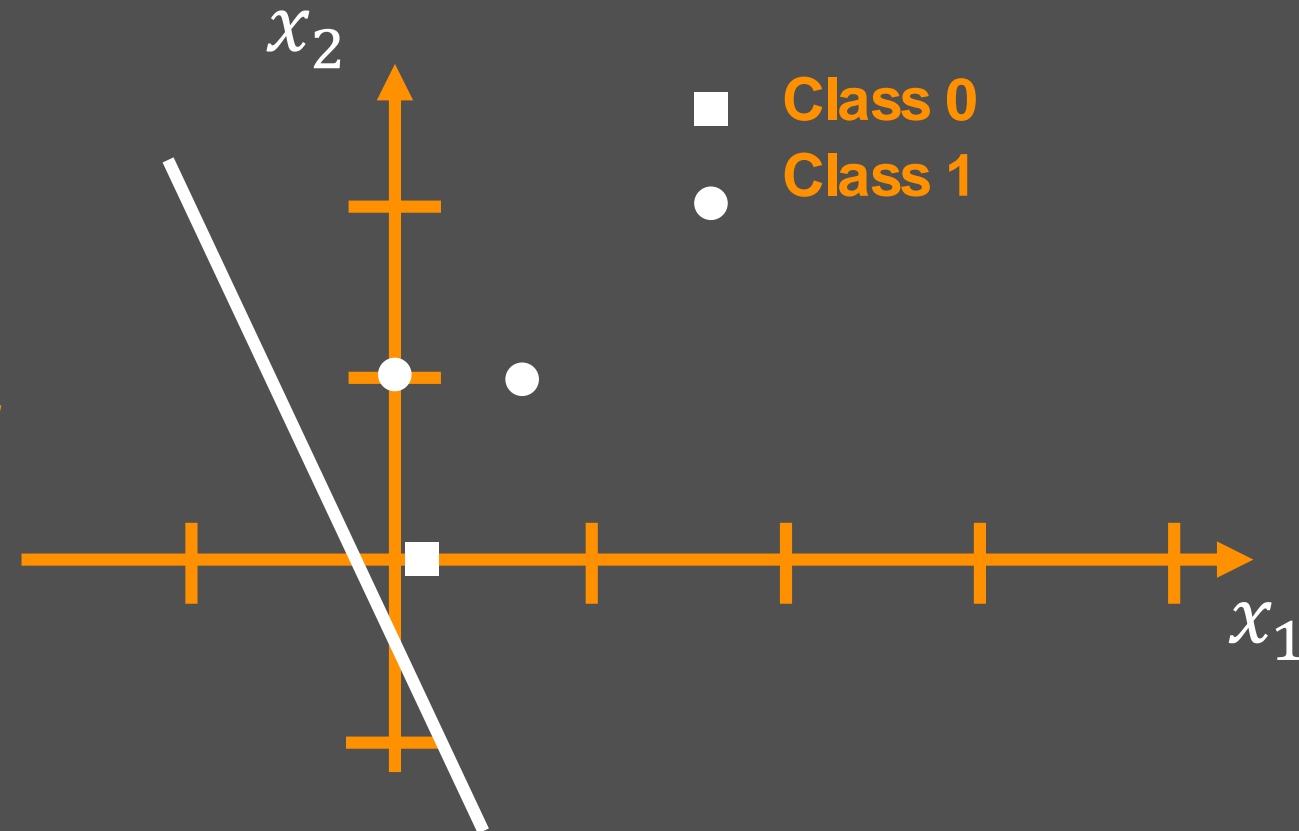
$d = y - \hat{y}$, $y \in \{0, 1\}$

if ($d == 1$) $\theta = \theta + x$

elif ($d == -1$) $\theta = \theta - x$

}

$\theta_0 = 0.5, \theta_1 = 1.4, \theta_2 = 1$,
 $\hat{y} = 1$, $d = -1$



if($d == -1$) $\theta = \theta - x$

$$\theta = \begin{pmatrix} 0.5 \\ 1.4 \\ 1 \end{pmatrix} - \begin{pmatrix} 1 \\ 0.1 \\ 0 \end{pmatrix}$$

$$\theta = \begin{pmatrix} -0.5 \\ 1.3 \\ 1 \end{pmatrix}$$

$$\theta_0 = 0.5, \theta_1 = 1.4, \theta_2 = 1, \\ \hat{y} = 1, d = -1$$



x_0	x_1	x_2	y
1	0.1	0	0
1	0.6	1	1
1	0	1	1

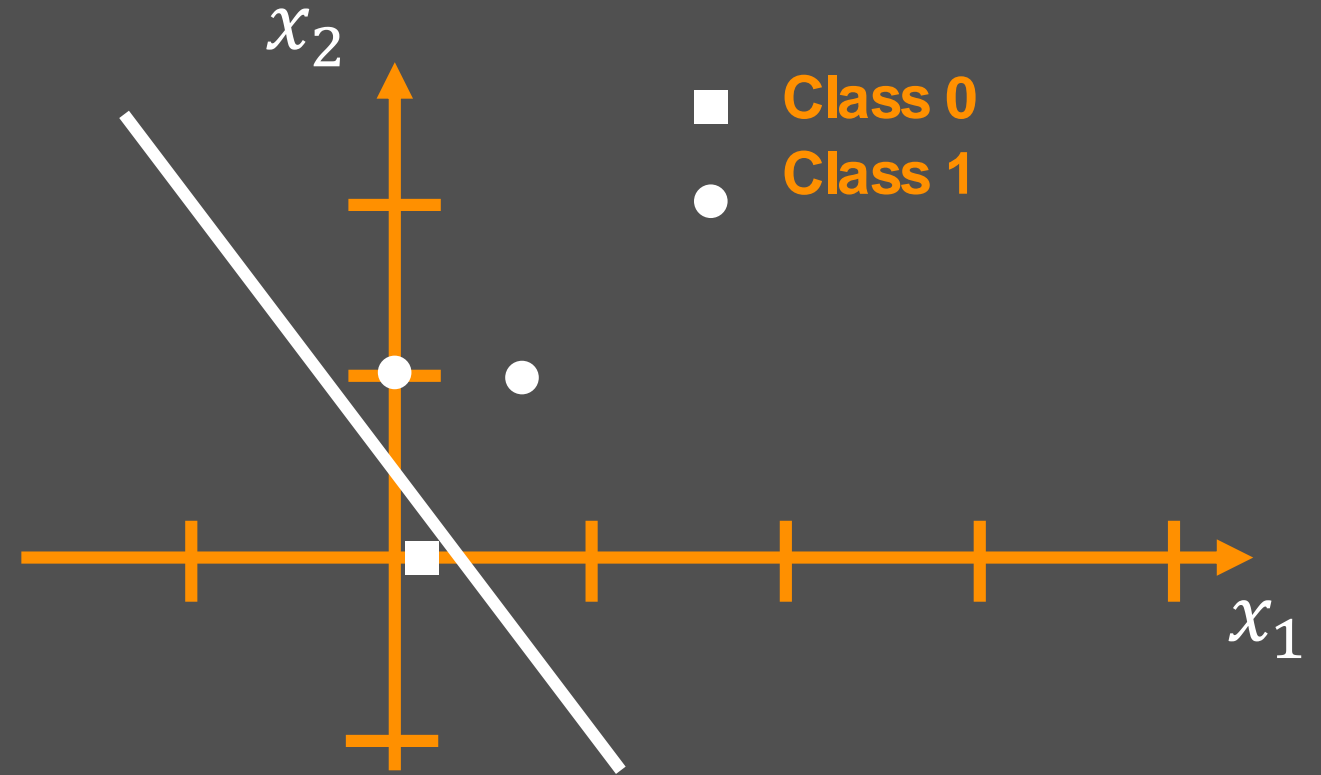
$$x_2\theta_2 + x_1\theta_1 + x_0\theta_0 = 0$$

$$x_2 = \frac{-(x_1\theta_1 + x_0\theta_0)}{\theta_2}$$

$$x_2 = \frac{-(1.3x_1 - 0.5)}{1}$$

$$x_2 = -1.3(x_1 - 0.385)$$

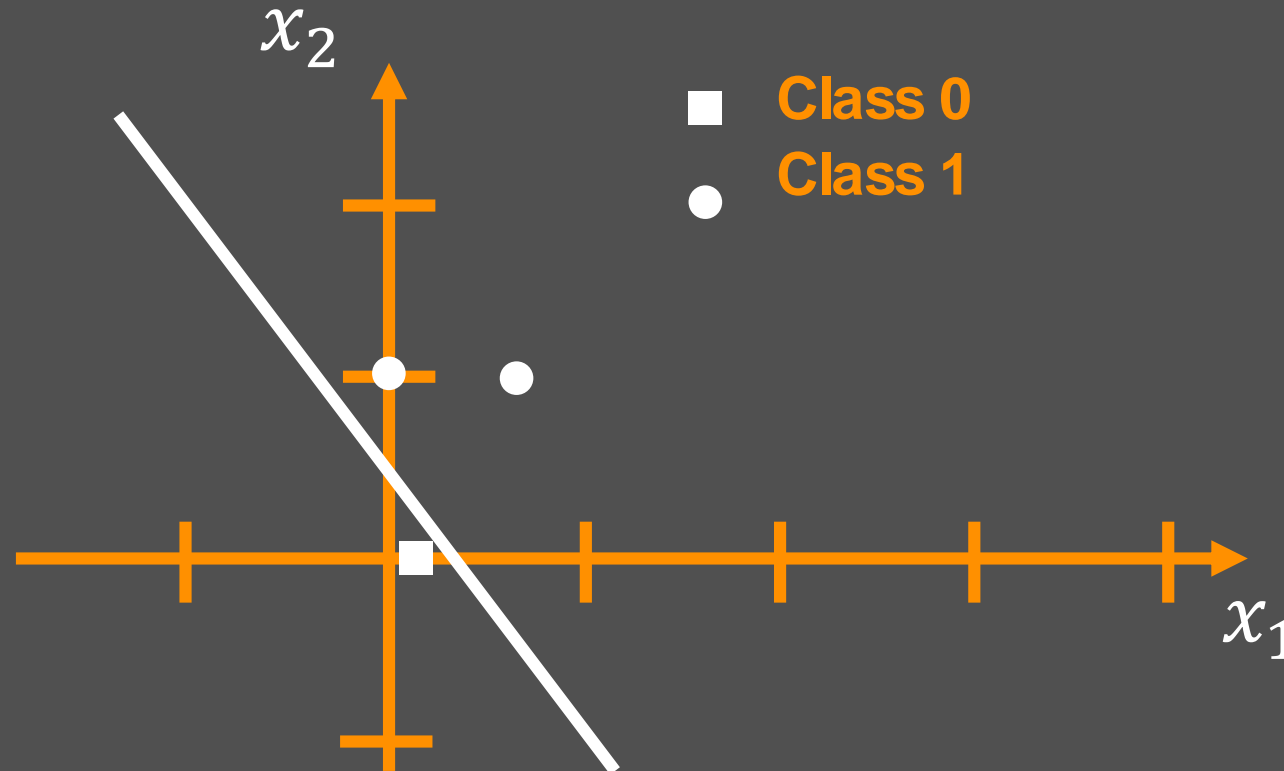
$$\theta_0 = -0.5, \theta_1 = 1.3, \theta_2 = 1$$



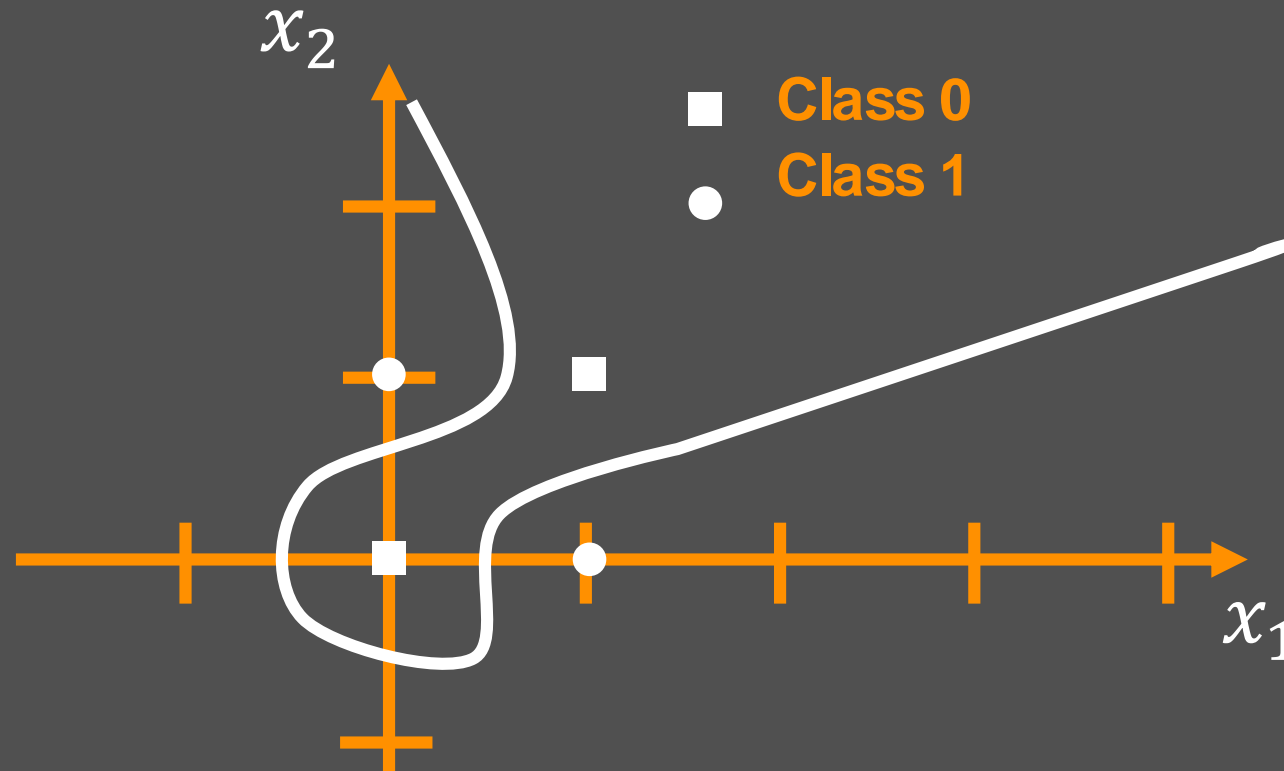
Training Finished!

Binary Classification with Perceptron

- Need linearly separable dataset



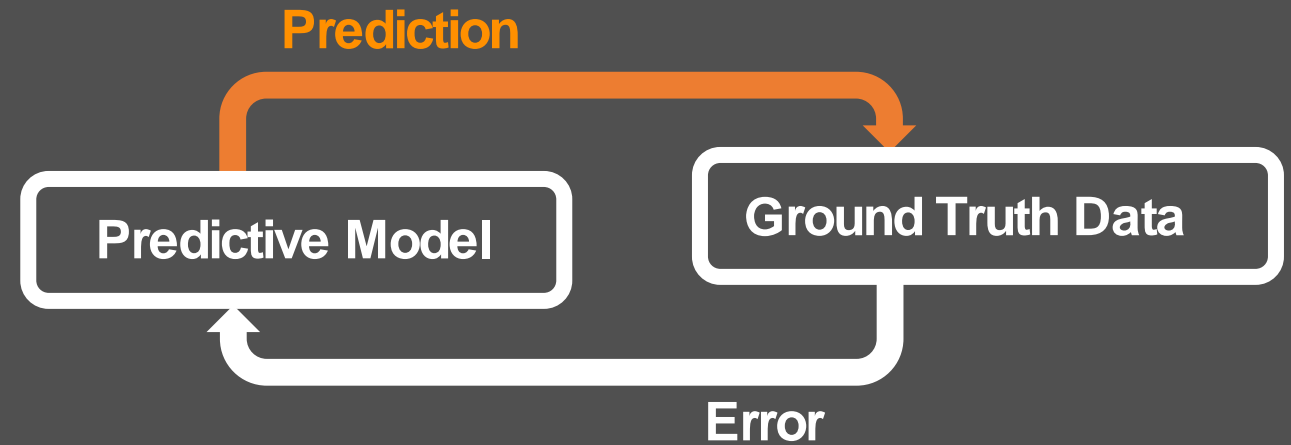
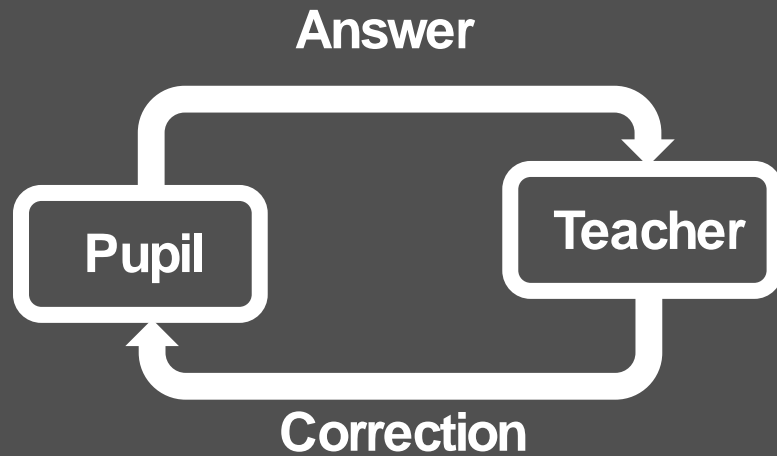
Assume a more complex problem



Multilayer Perceptron (MLP)

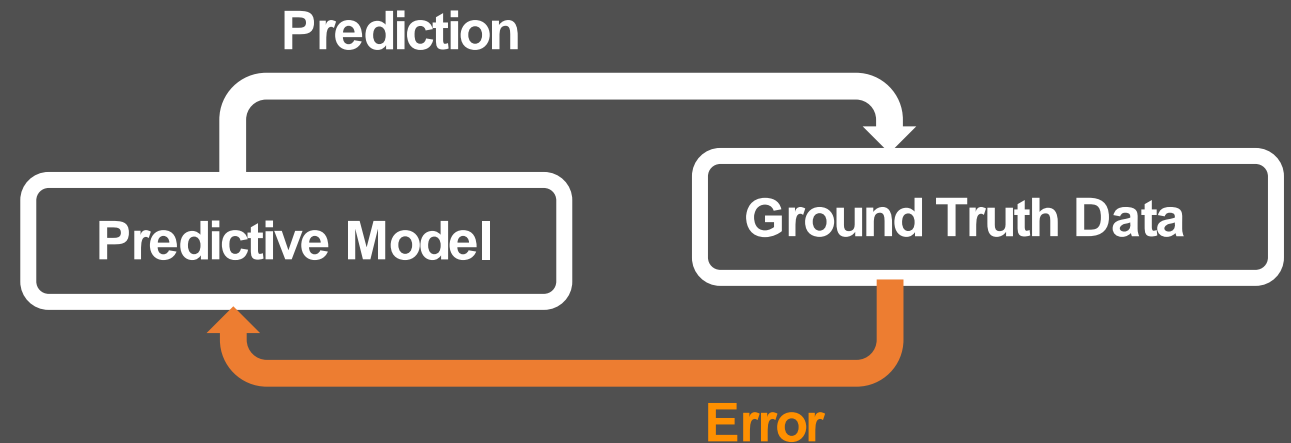
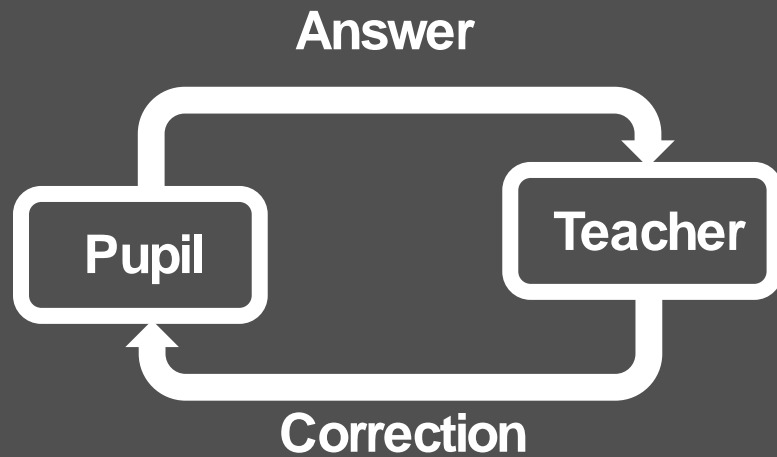
Two Processes:

- **Forward Propagation (Prediction)**
- **Backward Propagation (Weight Tuning)**



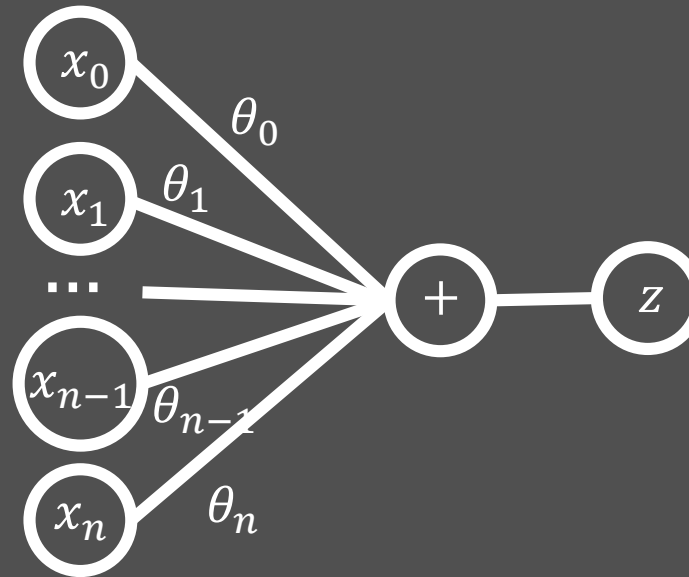
Two Processes:

- Forward Propagation (Prediction)
- **Backward Propagation (Weight Tuning)**

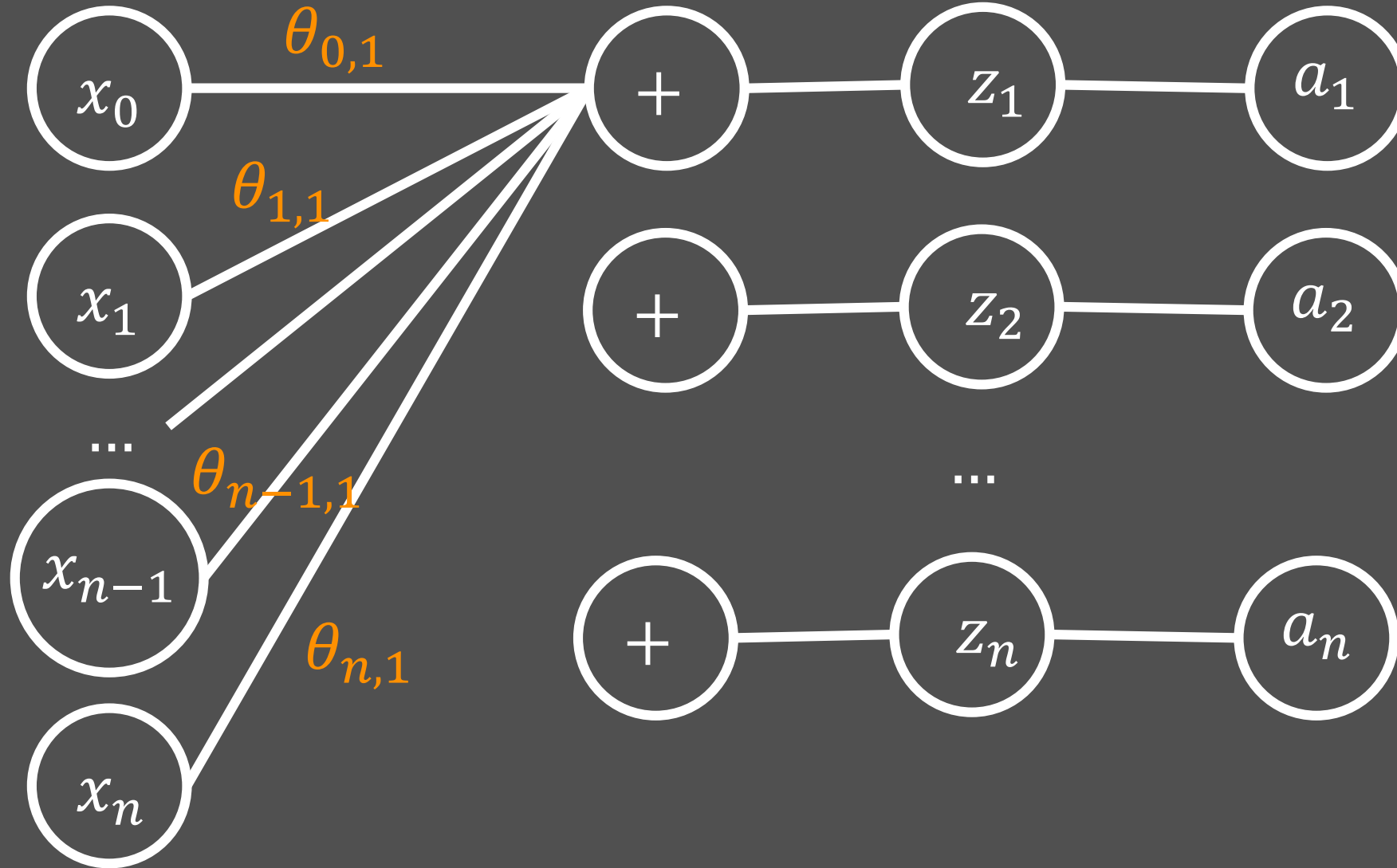


Forward Propagation

- Compute multiple weighted sums z_i
- Activate them to get $a_i(z_i)$
- Remember from Perceptron: $z = \theta x$

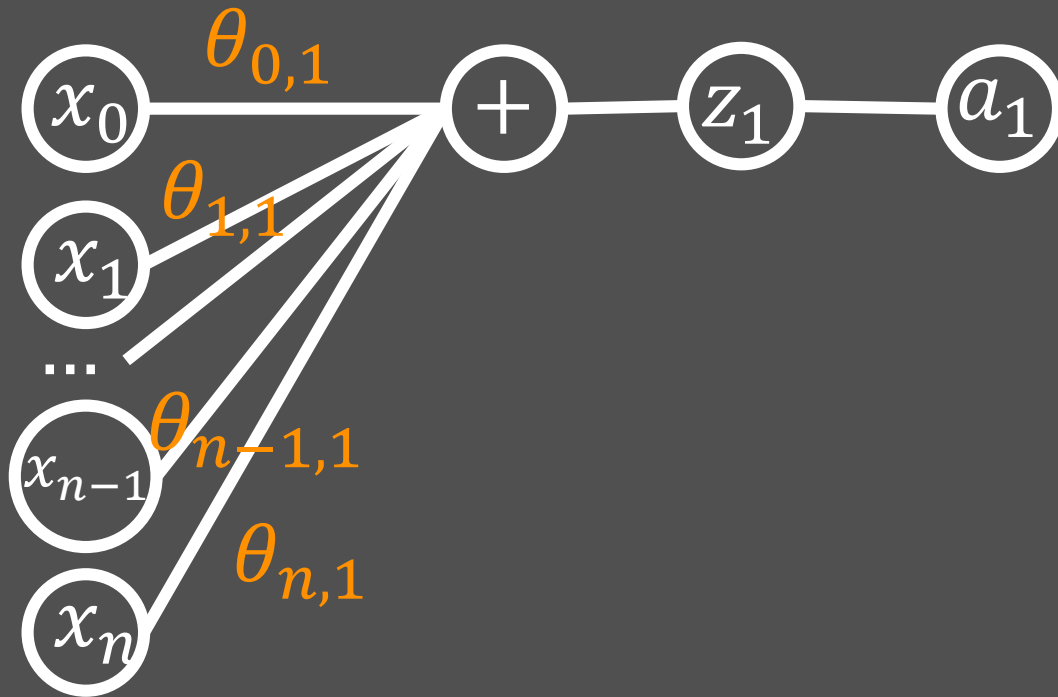


Forward Propagation

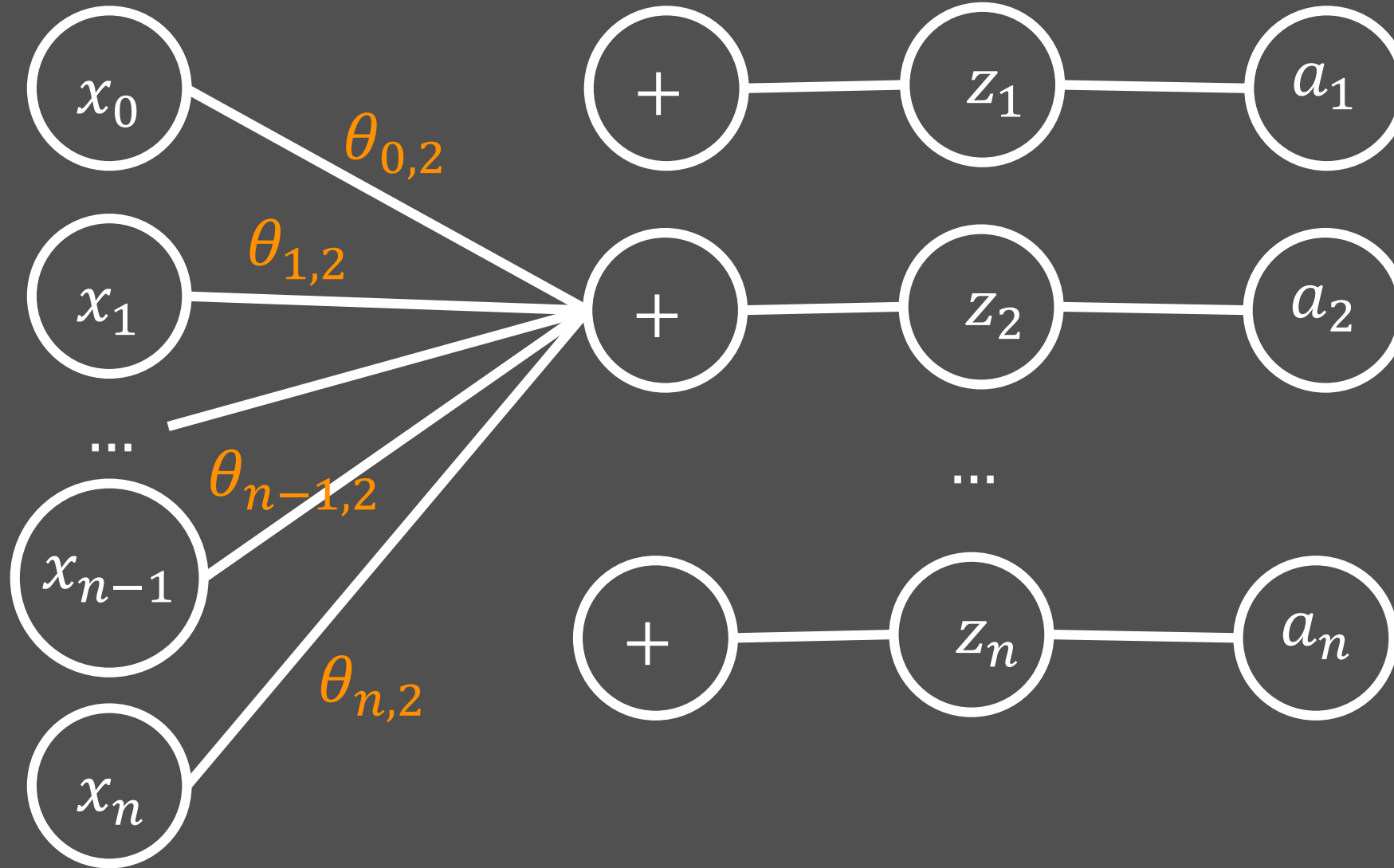


$$z_{to} = \theta_{from,to} x_{from}$$

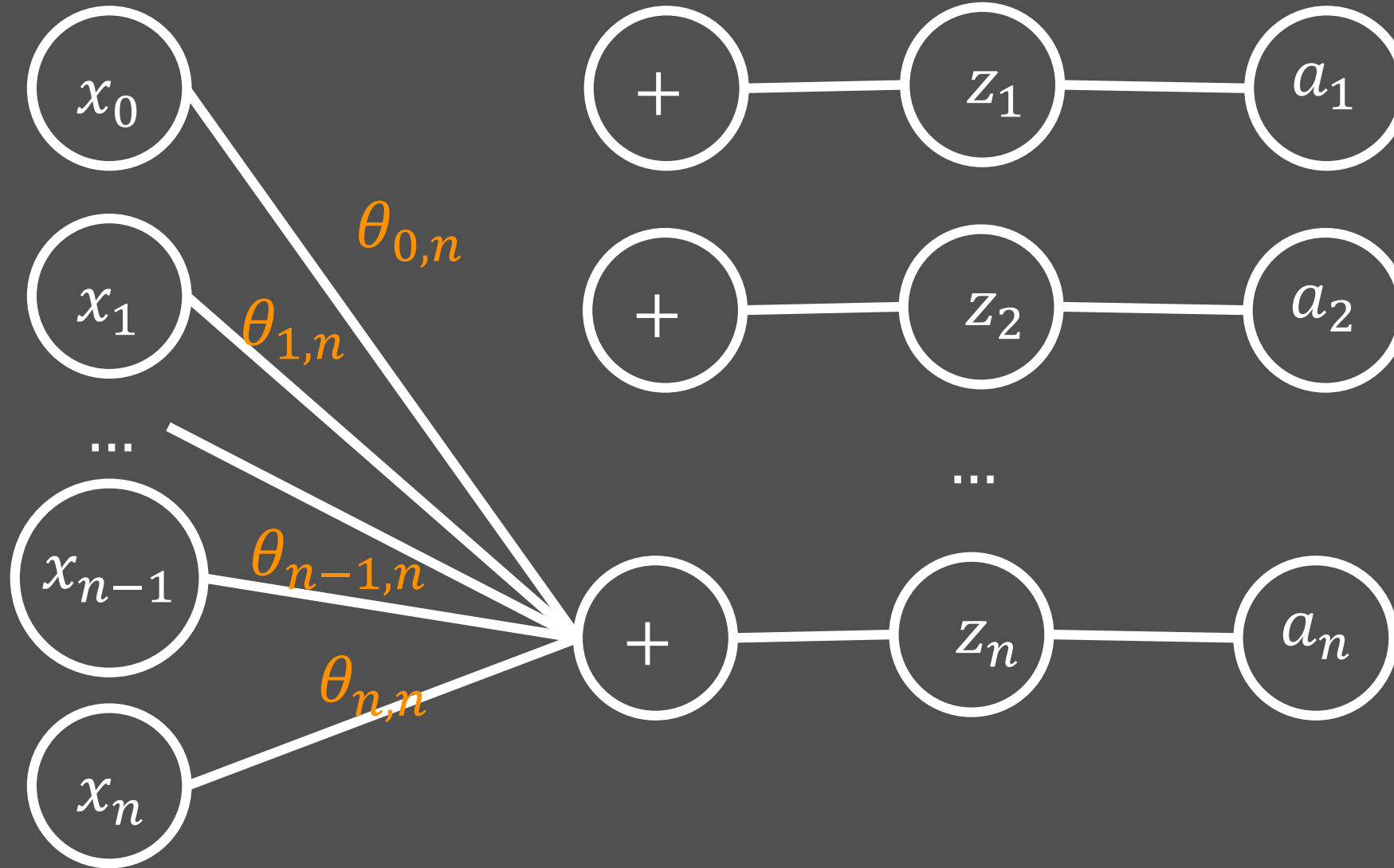
e.g. $z_1 = \theta_{0,1} x_0$



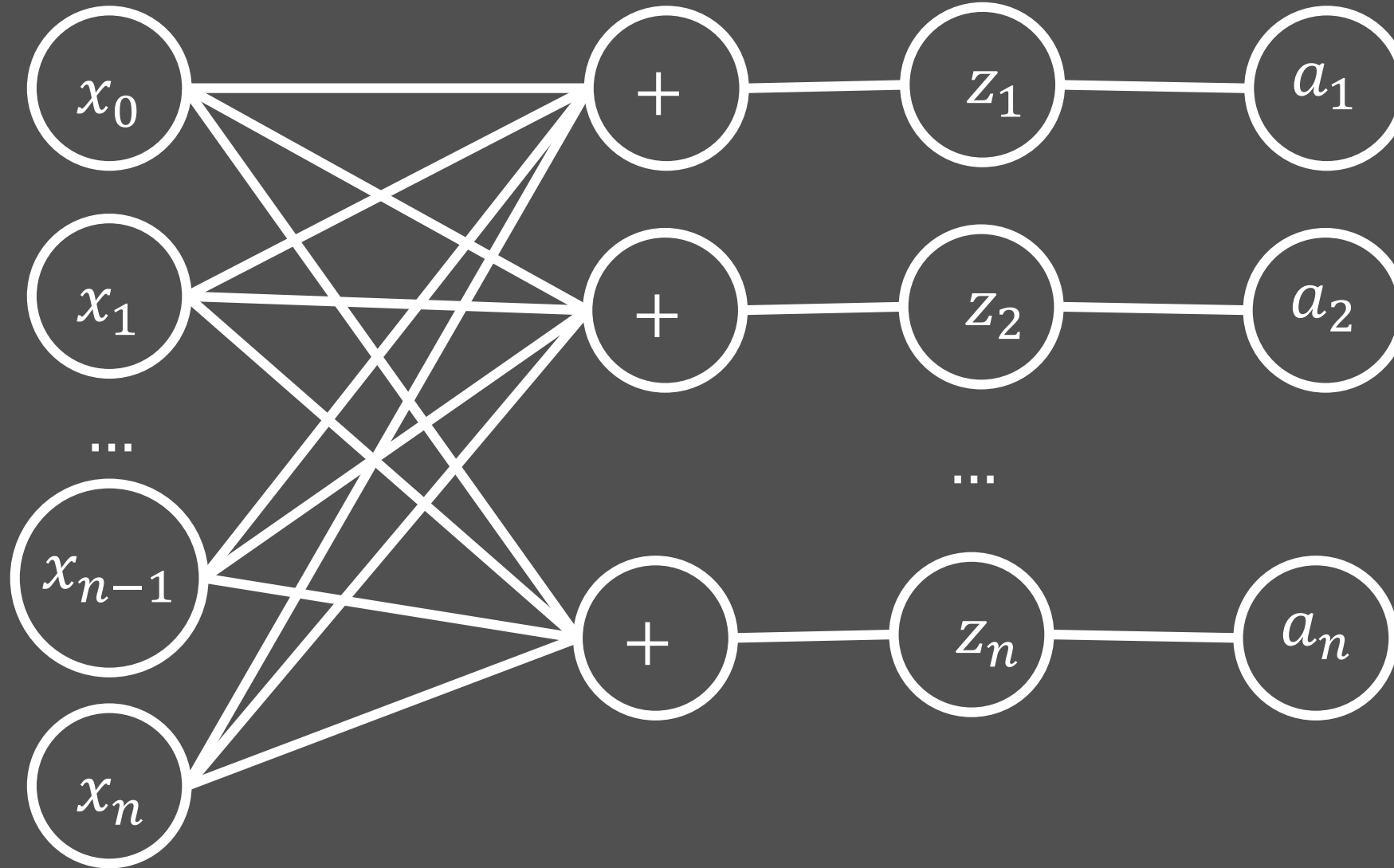
Forward Propagation



Forward Propagation

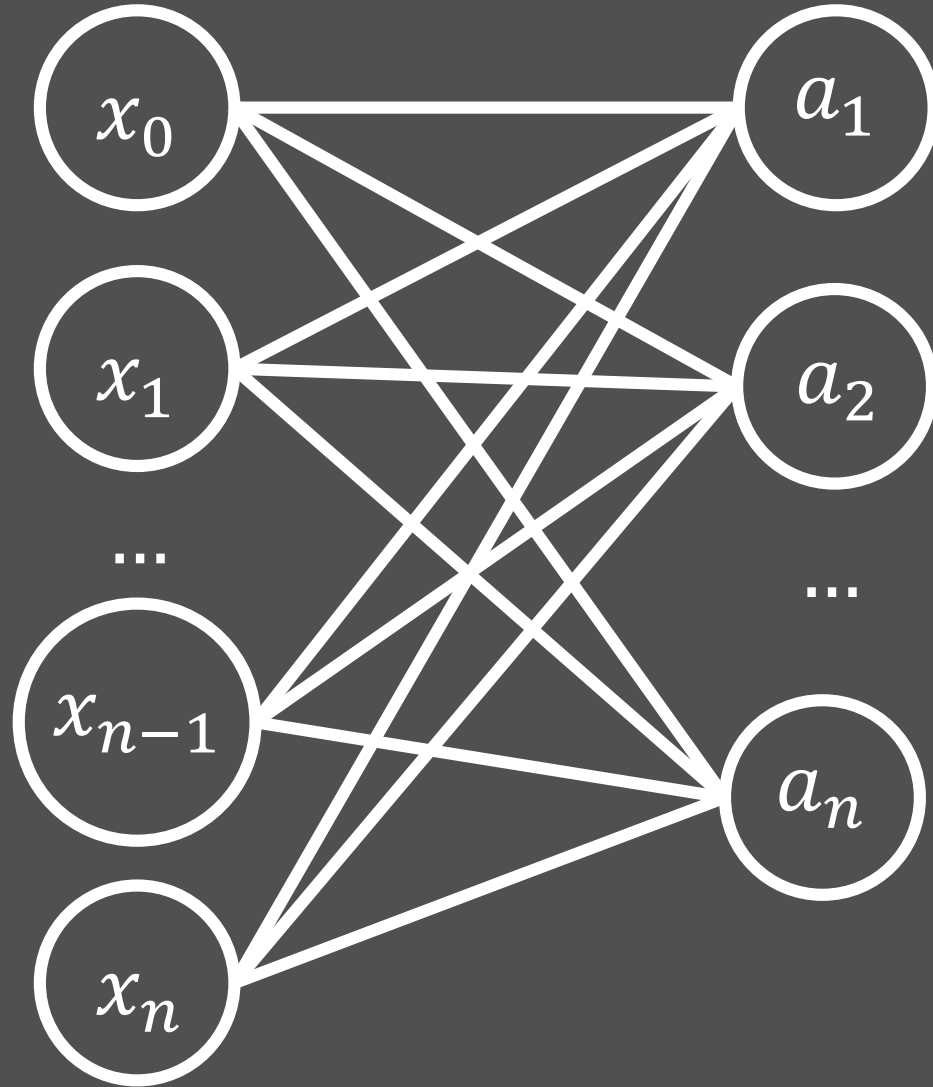


Forward Propagation

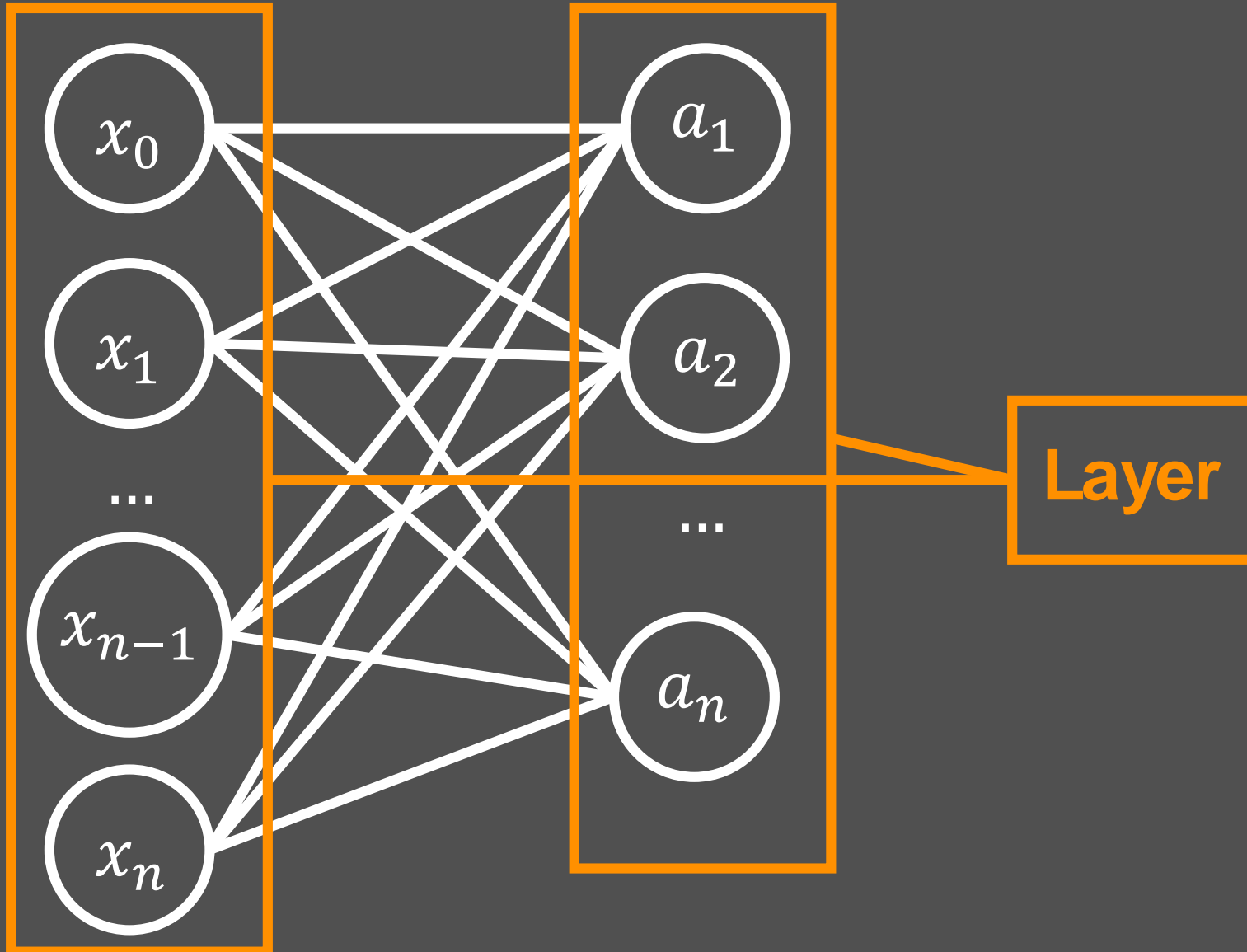


In Short (without z)

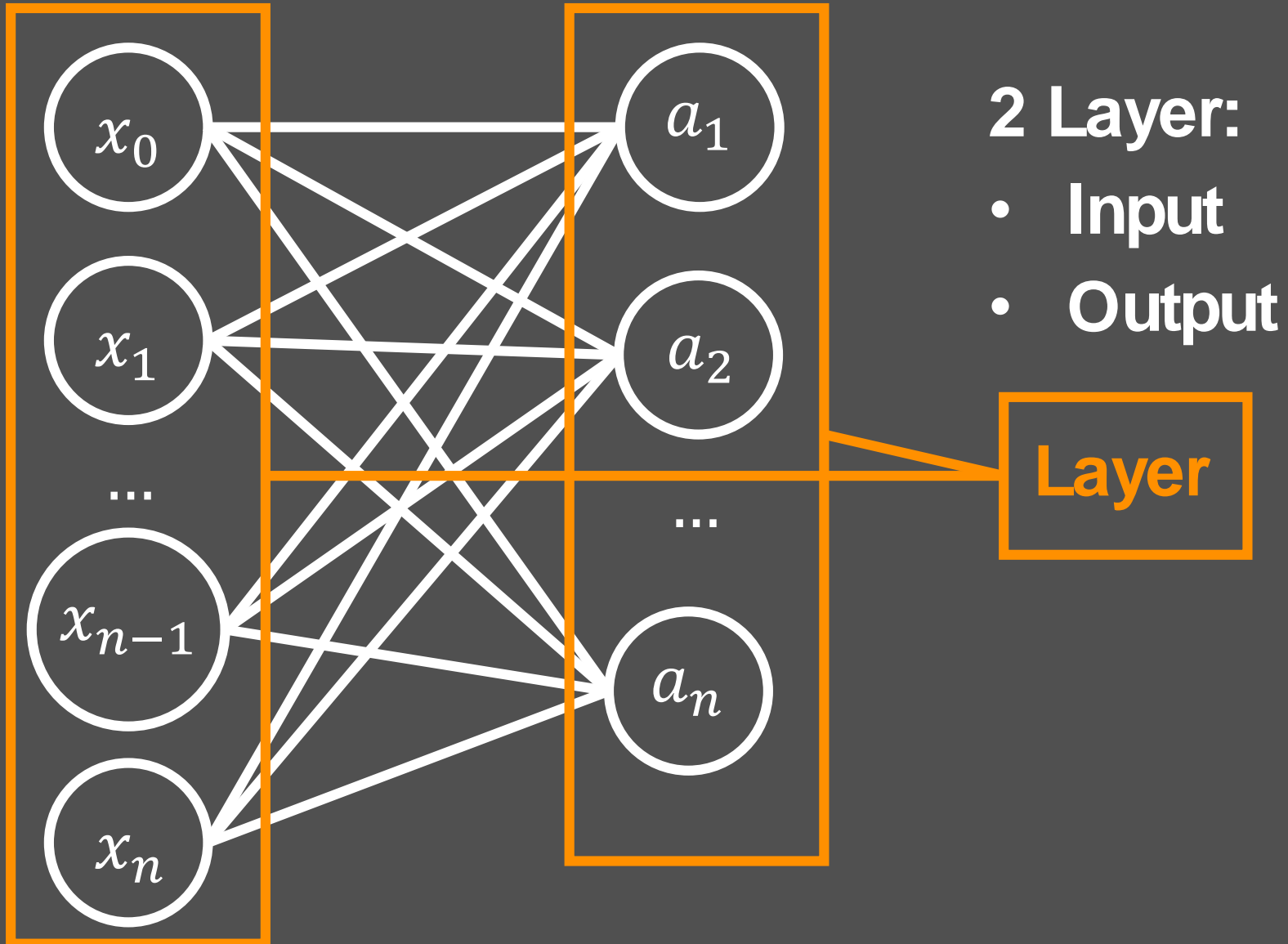
Forward Propagation



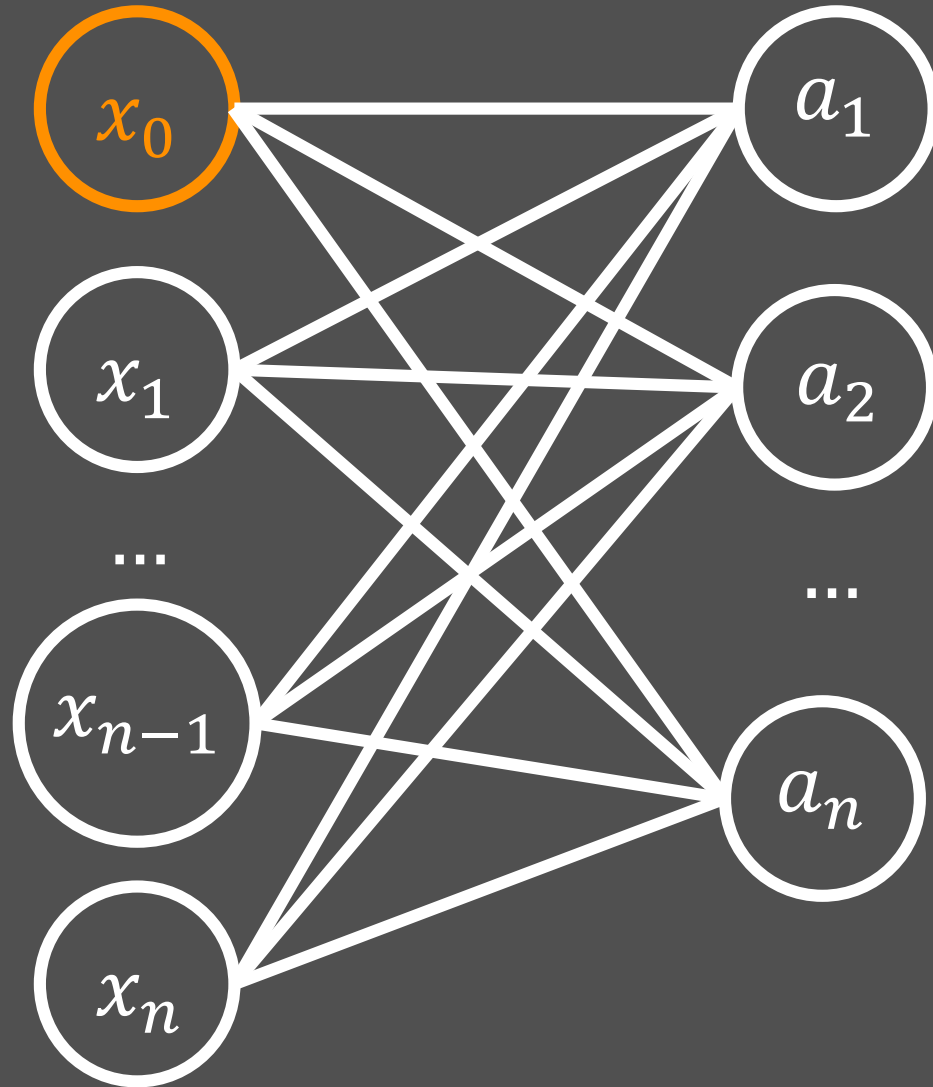
Forward Propagation



Forward Propagation

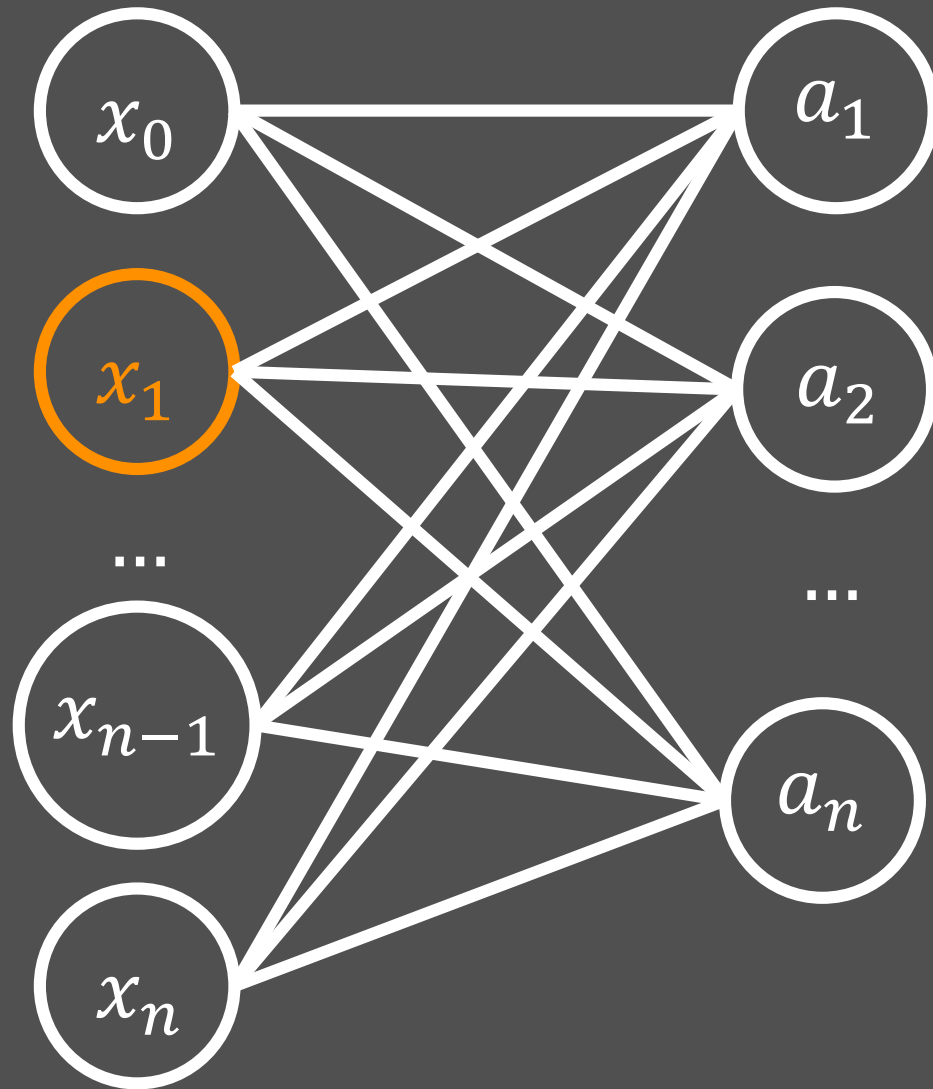


Forward Propagation



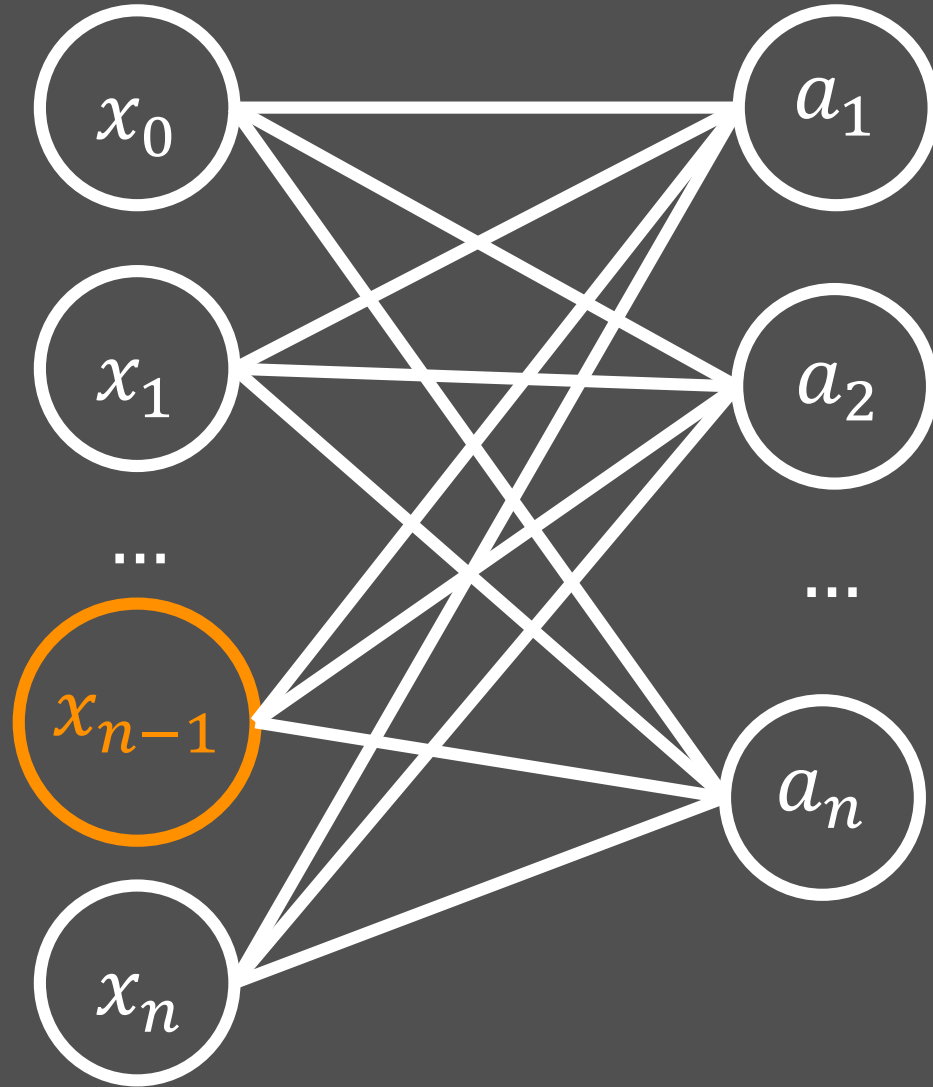
Input Neuron

Forward Propagation



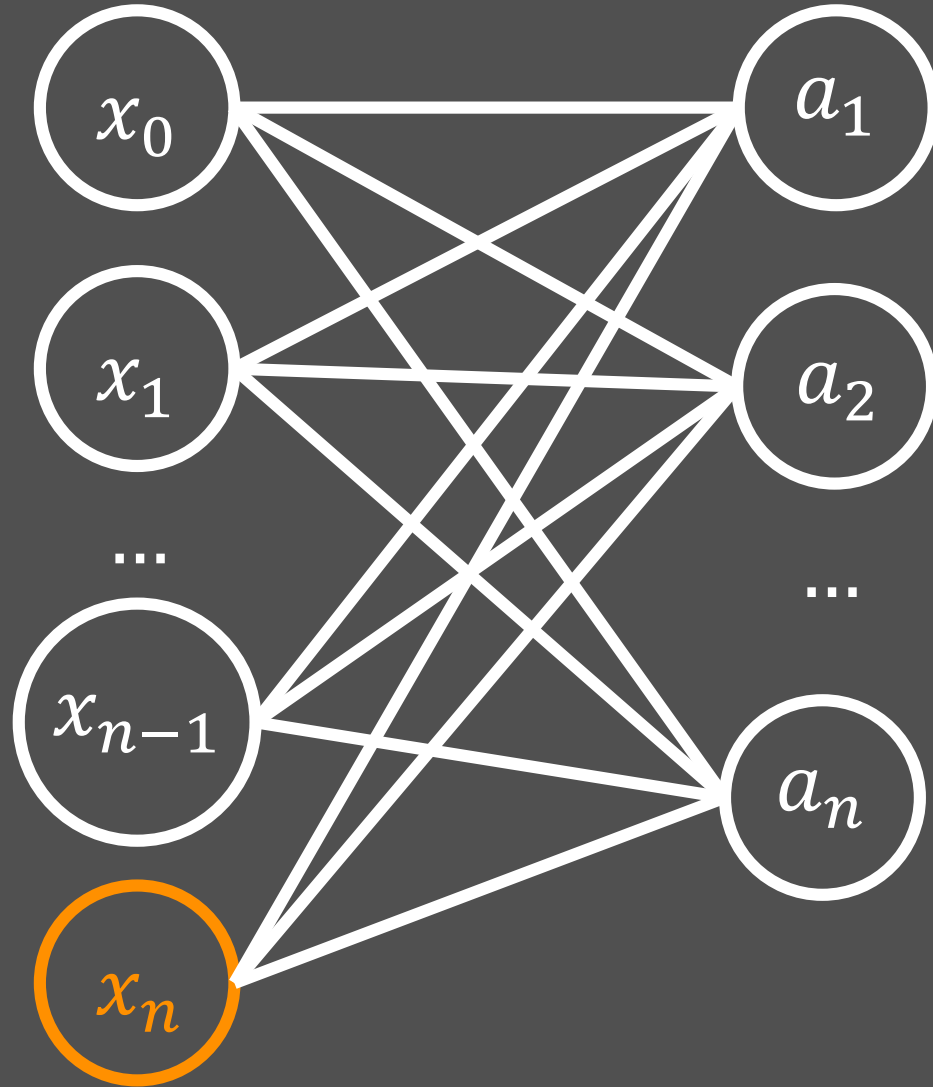
Input Neuron

Forward Propagation



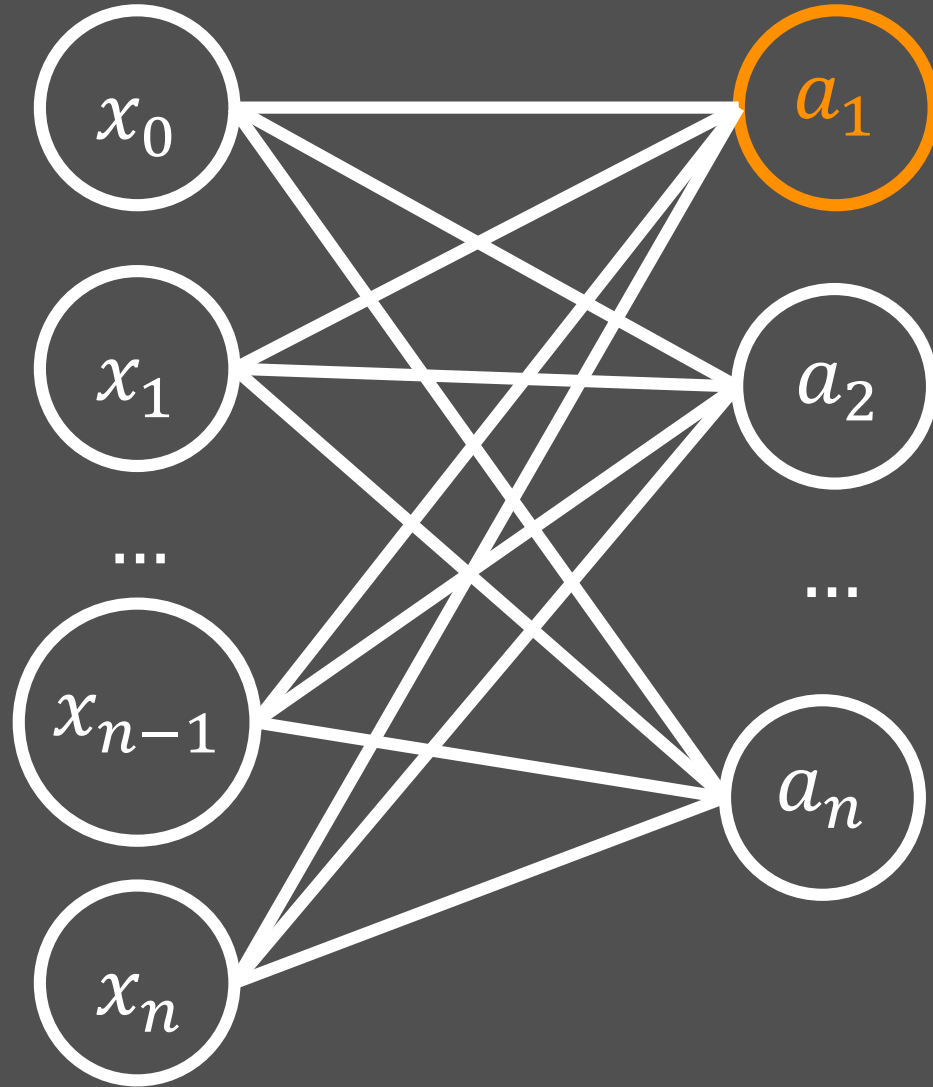
Input Neuron

Forward Propagation



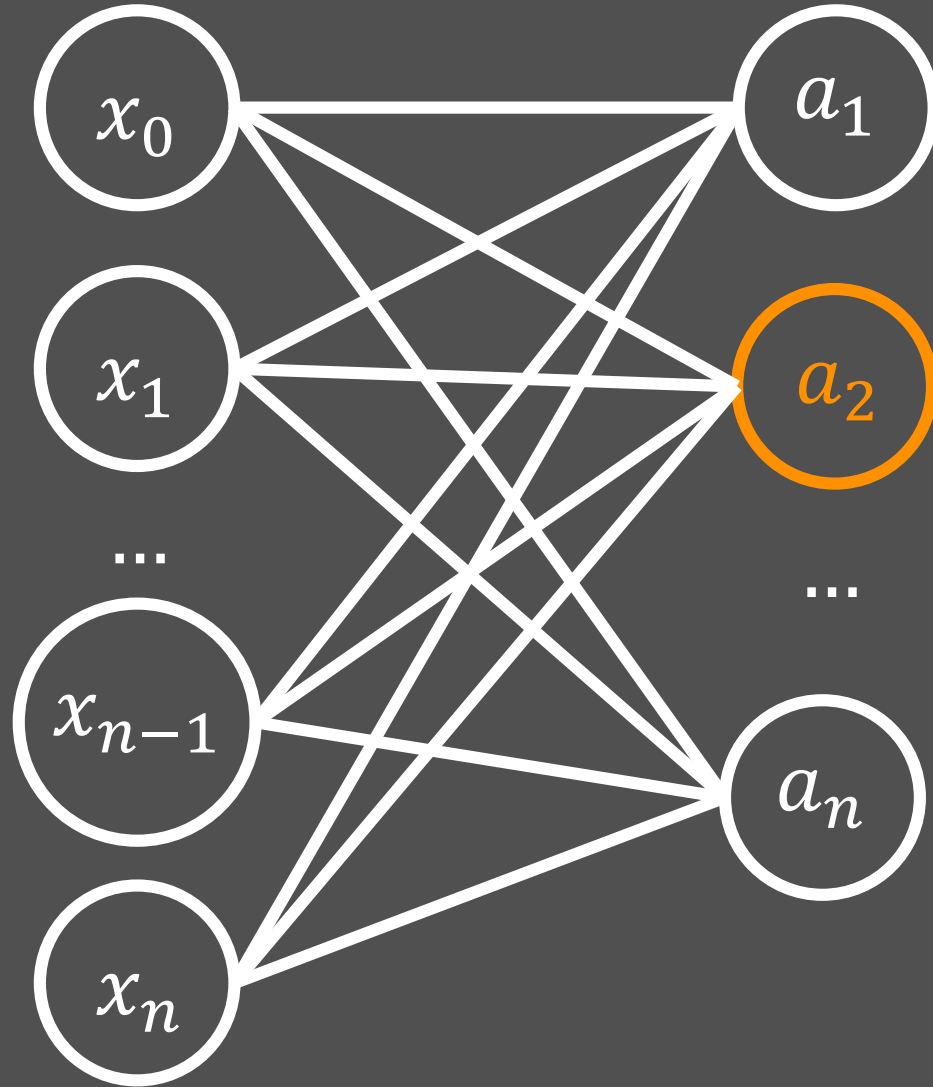
Input Neuron

Forward Propagation



Output Neuron

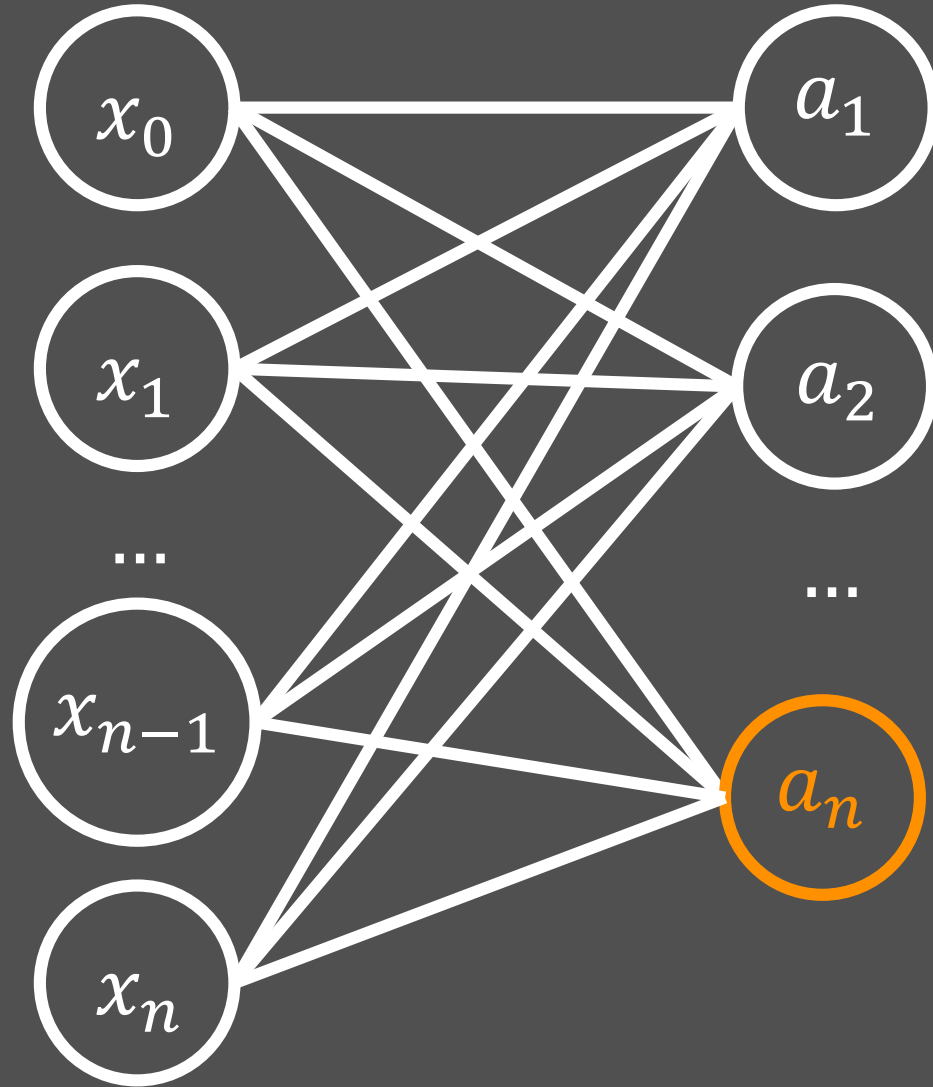
Forward Propagation



Output Neuron

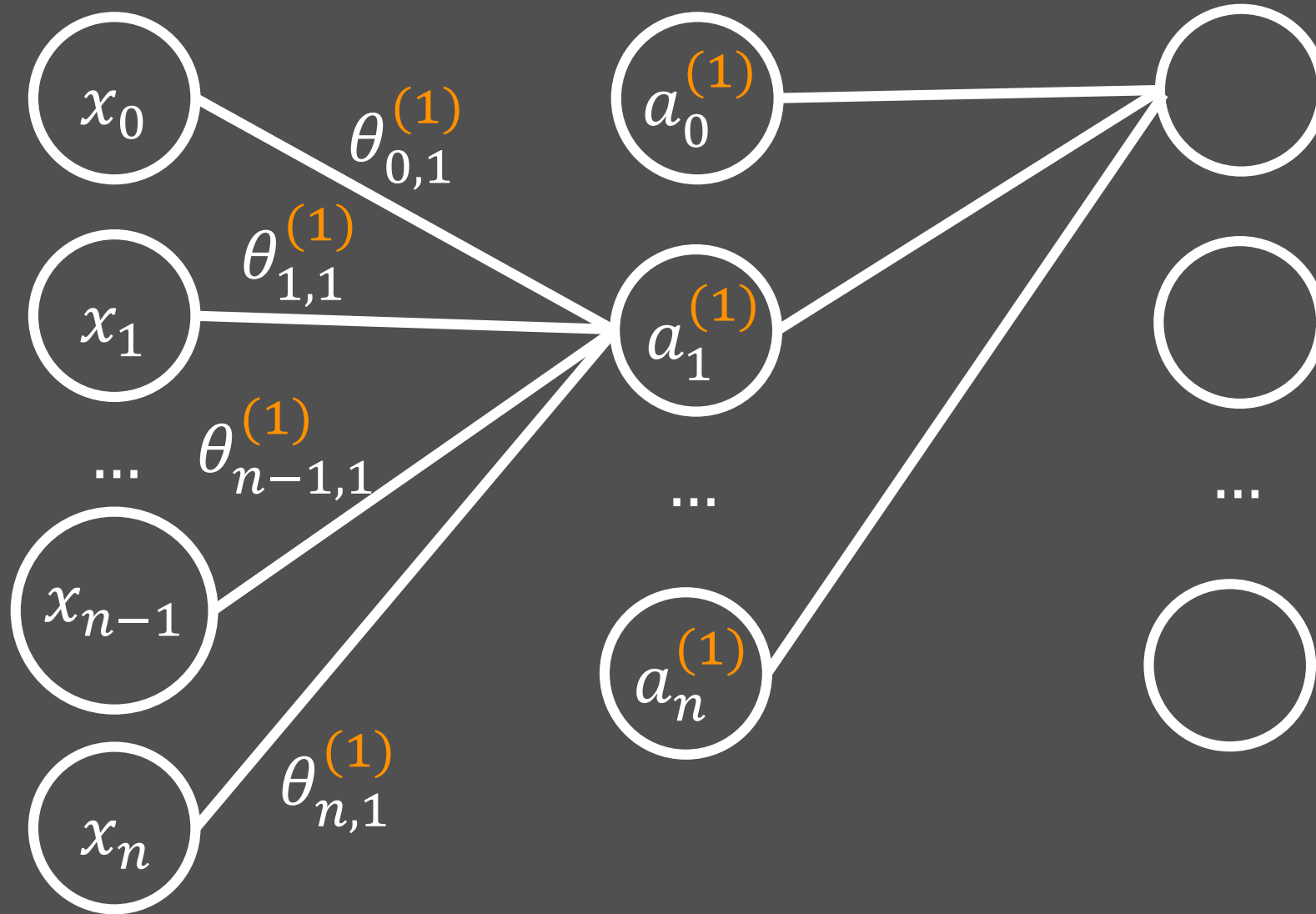
Forward Propagation

Output Neuron

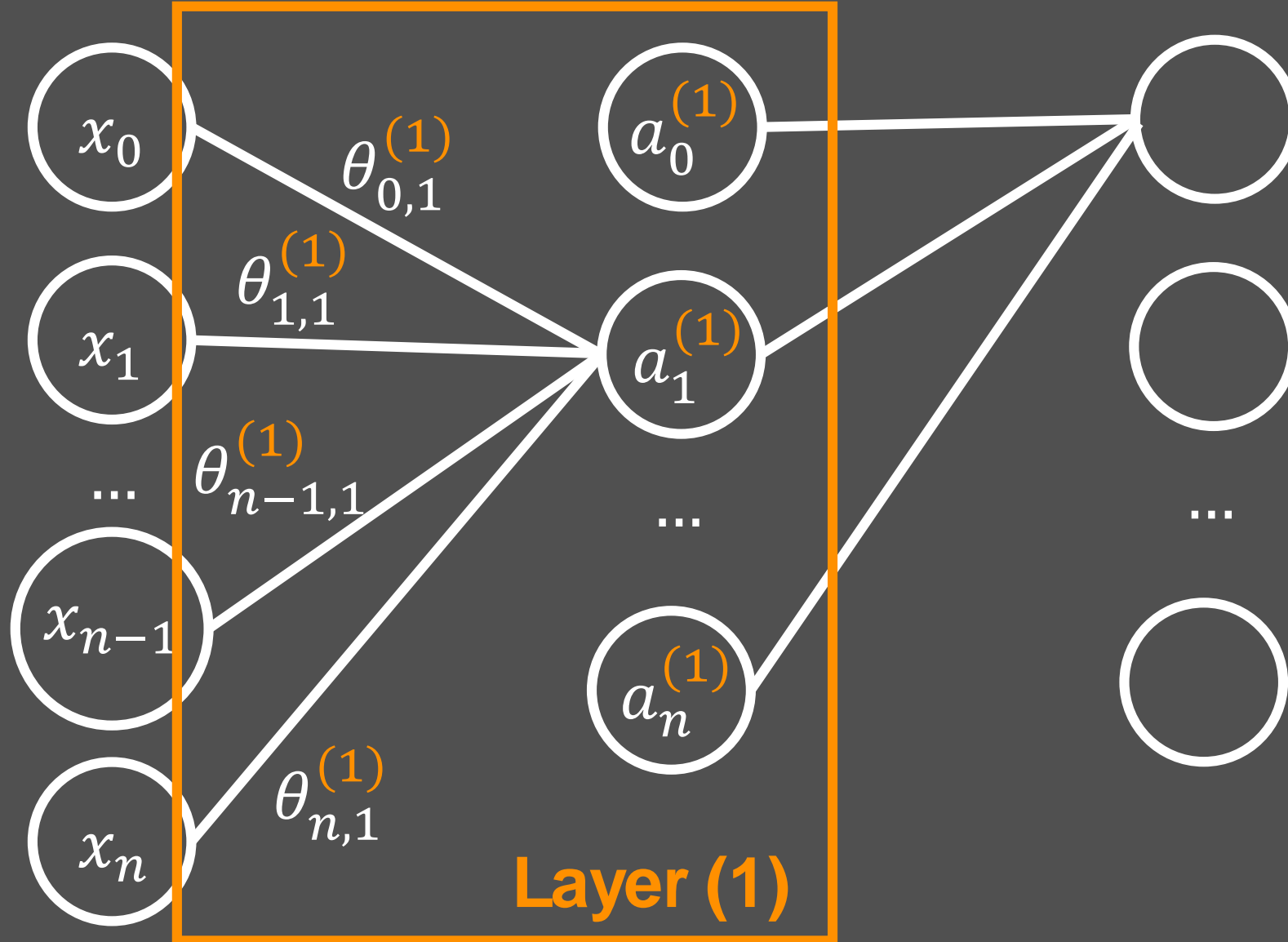


Add More Layer!

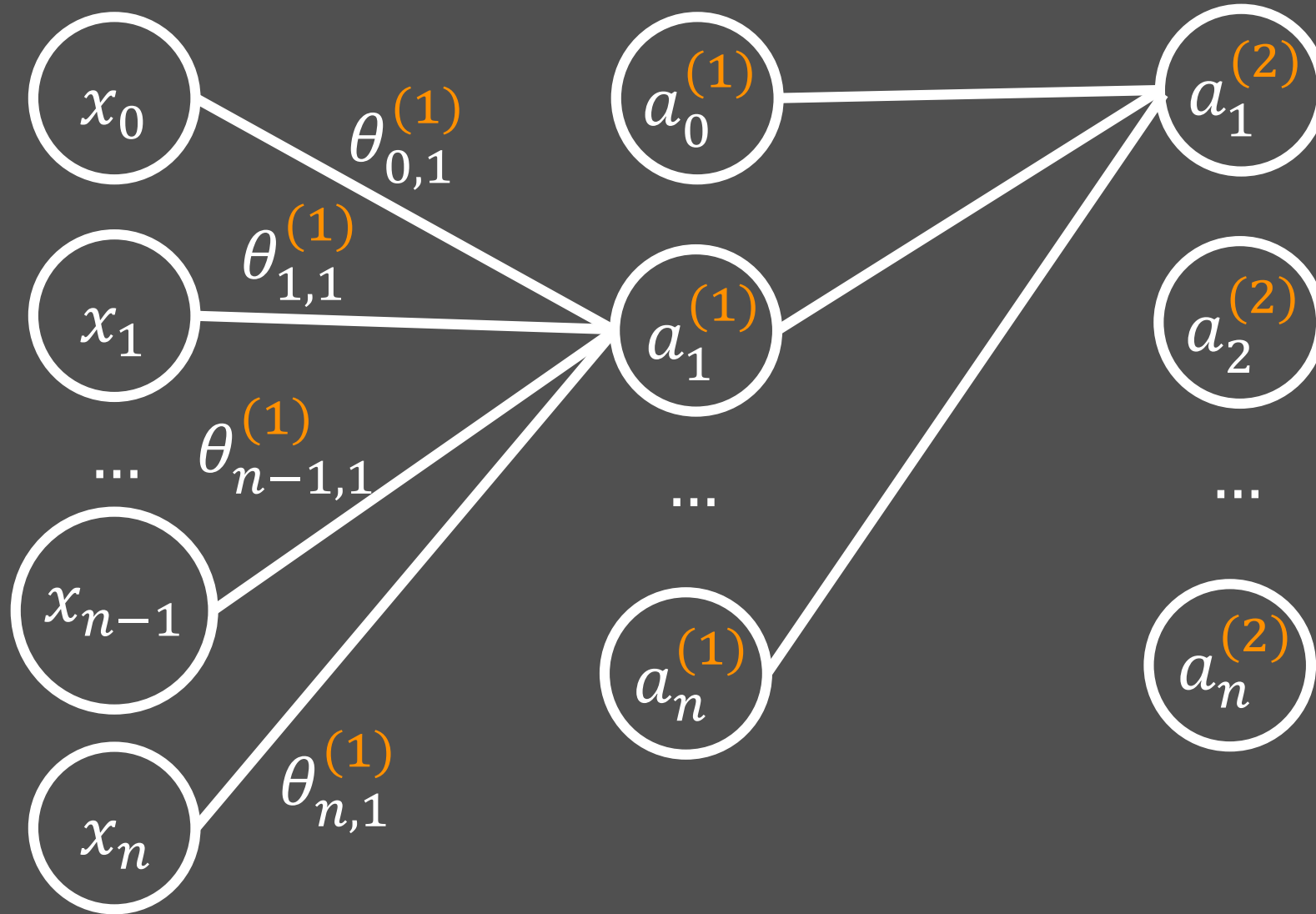
Forward Propagation



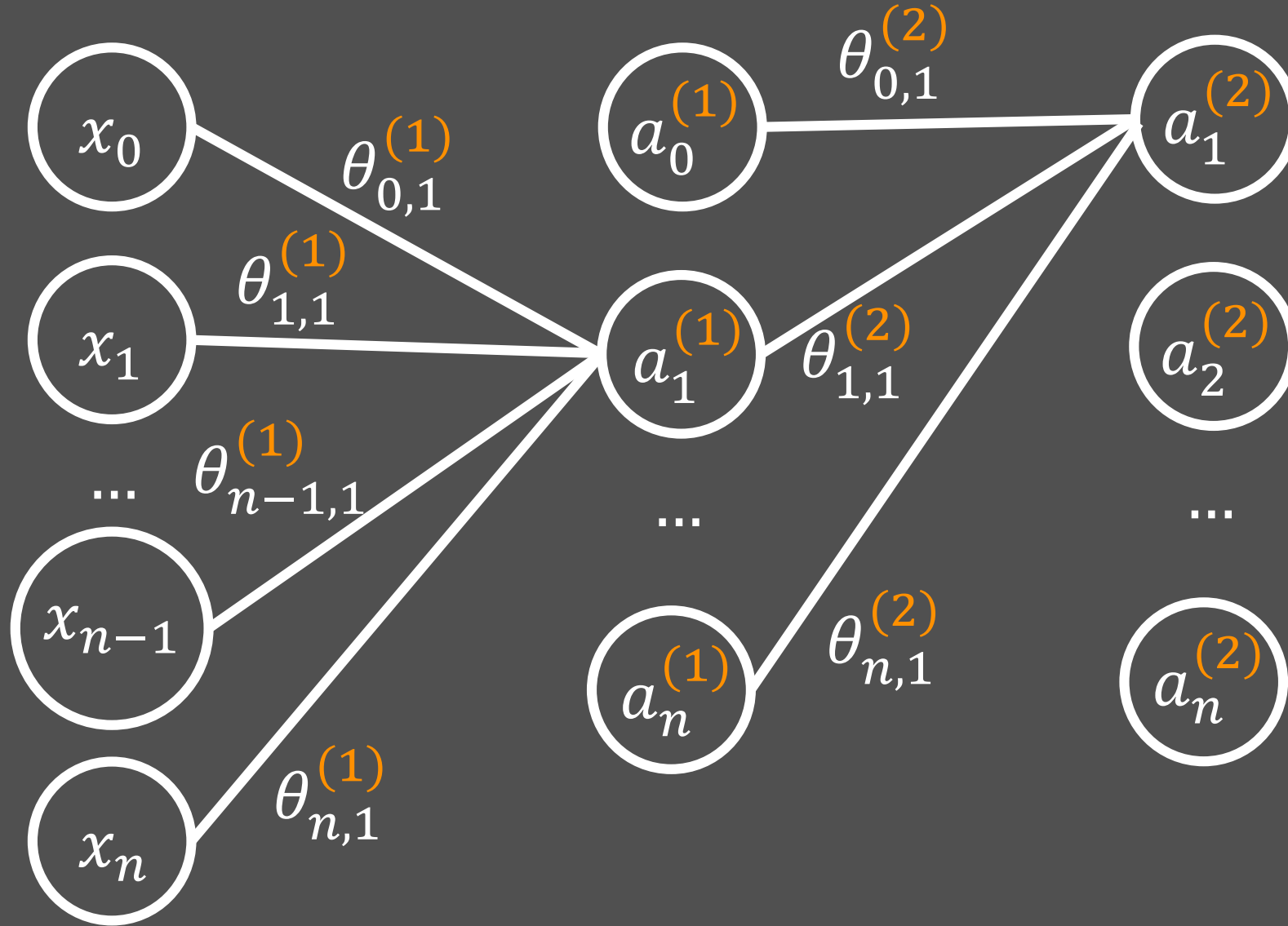
Forward Propagation



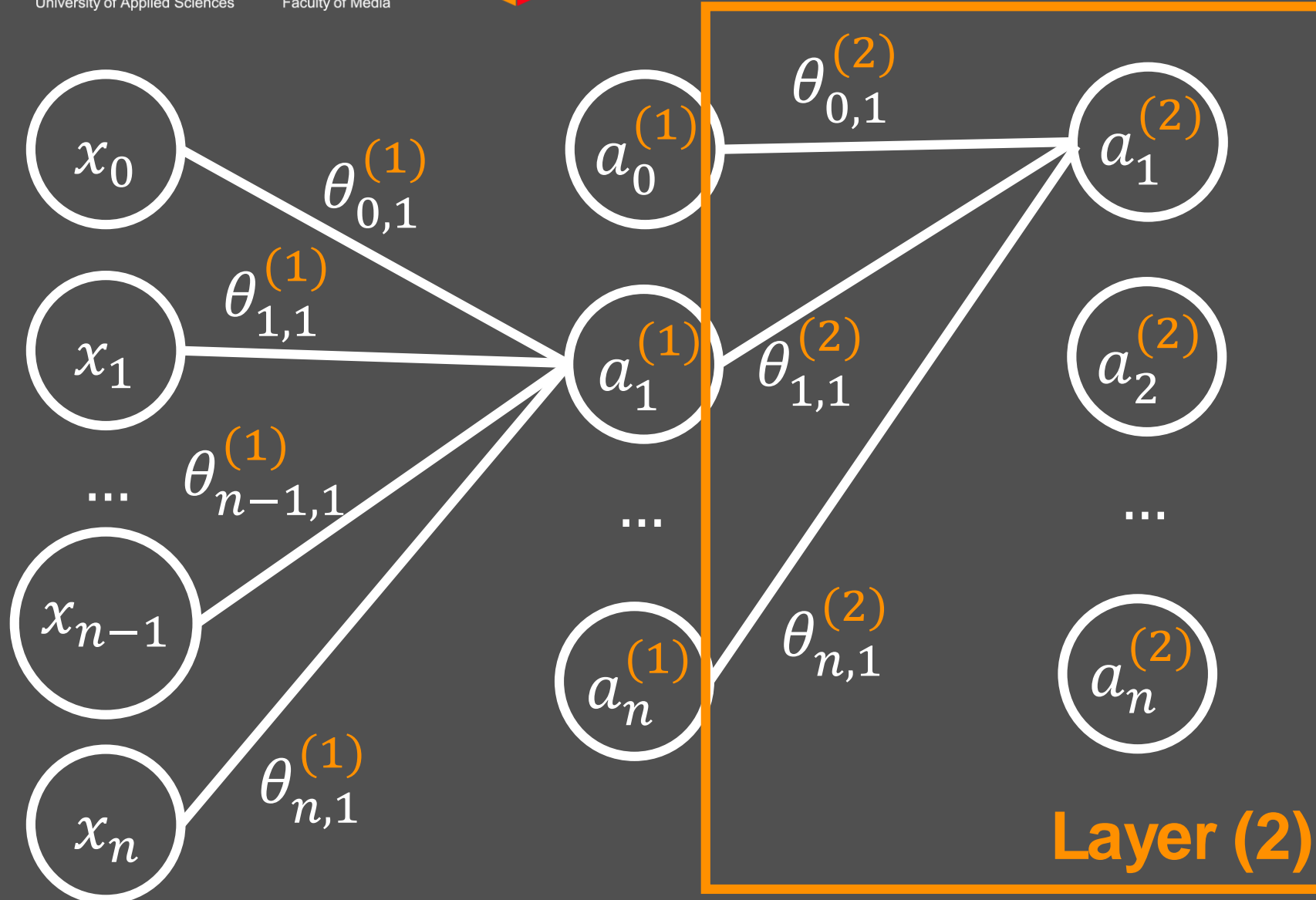
Forward Propagation



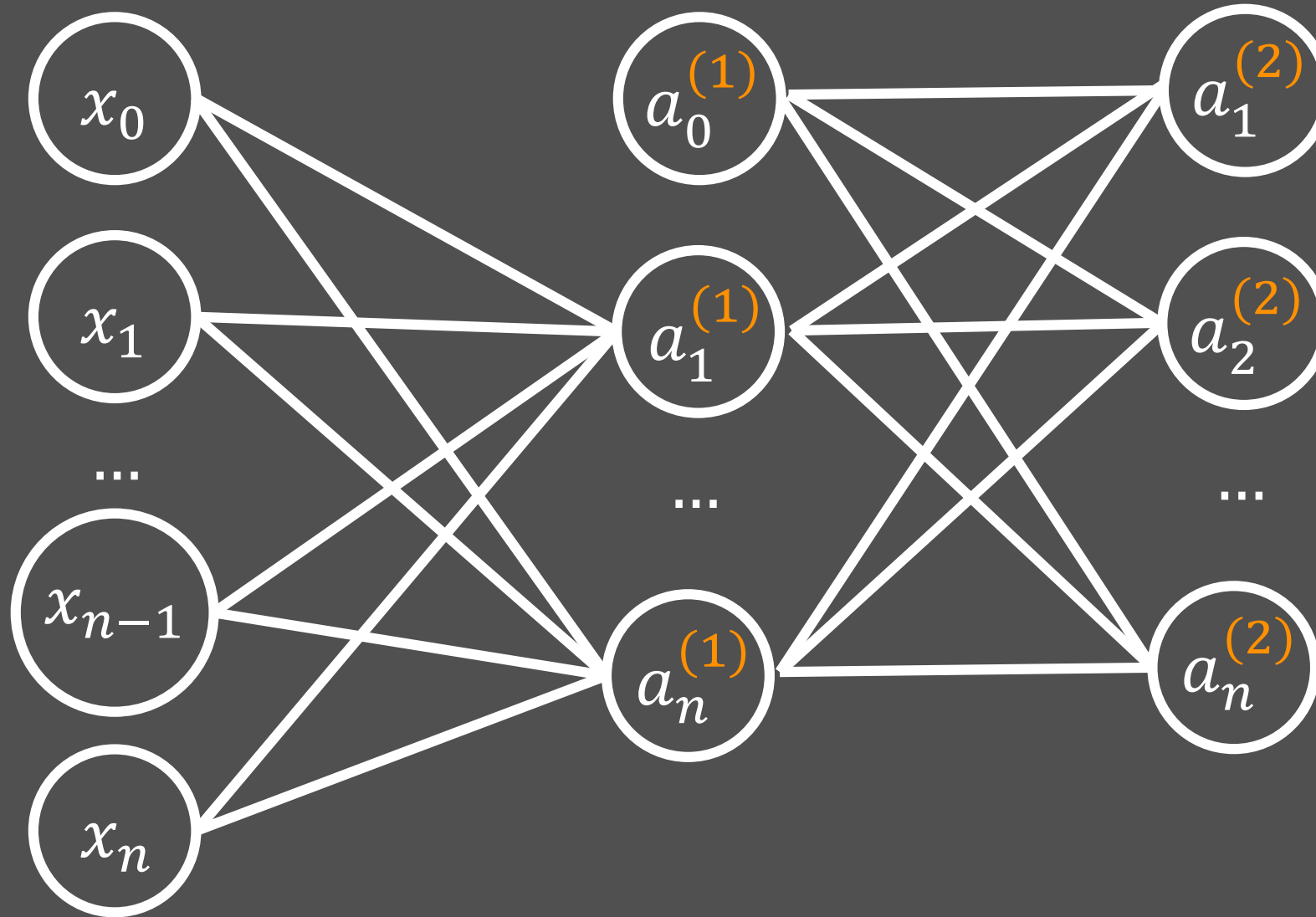
Forward Propagation



Forward Propagation

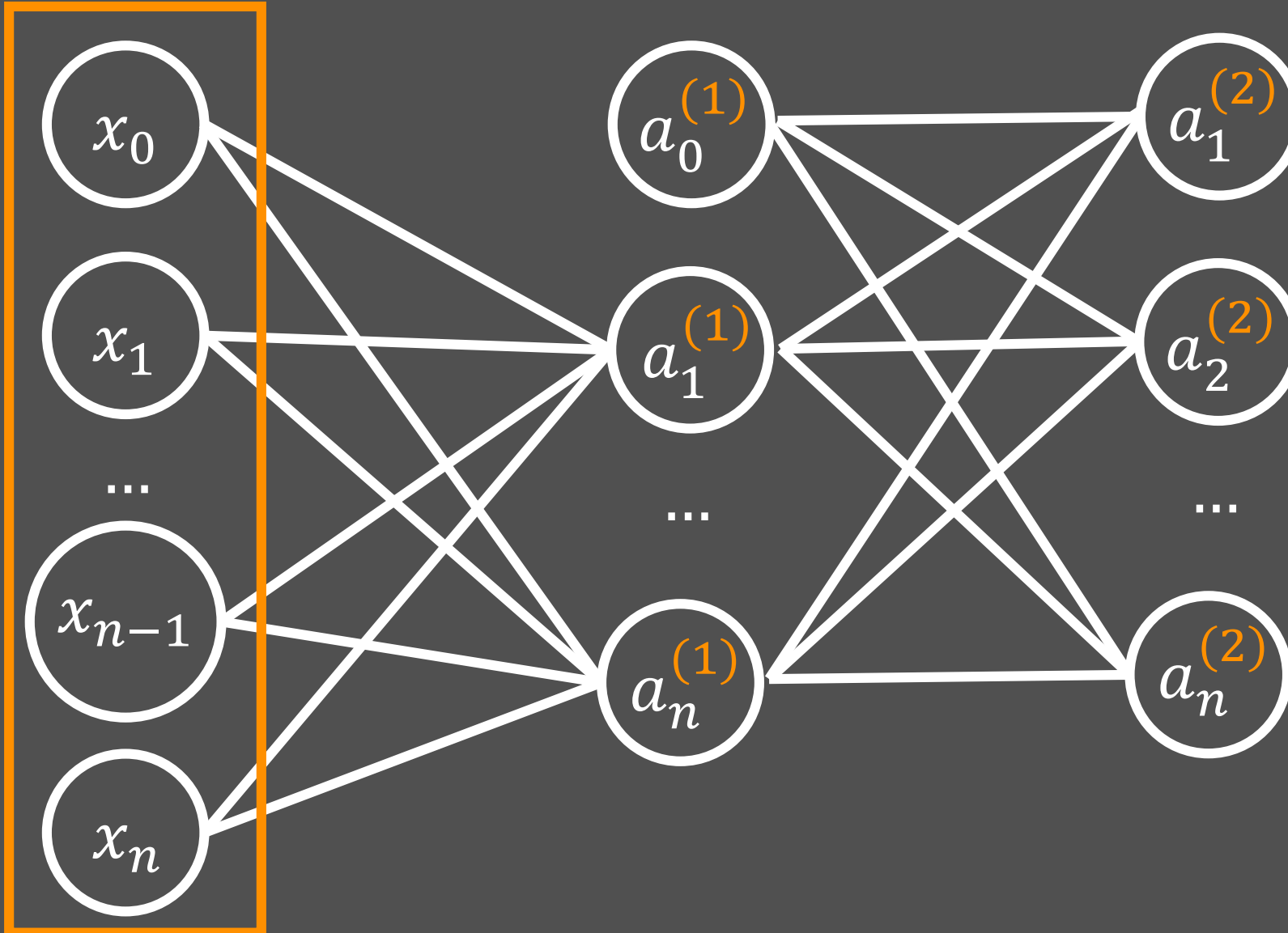


Forward Propagation



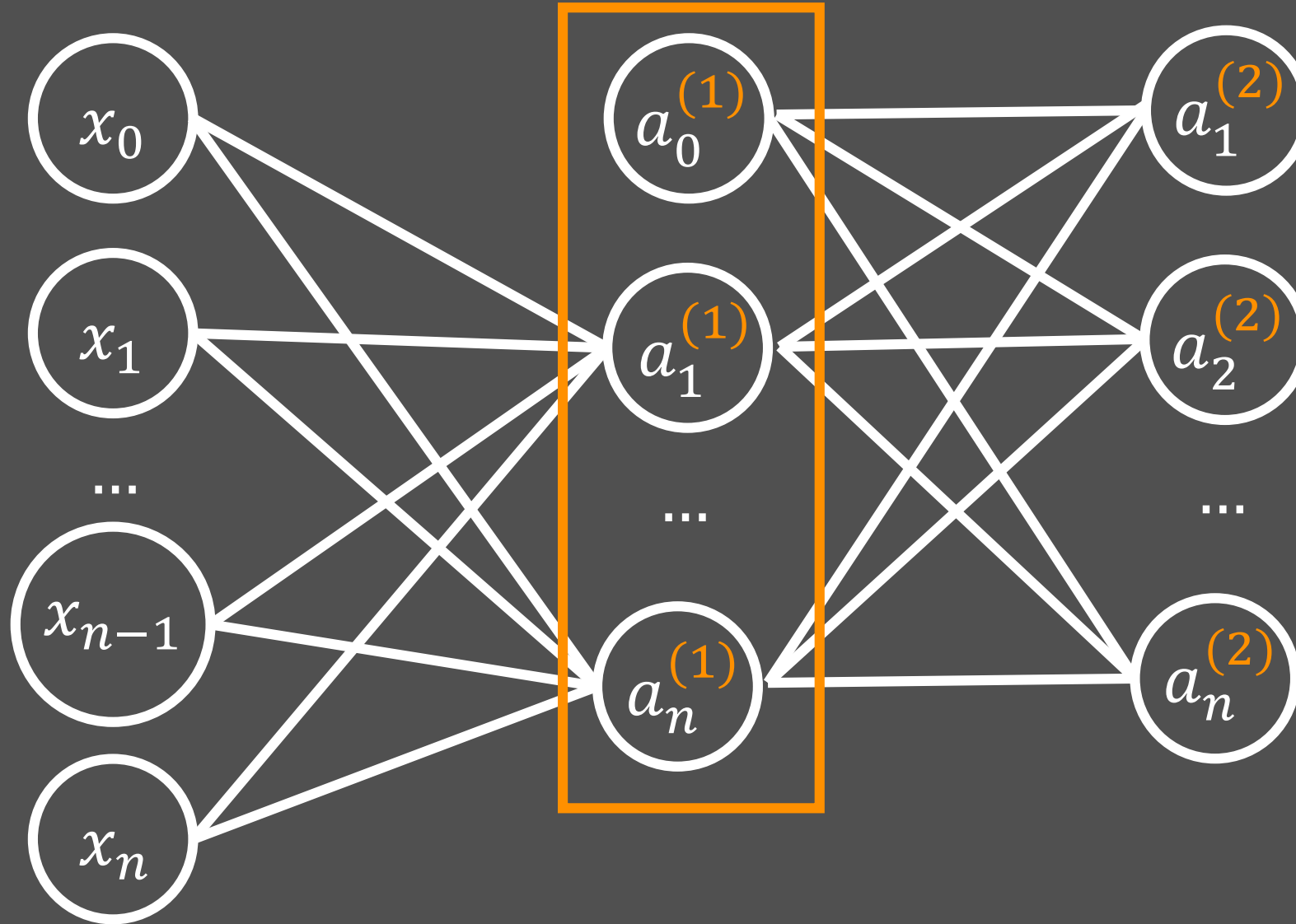
Forward Propagation

Input Layer



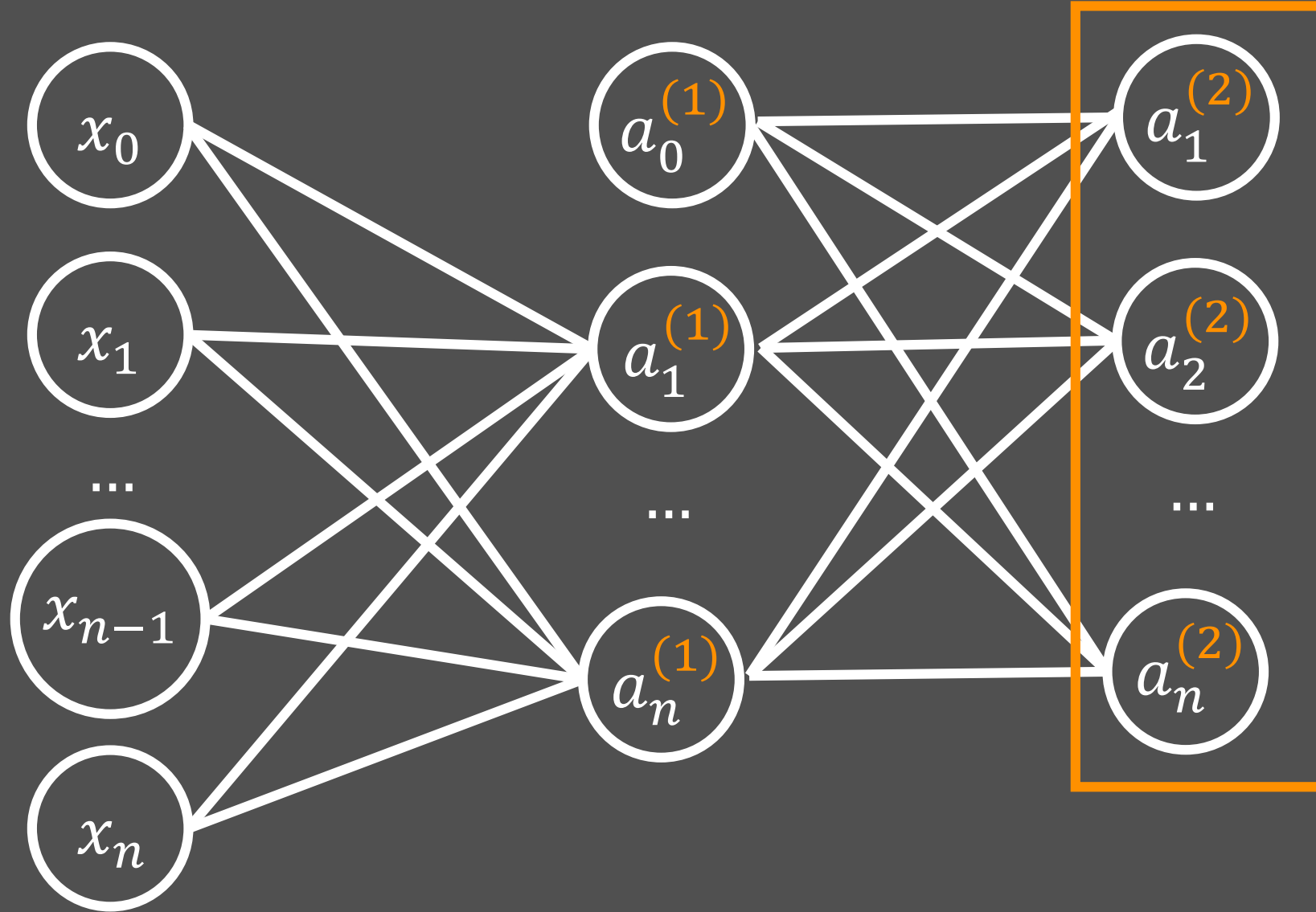
Forward Propagation

Hidden Layer

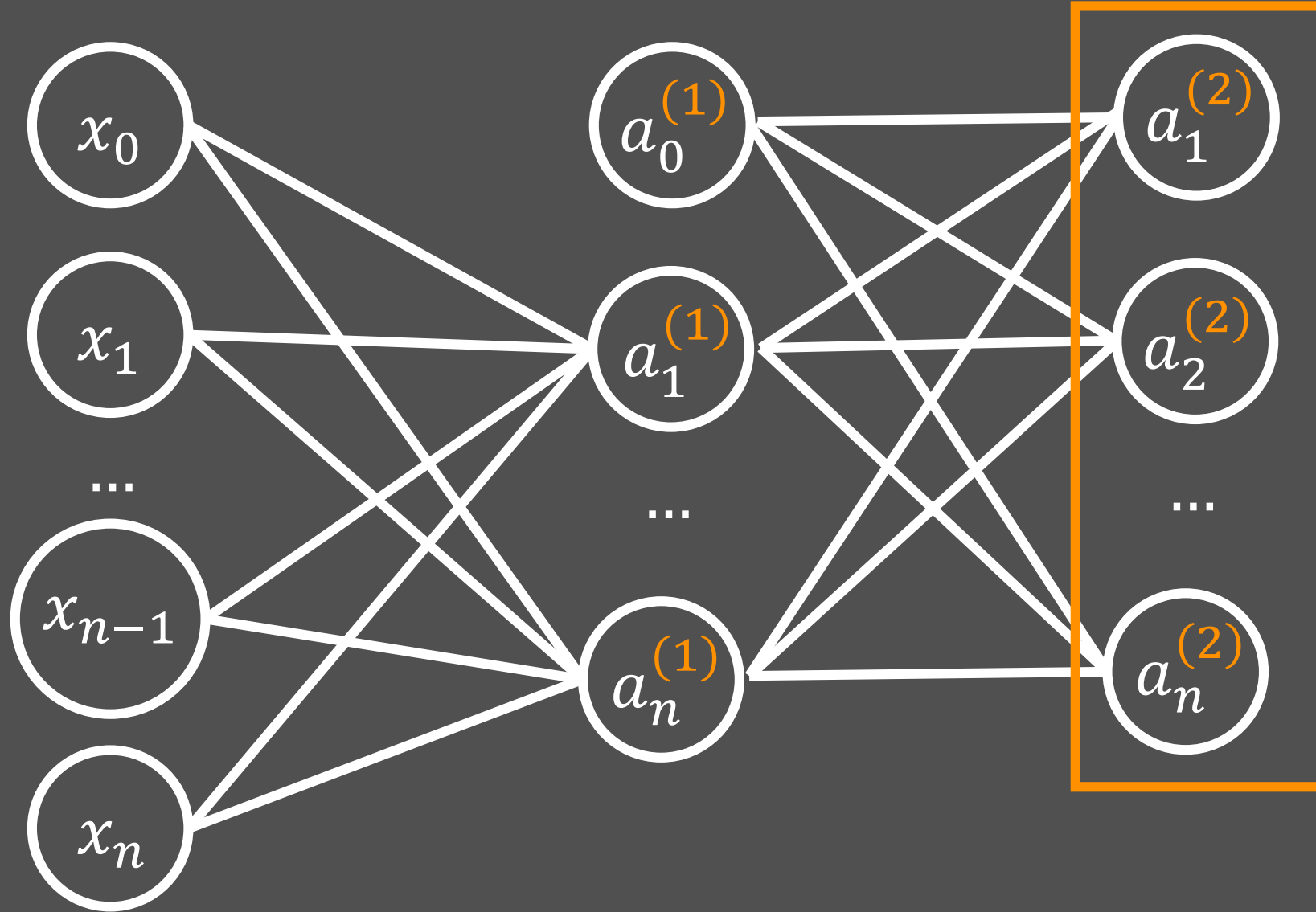


Forward Propagation

Output Layer



Forward Propagation



Output Layer:

- **Multiclass Classification**
- **Each neuron could be interpreted as class probability**

Forward Propagation

- Object Recognition
- Assume 4 classes
- L-Layer (L is the last layer)

$$a_1^{(L)}$$

Car

$$a_2^{(L)}$$

Airplane

$$a_3^{(L)}$$

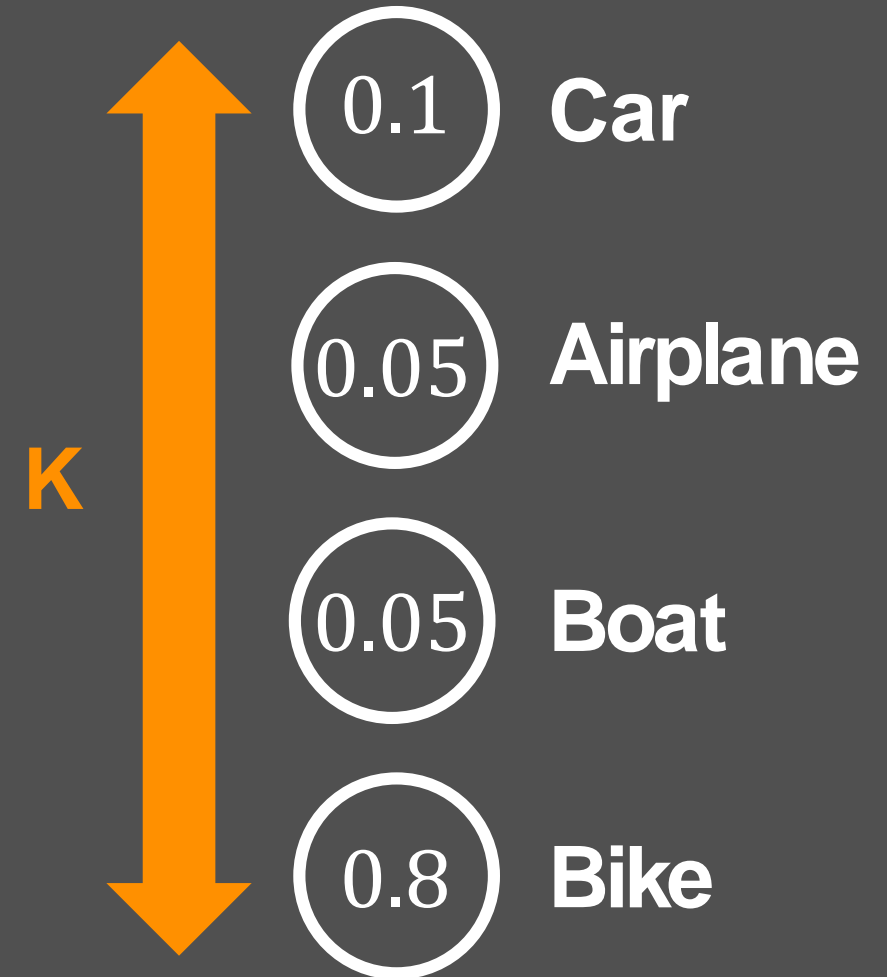
Boat

$$a_4^{(L)}$$

Bike

Forward Propagation

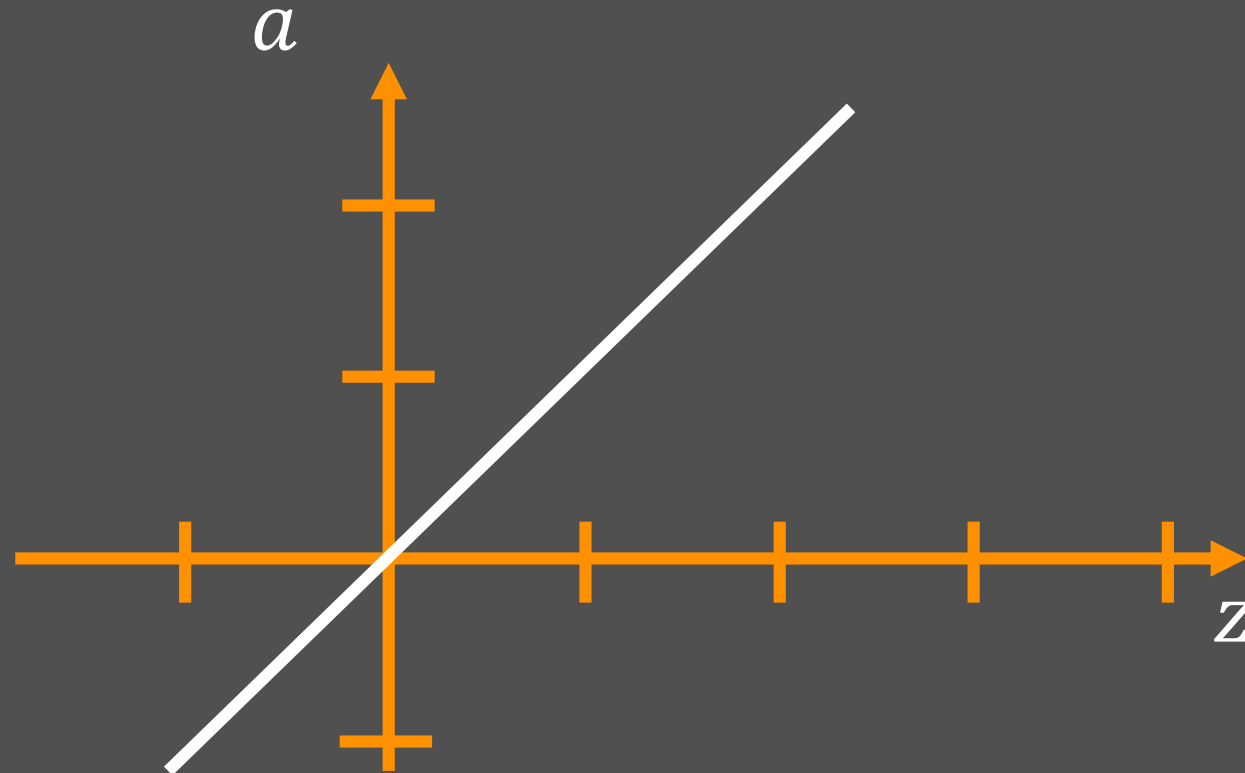
- Object Recognition
- Assume 4 classes
- L-Layer (L is the last layer)
- Softmax: $a = \frac{e^{z_j}}{\sum_{k=1}^K e^{z_k}}$
- Use Softmax for the last Layer



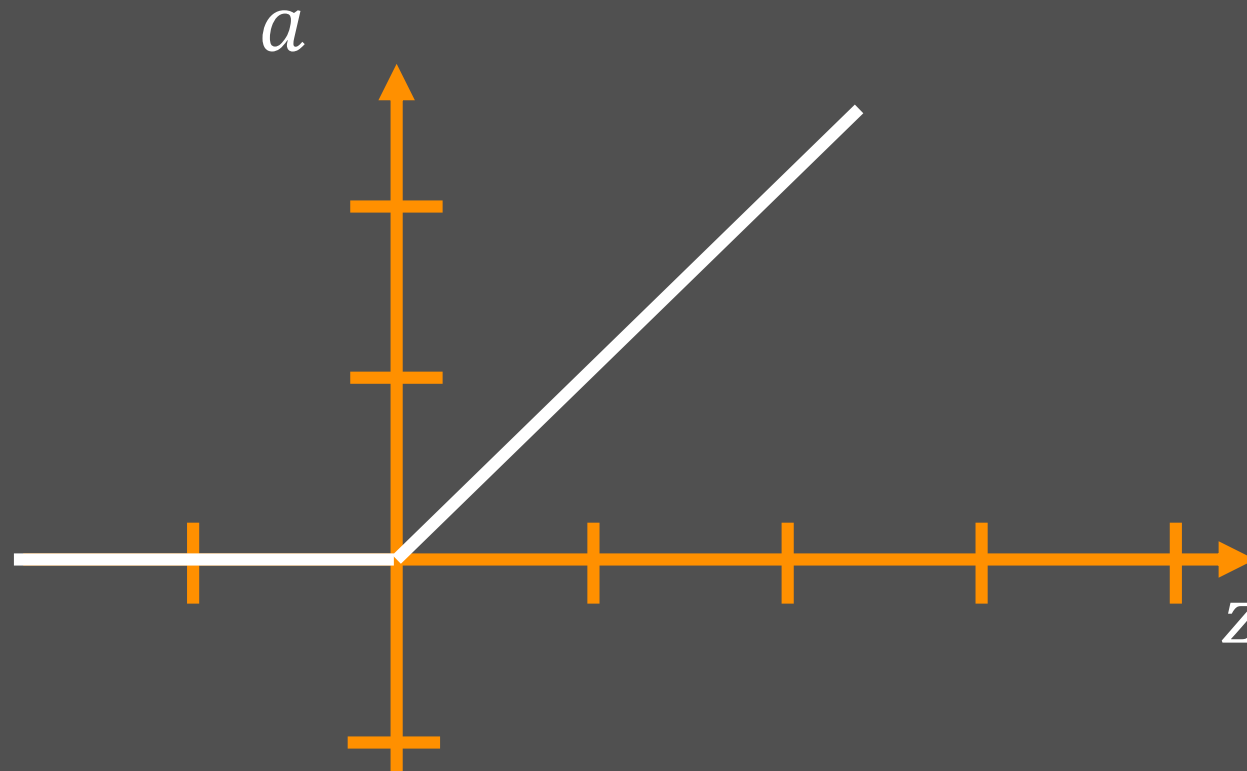
Hidden Layer Activation Functions $a(z)$:

- **Linear Neuron:** $a(z) = z$
- **Binary Threshold Unit:** $a(z) = \begin{cases} 0, & z < 0 \\ 1, & z \geq 0 \end{cases}$
- **Rectified Linear (ReLU):** $a(z) = \begin{cases} 0, & z < 0 \\ z, & z \geq 0 \end{cases}$
- **Sigmoid:** $a(z) = \frac{1}{1+e^{-z}}$
- **Tanh:** $a(z) = \tanh(z)$

Linear Neuron: $a(z) = z$

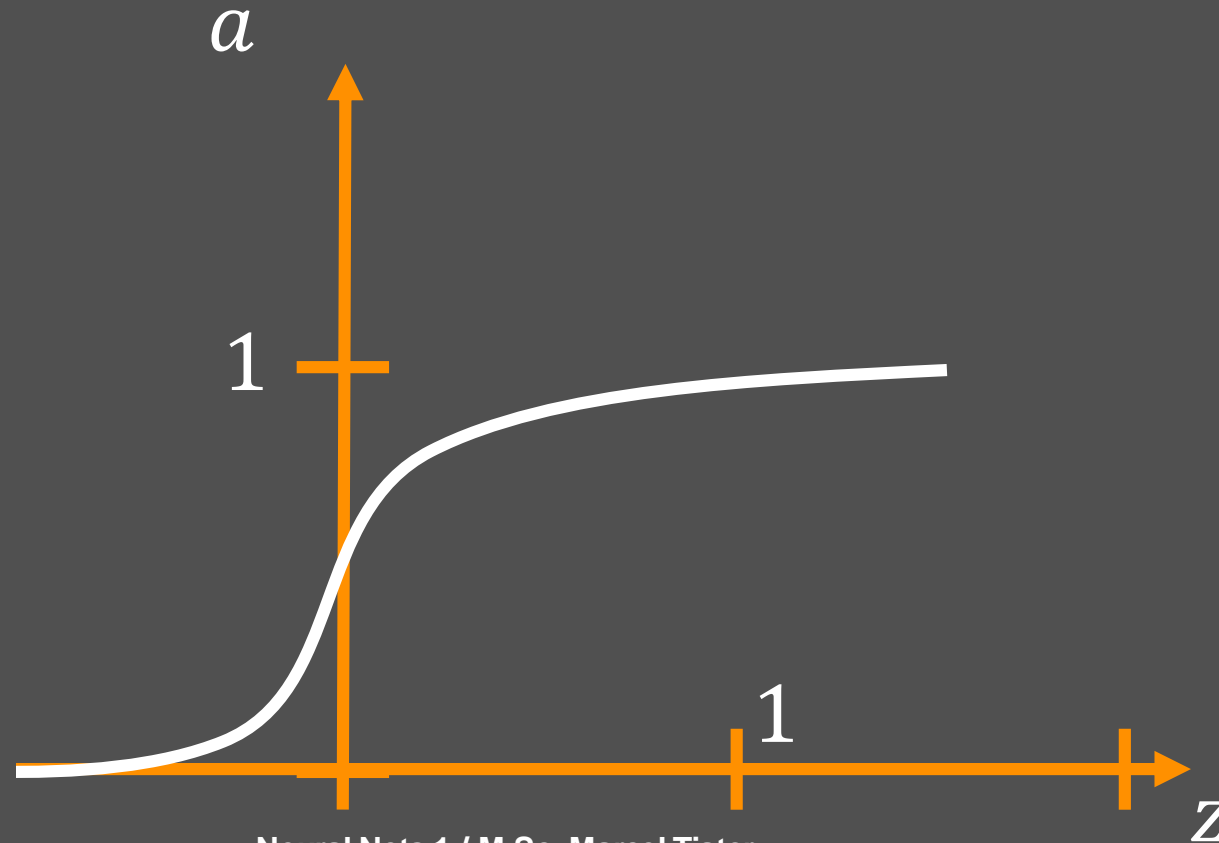


Rectified Linear (ReLu): $a(z) = \begin{cases} 0, & z < 0 \\ z, & z \geq 0 \end{cases}$

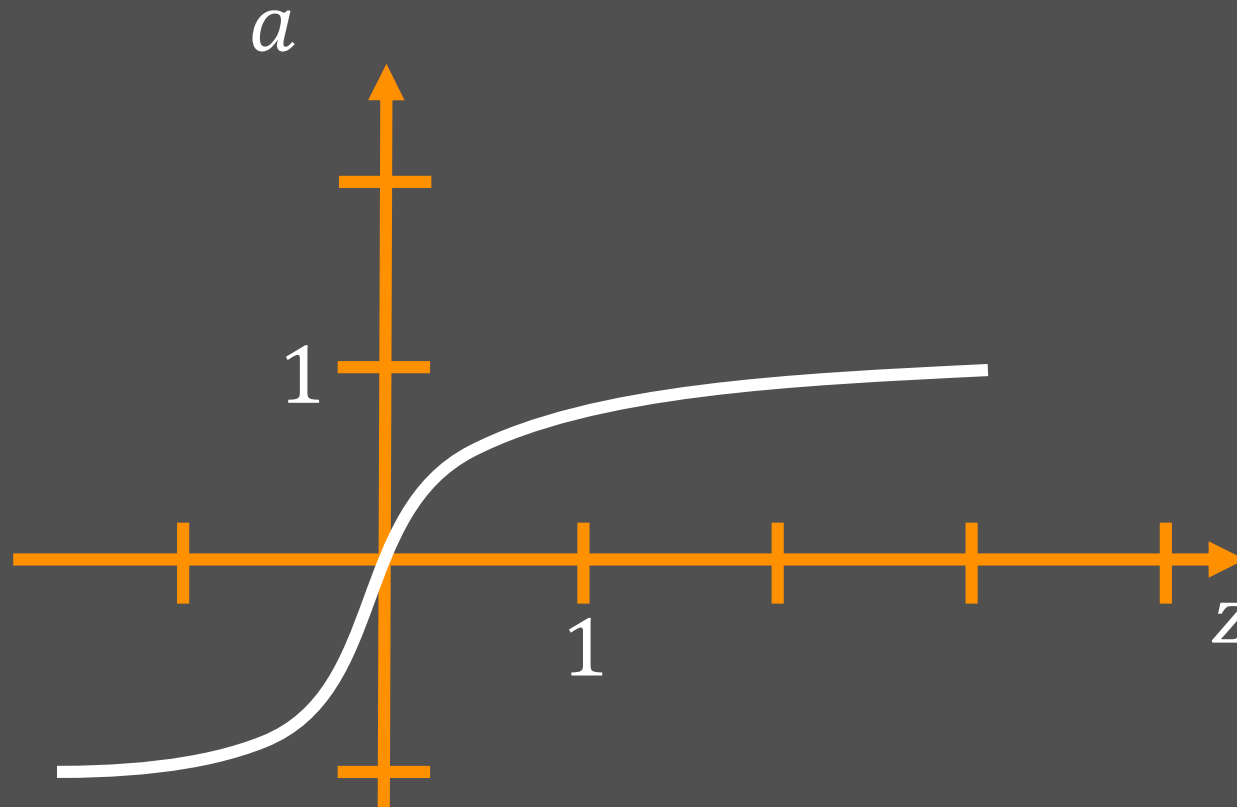


Forward Propagation

- **Sigmoid:** $a(z) = \frac{1}{1+e^{-z}}$

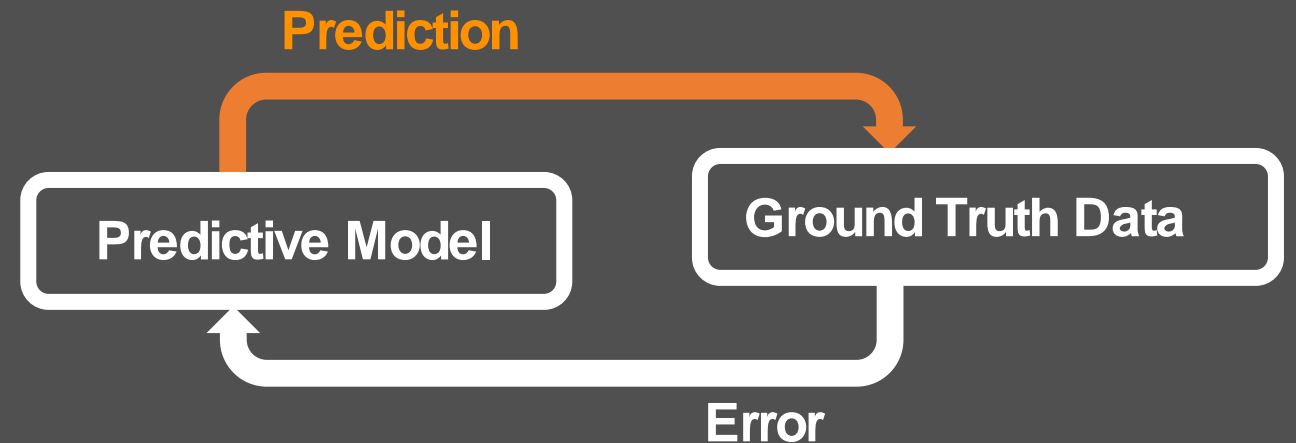
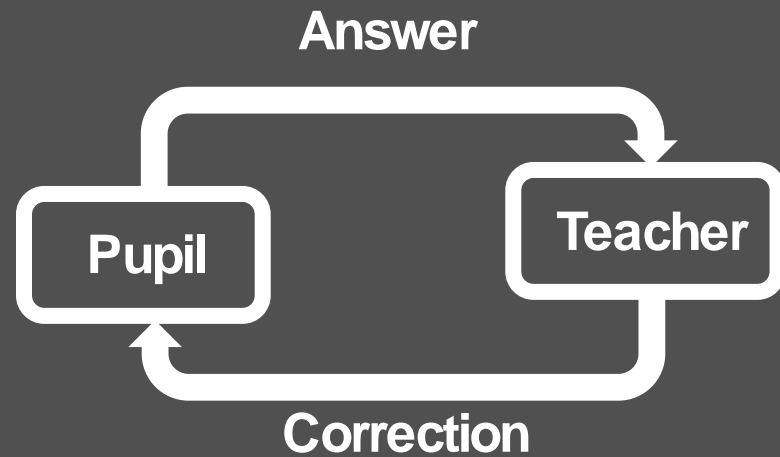


- **Tanh:** $a(z) = \tanh(z)$



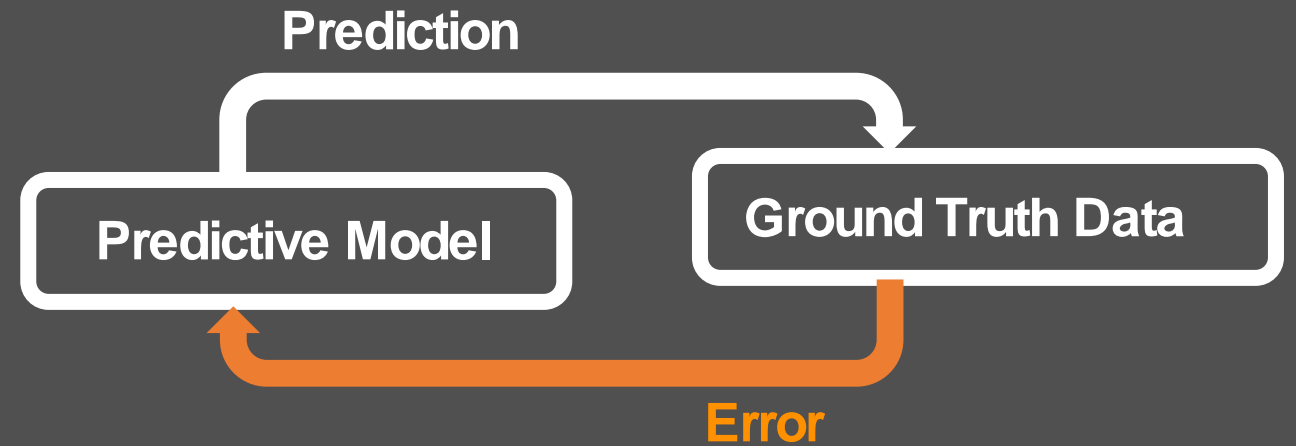
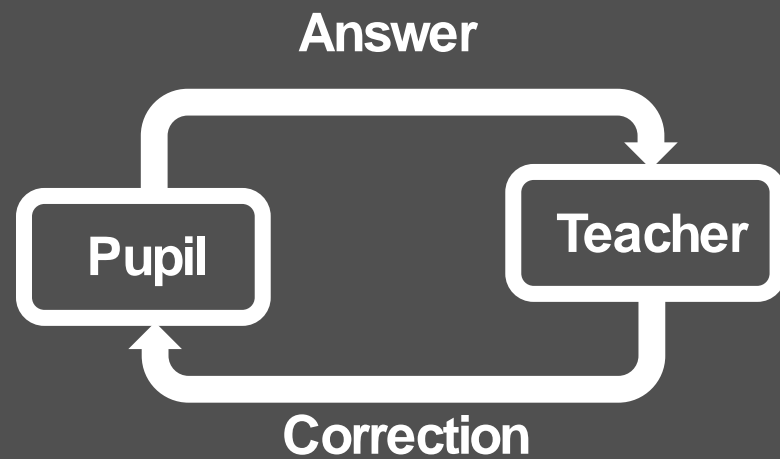
Two Processes:

- **Forward Propagation (Prediction)**
- **Backward Propagation (Weight Tuning)**



Two Processes:

- Forward Propagation (Prediction)
- **Backward Propagation (Weight Tuning)**

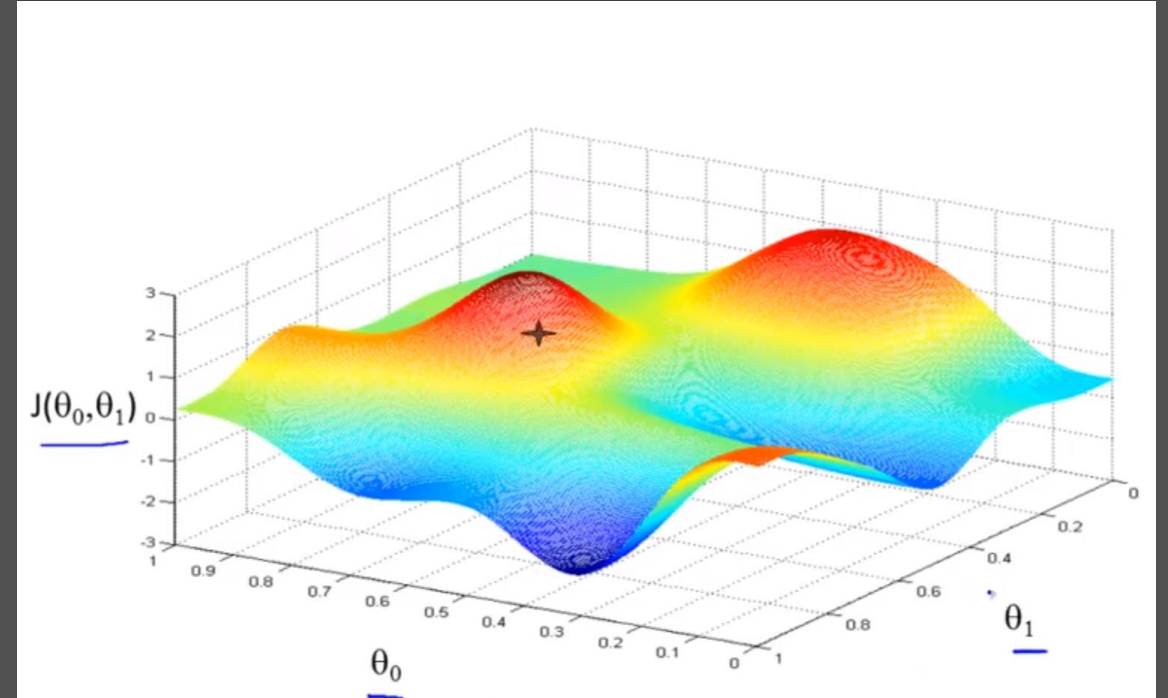
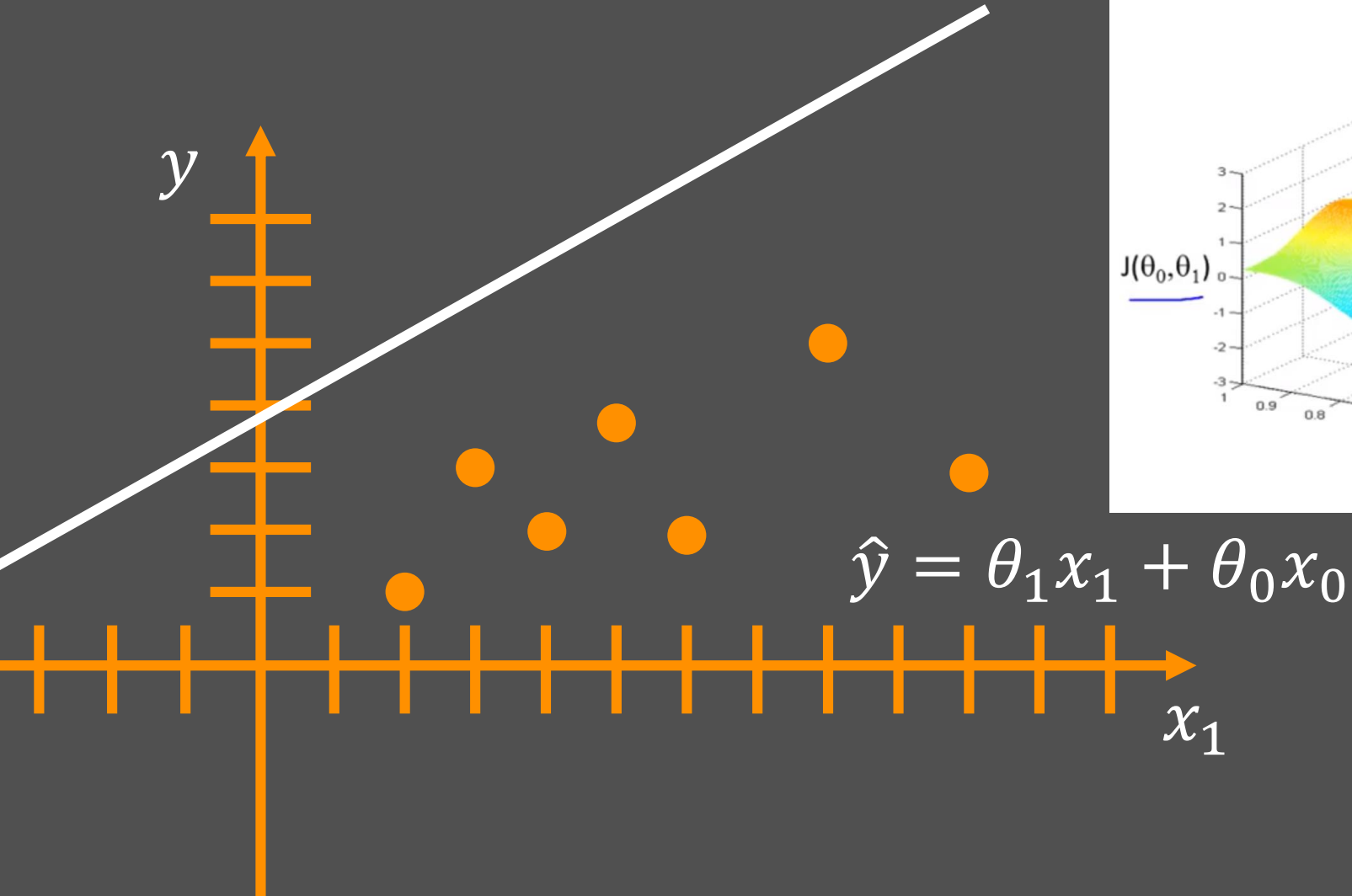


Training of MLP:

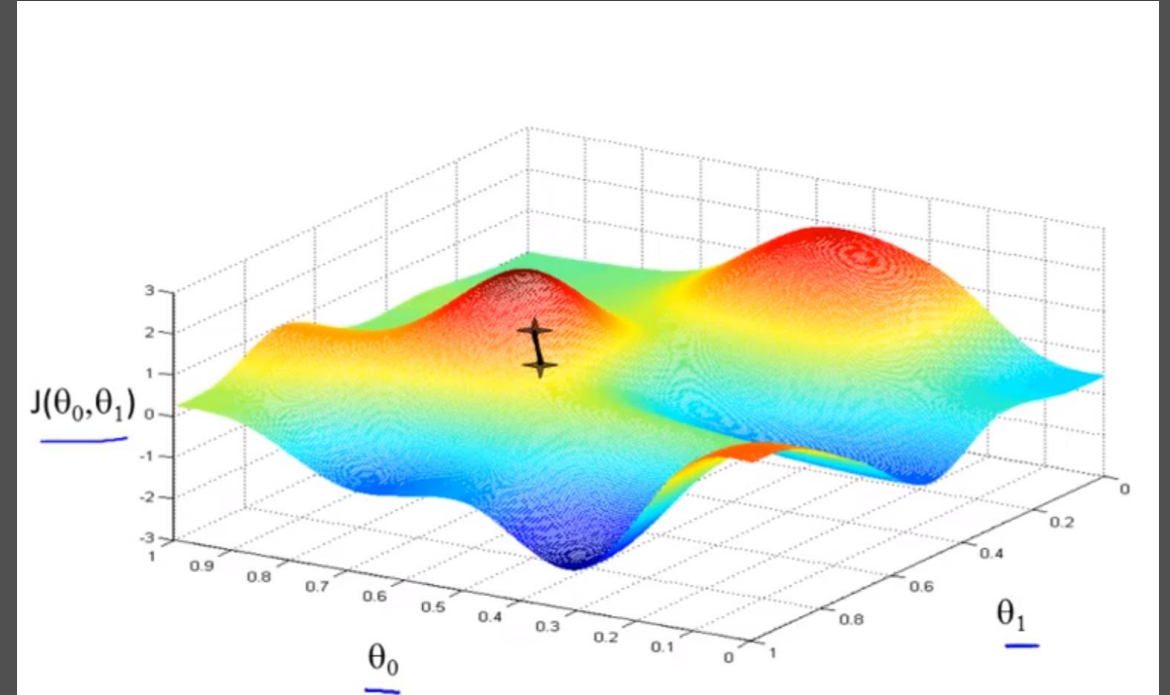
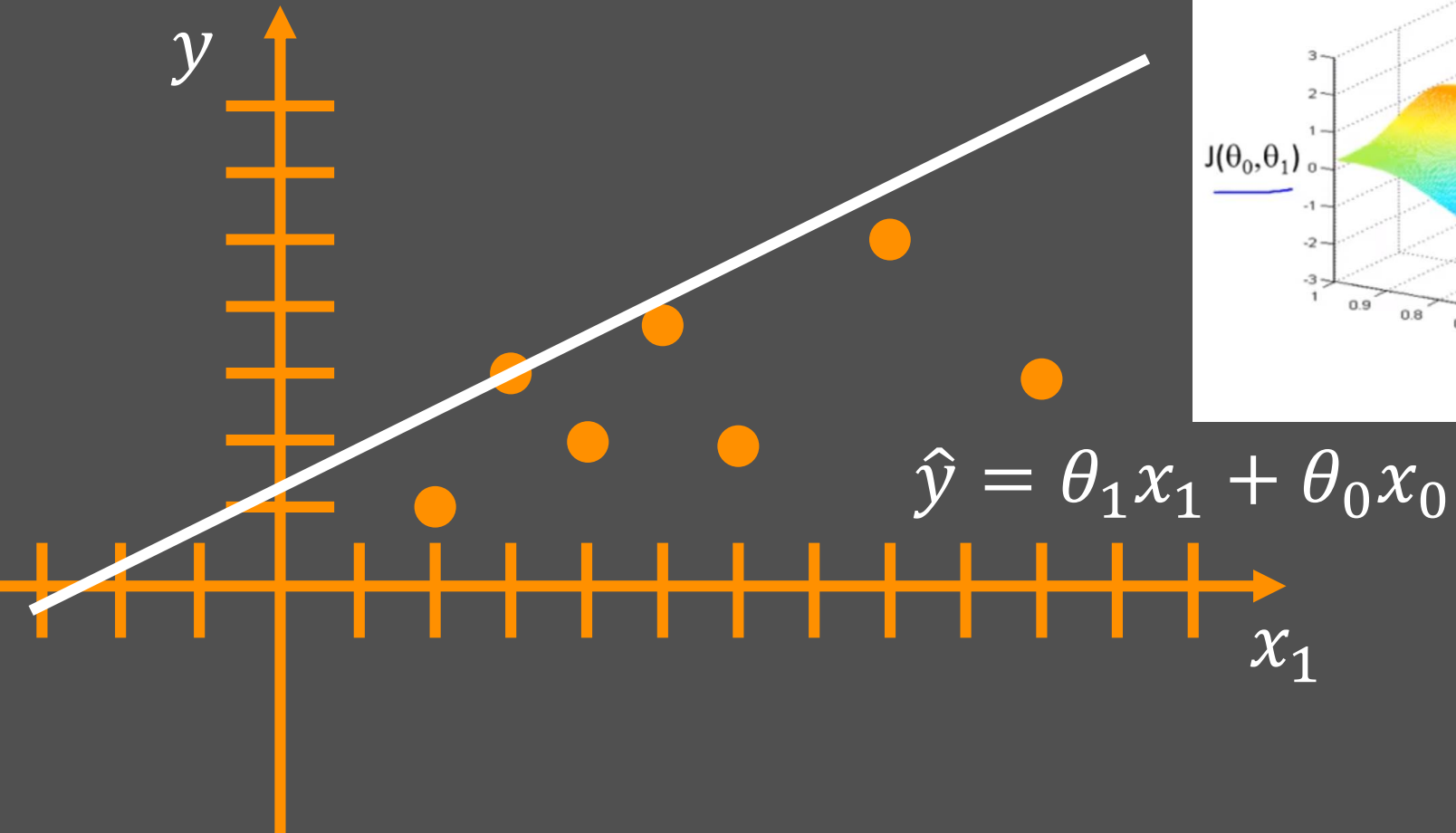
- Use Gradient-Based Algorithm: **Backpropagation**
- Similar to Gradient Descent
- Weight-Update is more complex

- Assume, we get some Output \hat{y} from MLP
- Compute function $E = \frac{1}{2m} \sum_{t=1}^m (\hat{y}_t - y_t)^2$
- Want to $\min_{\Theta_{i,j}^{(l)}} E(\Theta_{i,j}^{(l)})$ so calculate $\frac{\partial E(\Theta_{i,j}^{(l)})}{\partial \Theta_{i,j}^{(l)}}$
- Use $\frac{\partial E(\Theta_{i,j}^{(l)})}{\partial \Theta_{i,j}^{(l)}}$ to update weights with learning rate
- $\Theta_{i,j}^{(l)} := \Theta_{i,j}^{(l)} - \alpha \frac{\partial E(\Theta_{i,j}^{(l)})}{\partial \Theta_{i,j}^{(l)}}$

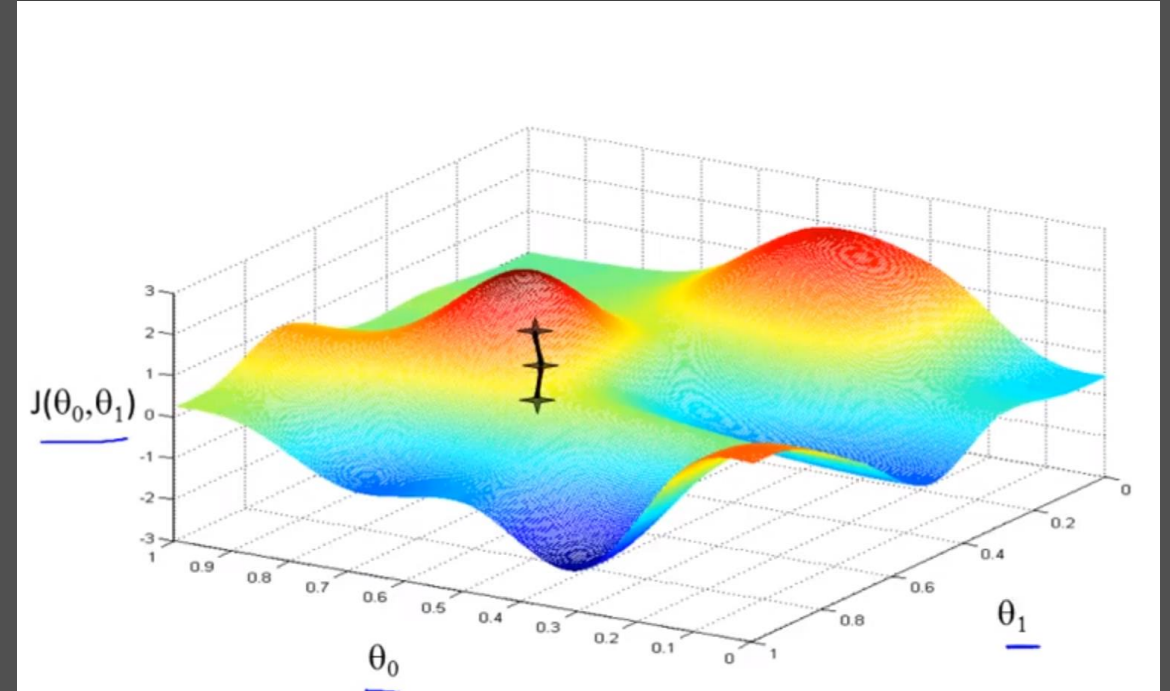
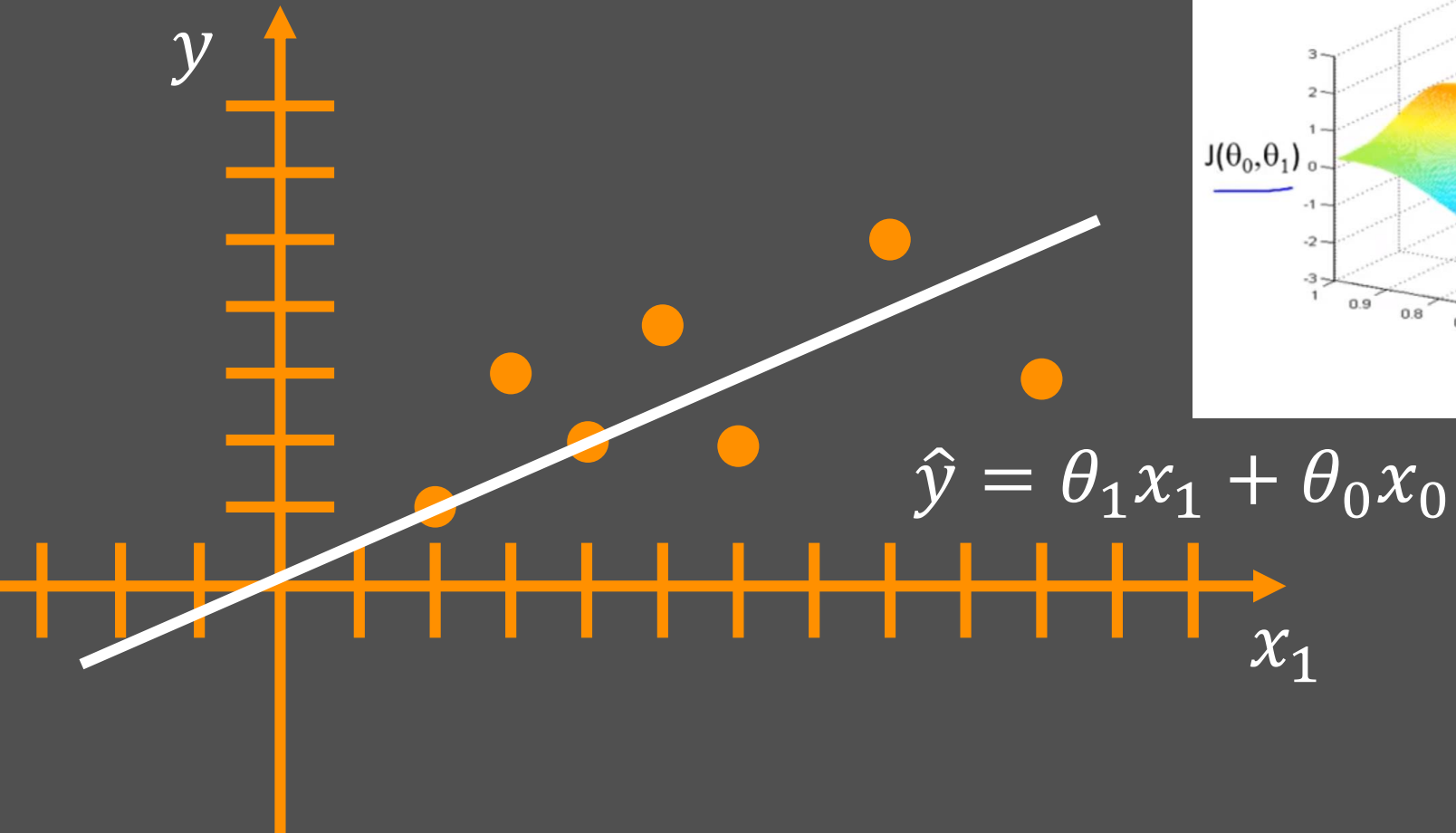
Backward Propagation



Backward Propagation



Backward Propagation



Hands-On Session

3 Scripts:

- **estimator.py**: Script that trains a **DNNClassifier**
- **load_data.py**: Utility functions to load data
- **predict.py**: Real world prediction with unlabeled data

Your task:

- Fill in the gaps which are highlighted with **TODO**
- Use internet and documentation

- **Pull code & slides from repository**
 - <https://github.com/mati3230/modalg181>
- **Download dataset from Nextcloud**
 - <https://nextcloud.mirevi.medien.hs-duesseldorf.de/index.php/s/kPXwJiac7vTQVeu>