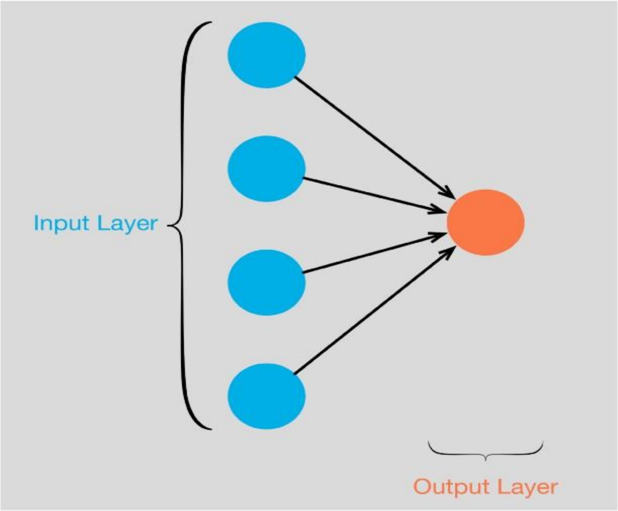
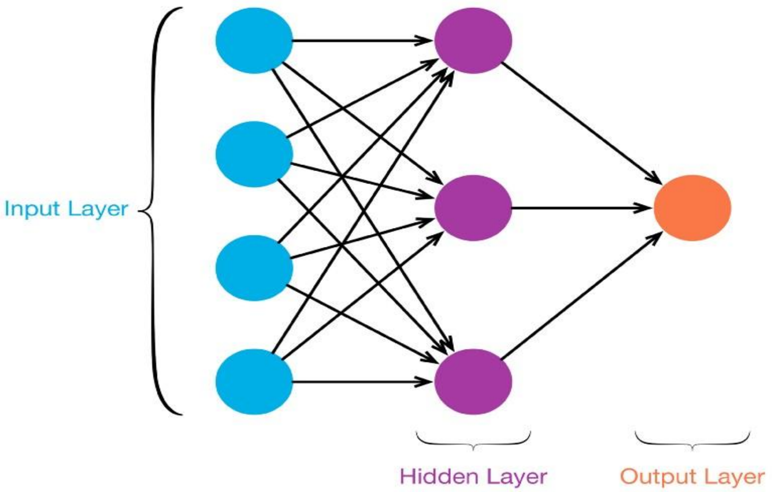
Deep Learning

Perceptron -



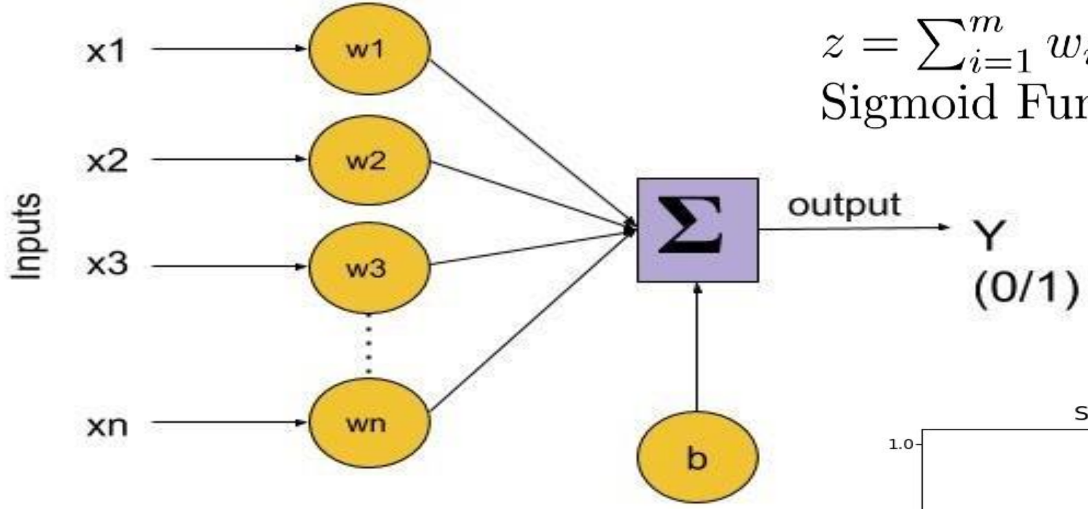
Single Layer Perceptron

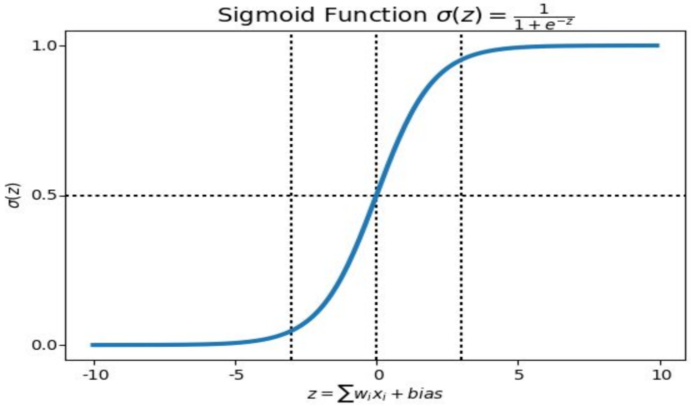


MultiLayer Perceptron

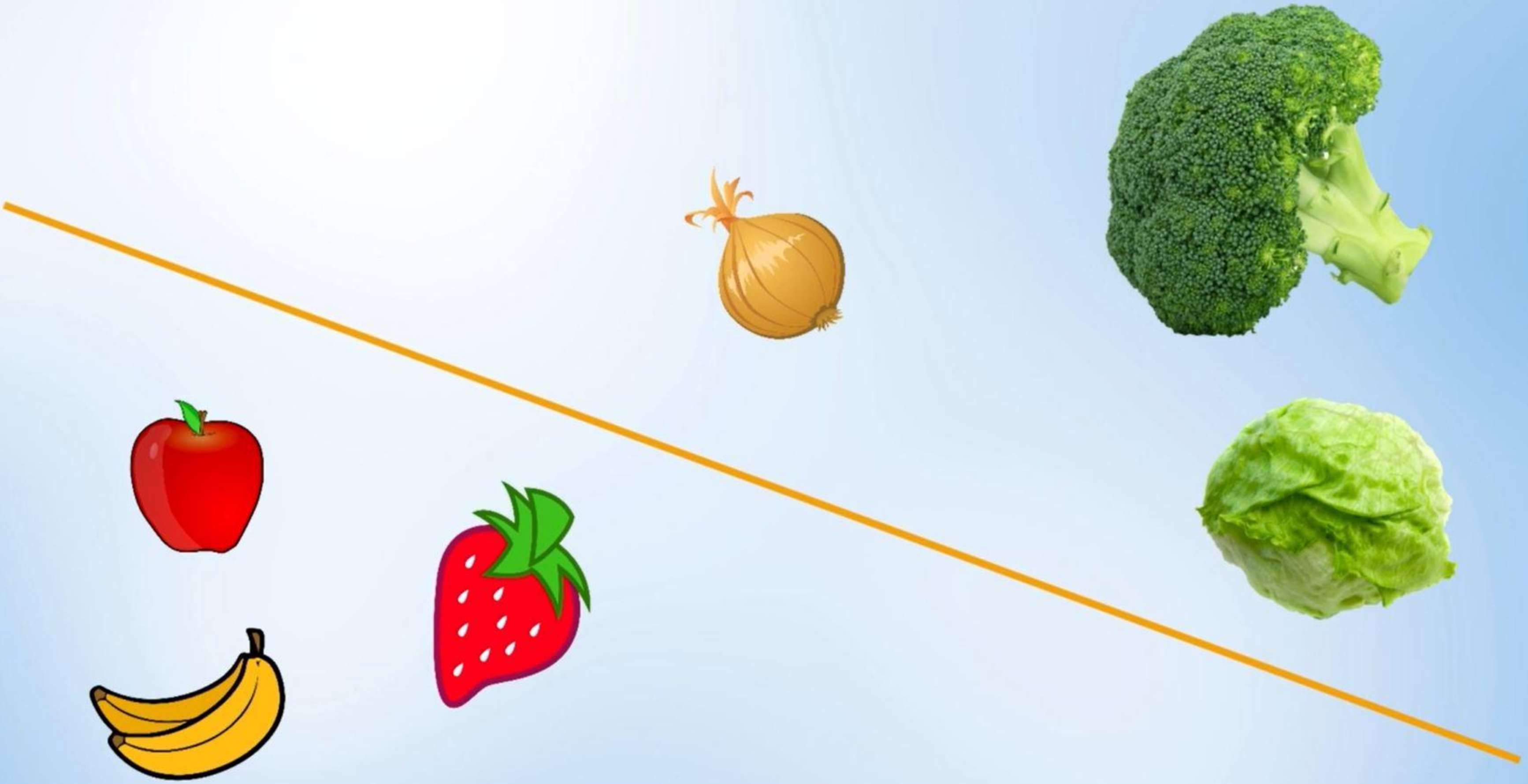
$$z = \sum_{i=1}^{m} w_i x_i + bias$$

Sigmoid Function is: $\sigma(z) = \frac{1}{1+e^{-z}}$





Classification



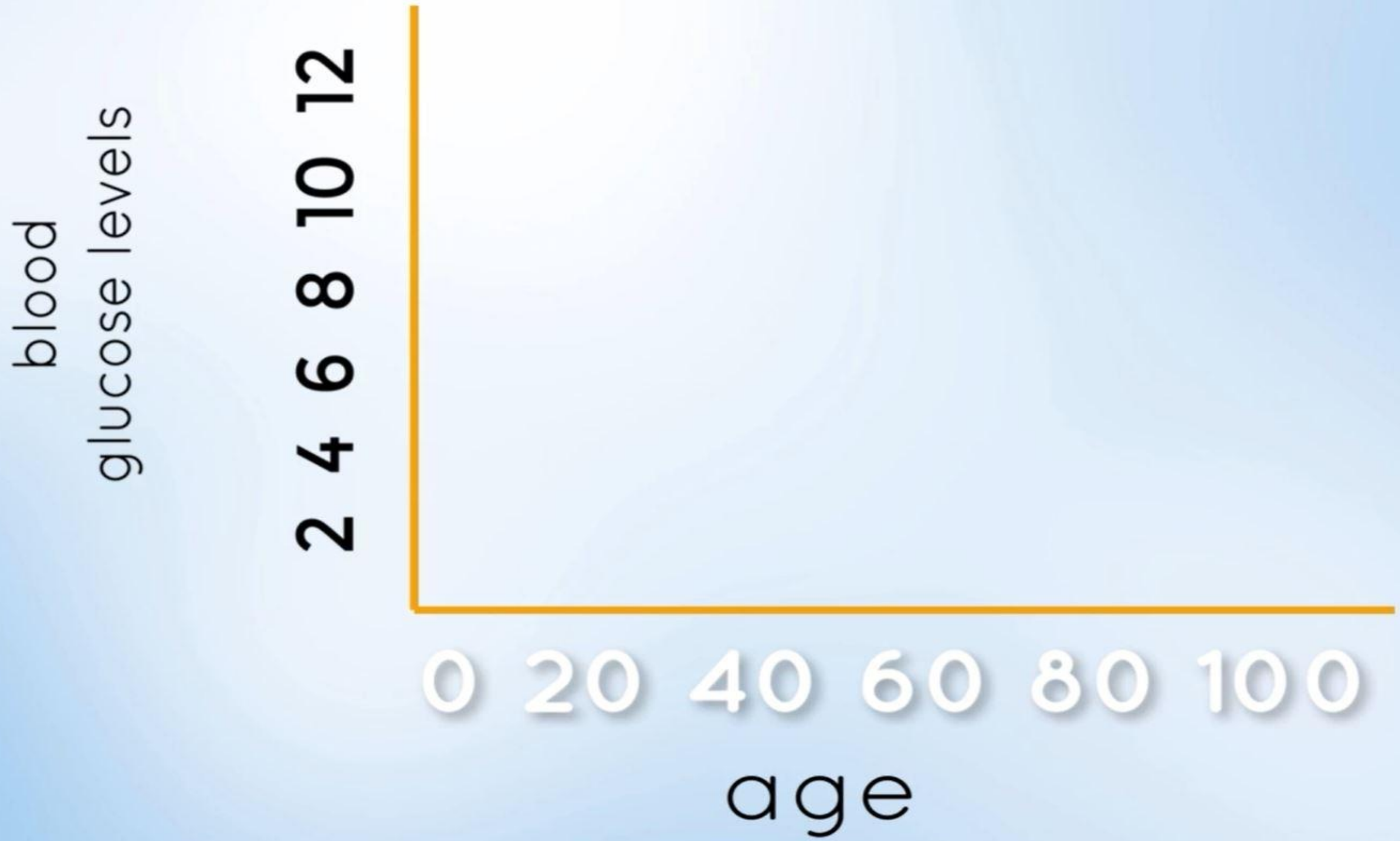
Classification

2 mmol/L, 20 years old

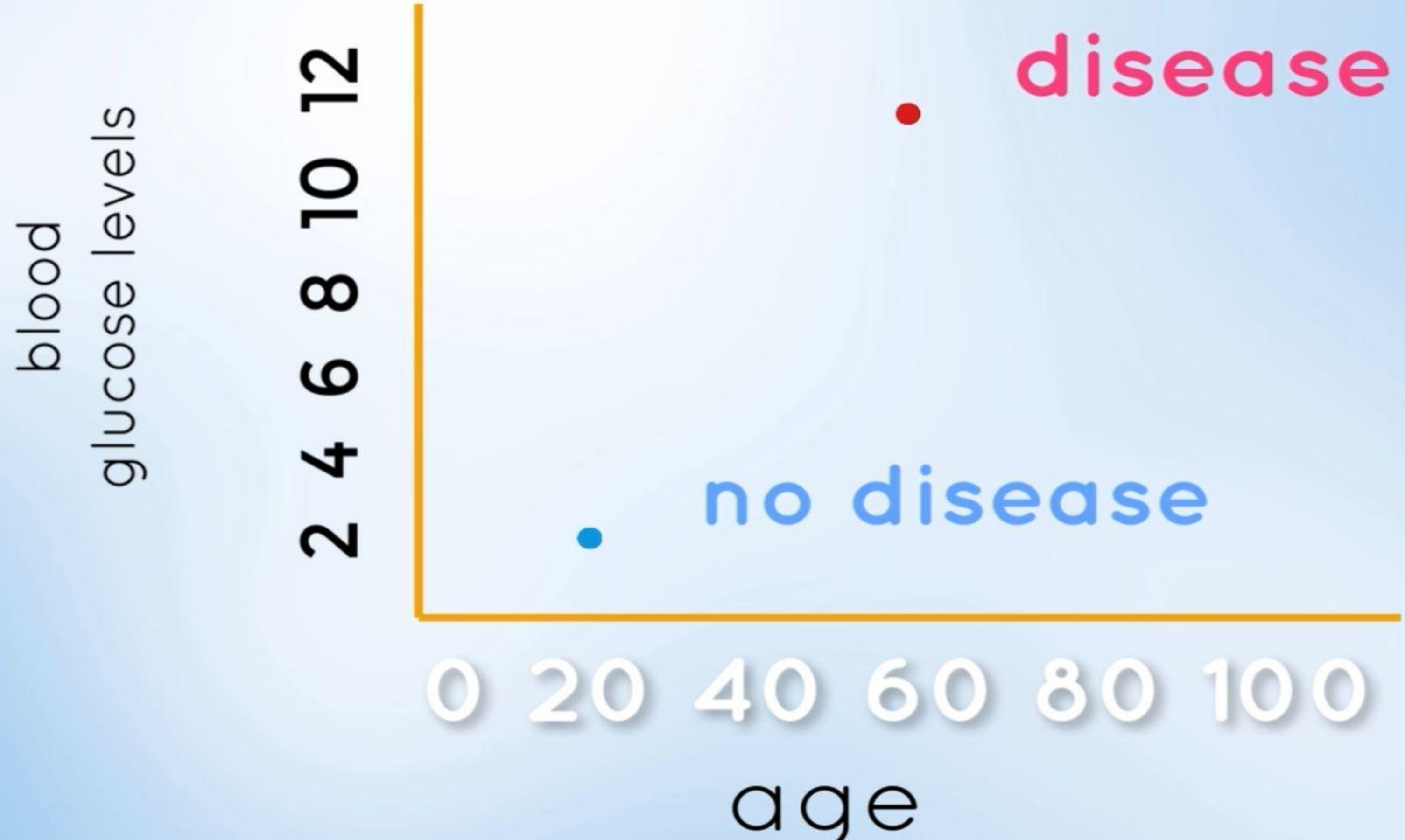
11 mmol/L, 60 years old

5 mmol/L, 45 years old

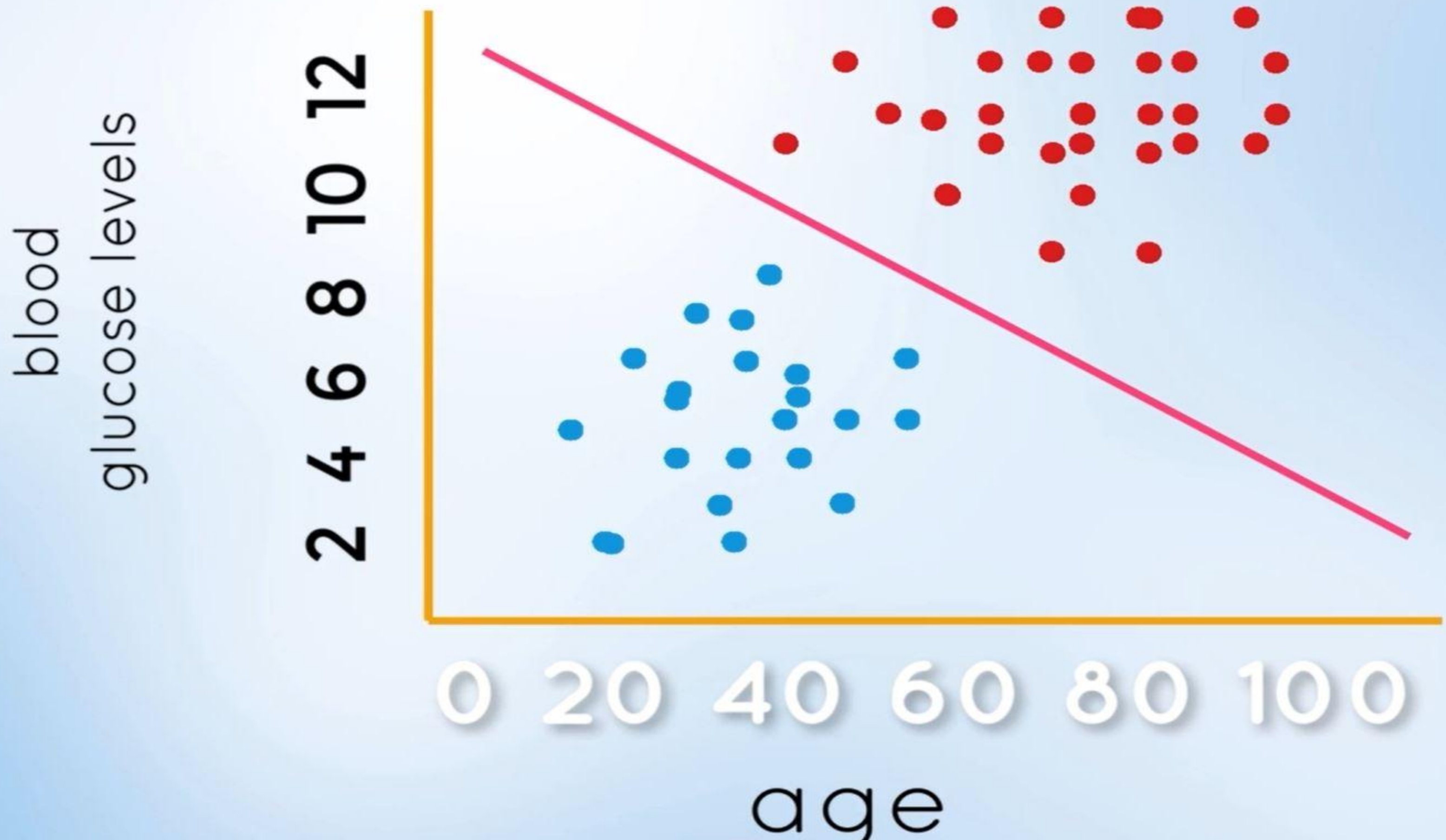
Classification



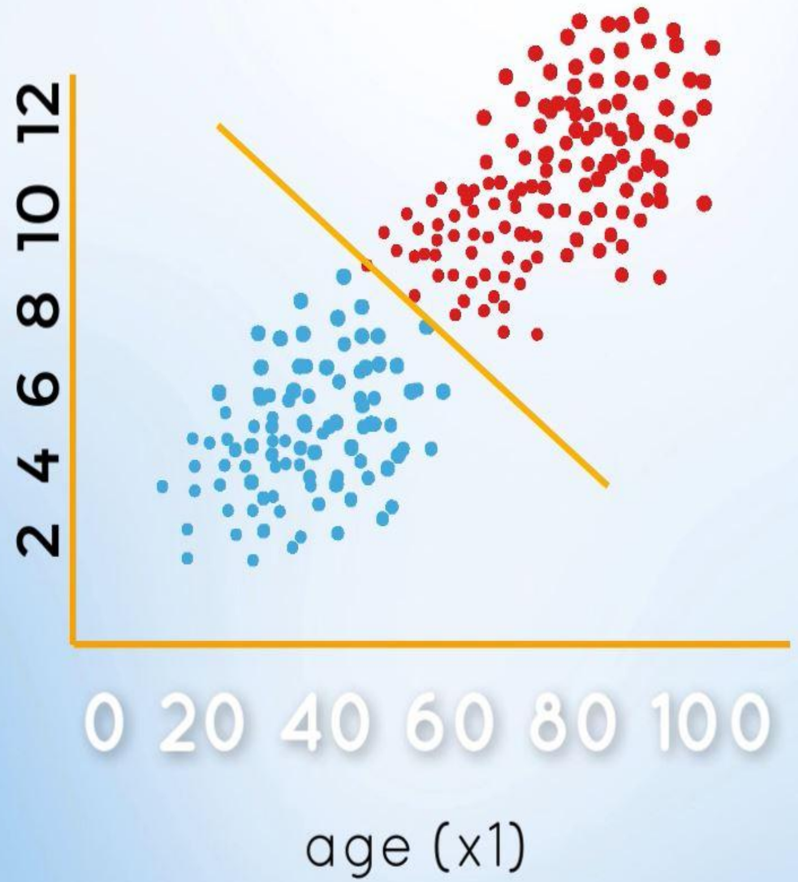
Classification

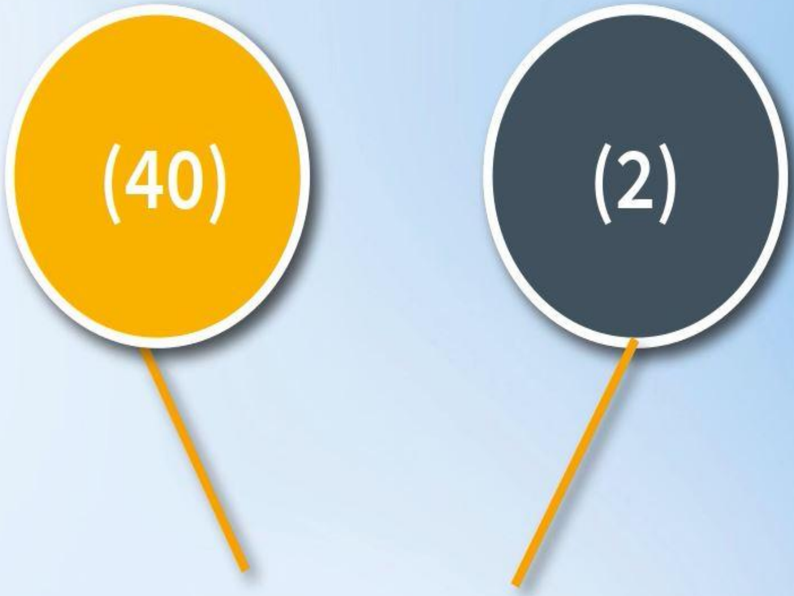


Classification



$$w1x1 + w2x2 + b = 0$$

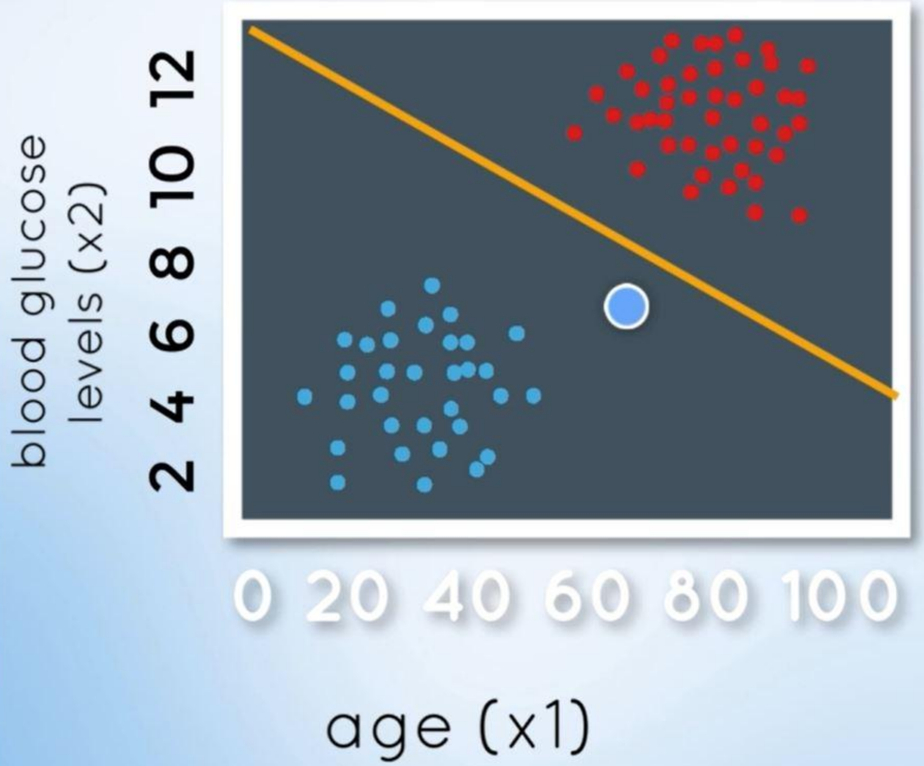




$$-0.1(40) - 1(2) + 14 = 8$$

$$\hat{y} \begin{cases} 1 \text{ if score} \geq 0 \\ 0 \text{ if score} < 0 \end{cases}$$

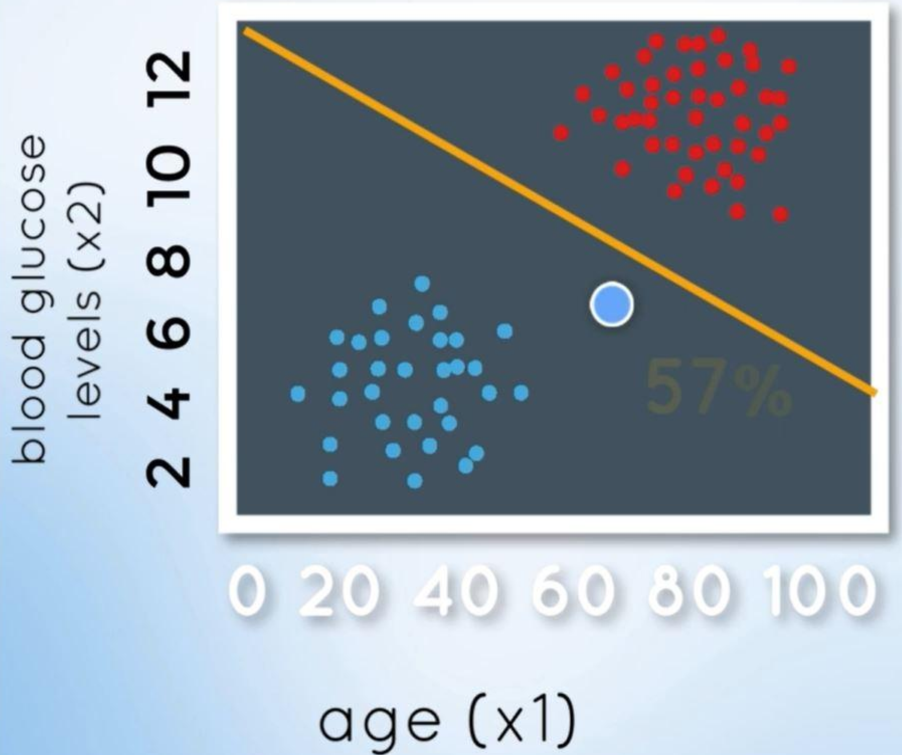
$$-0.1(62) - 7.5 + 14 = 0.3$$



$$\frac{1}{1+\frac{1}{e^{0.3}}}$$

7.5 mmol/L, 62 years old

$-0.1(62) - 7.5 + 14 = 0.3$

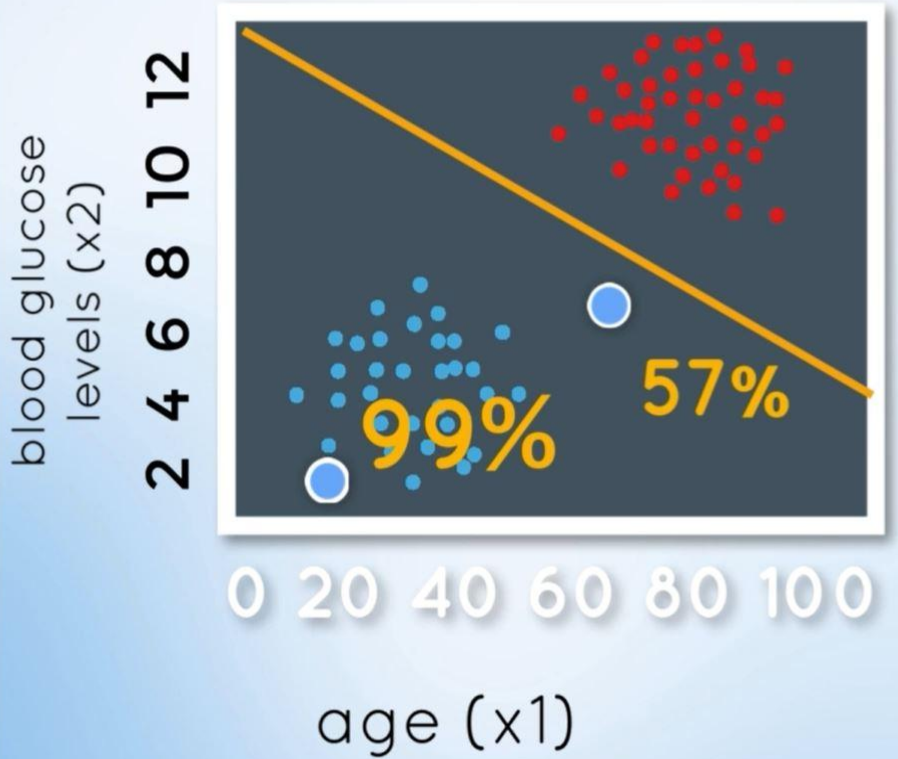


$$\frac{1}{1 + \frac{1}{e^{0.3}}}$$

57%

7.5 mmol/L, 62 years old

$-0.1(20) - 2 + 14 = 10$

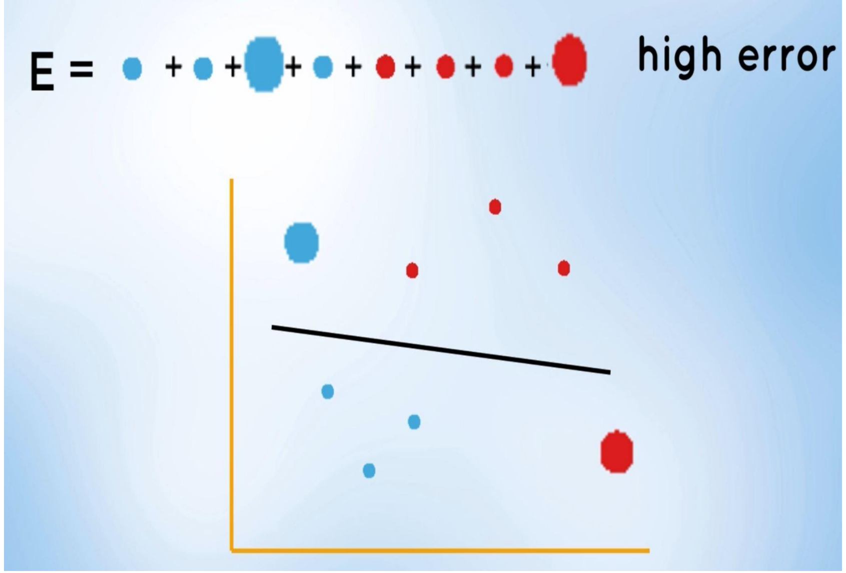


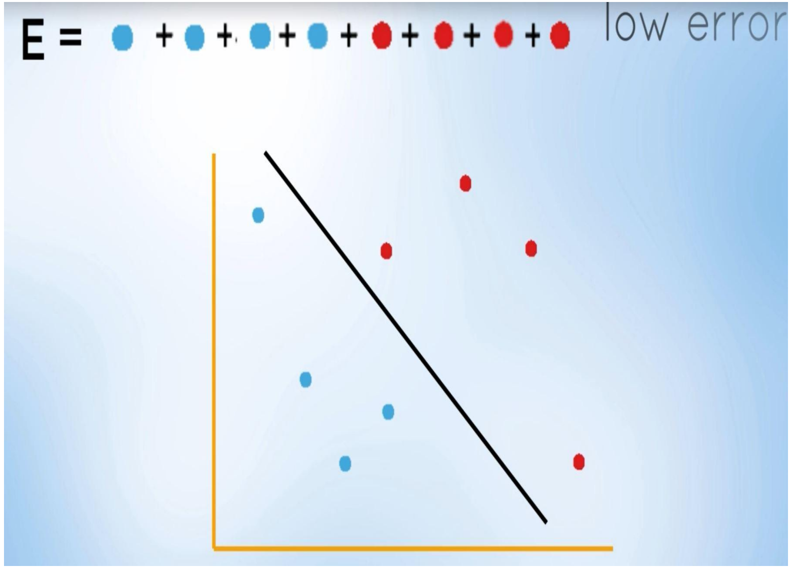
$$\frac{1}{1+\frac{1}{e^{10}}}$$

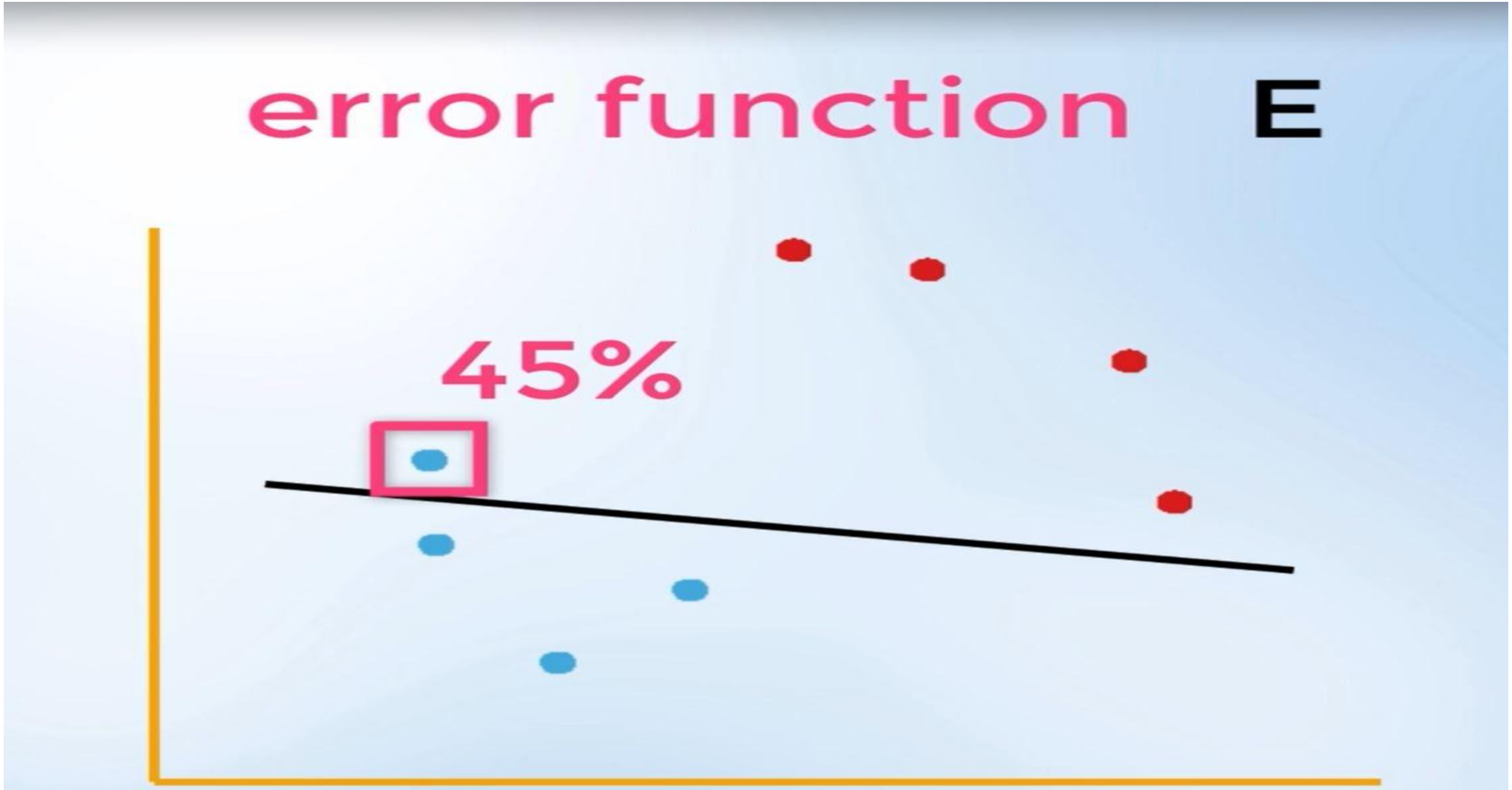
99%

2 mmol/L, 20 years old

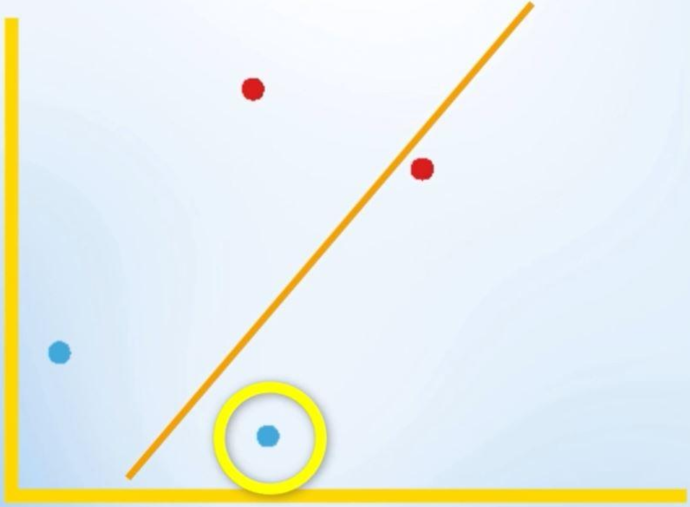
Error Function -

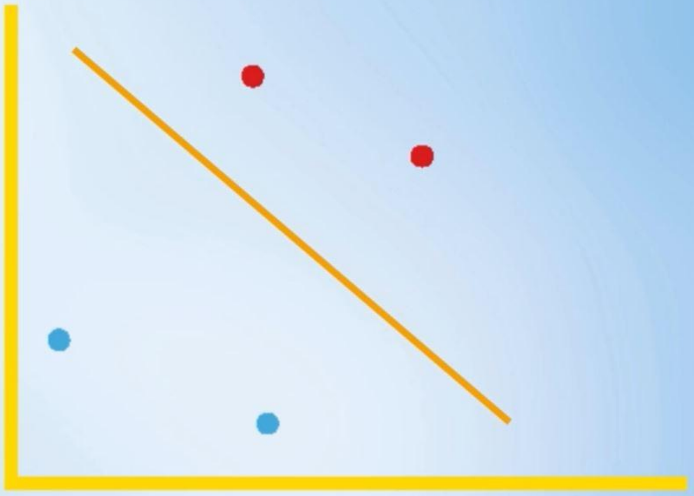


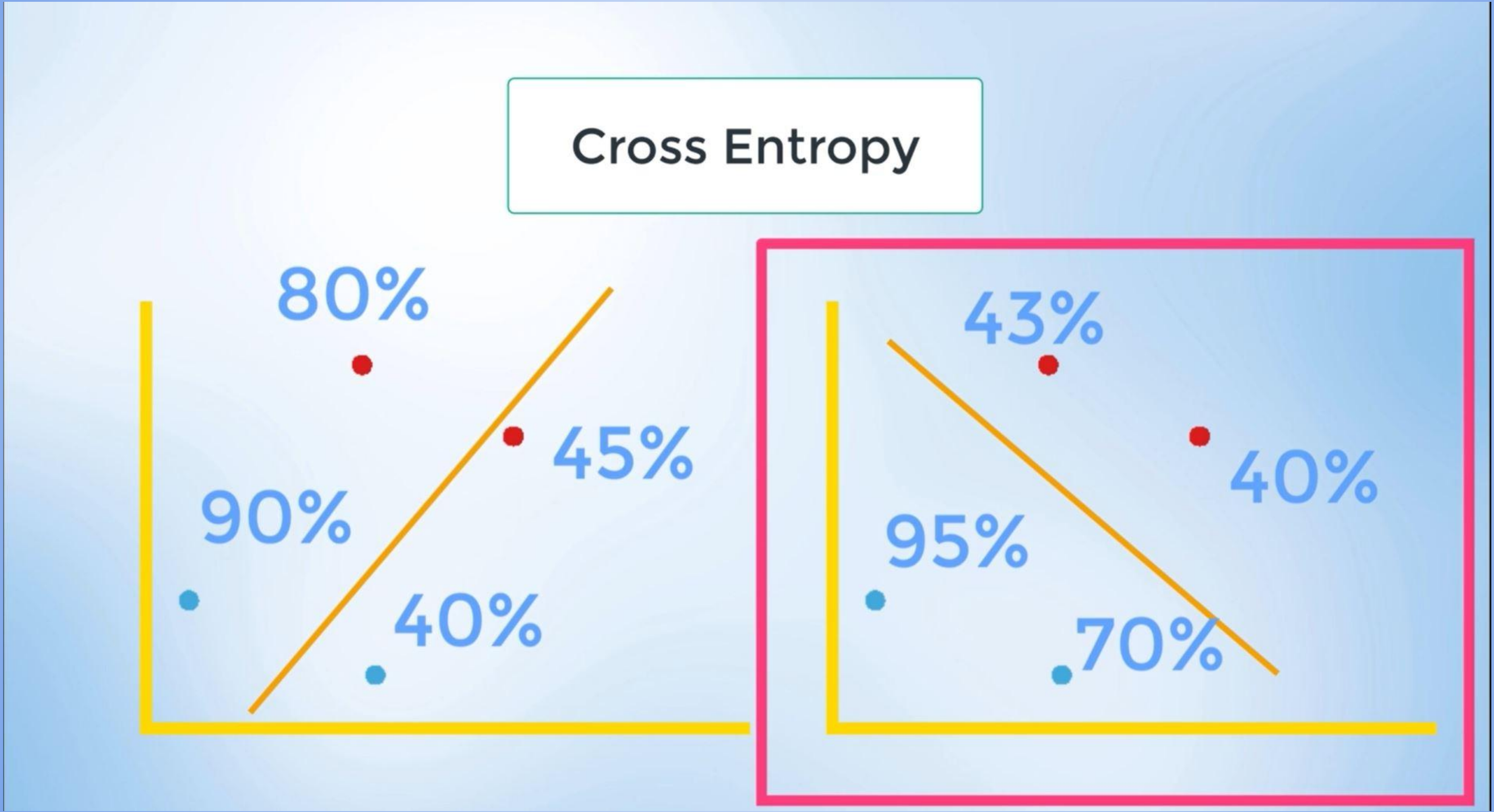



error function E
45%

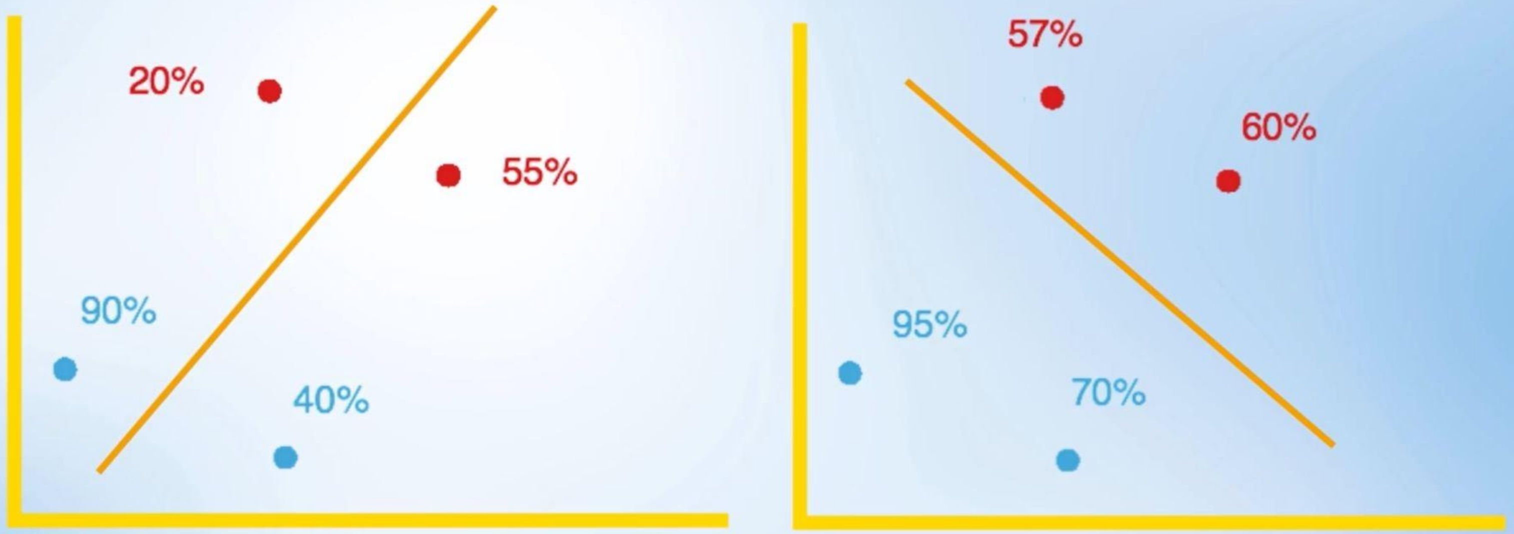
Cross Entropy



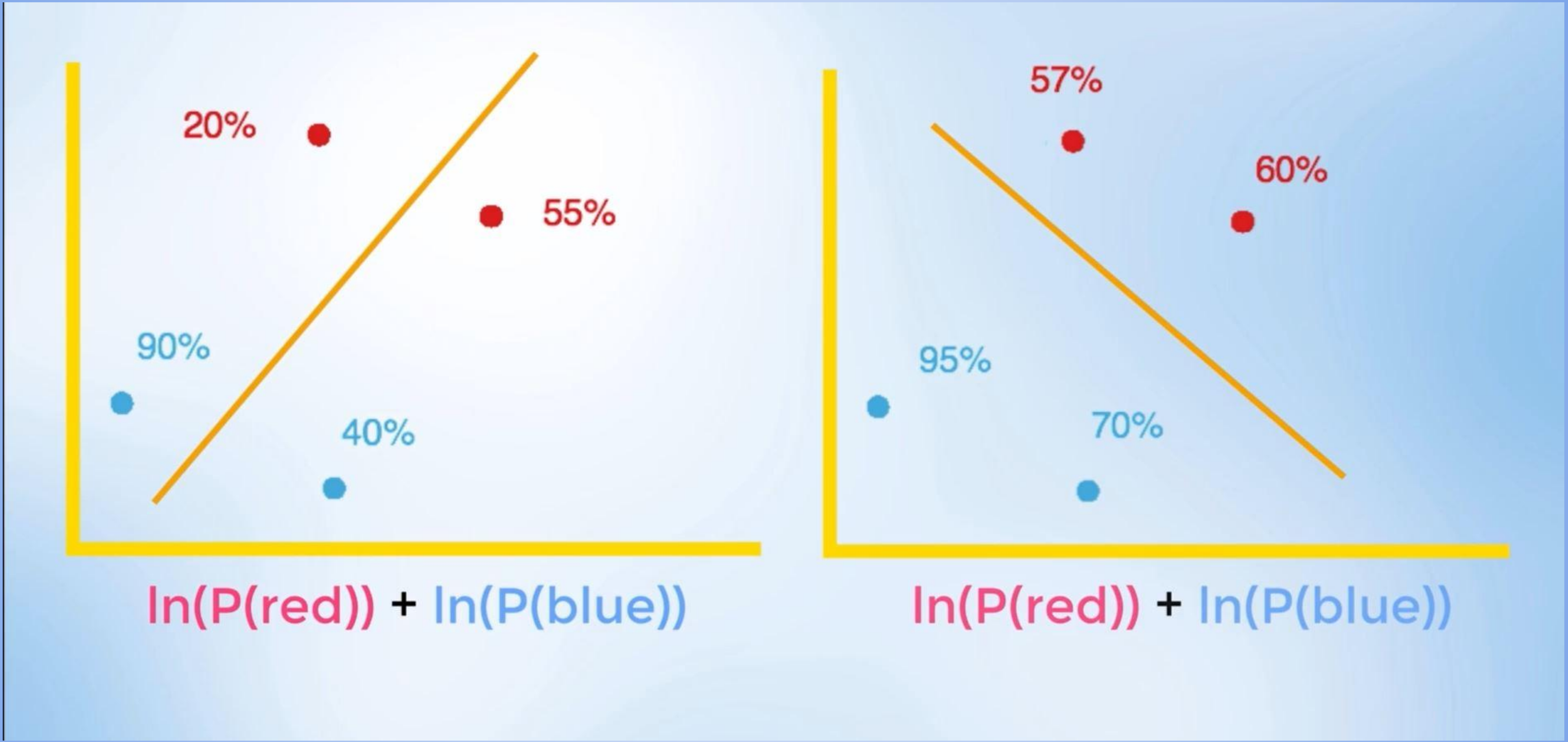




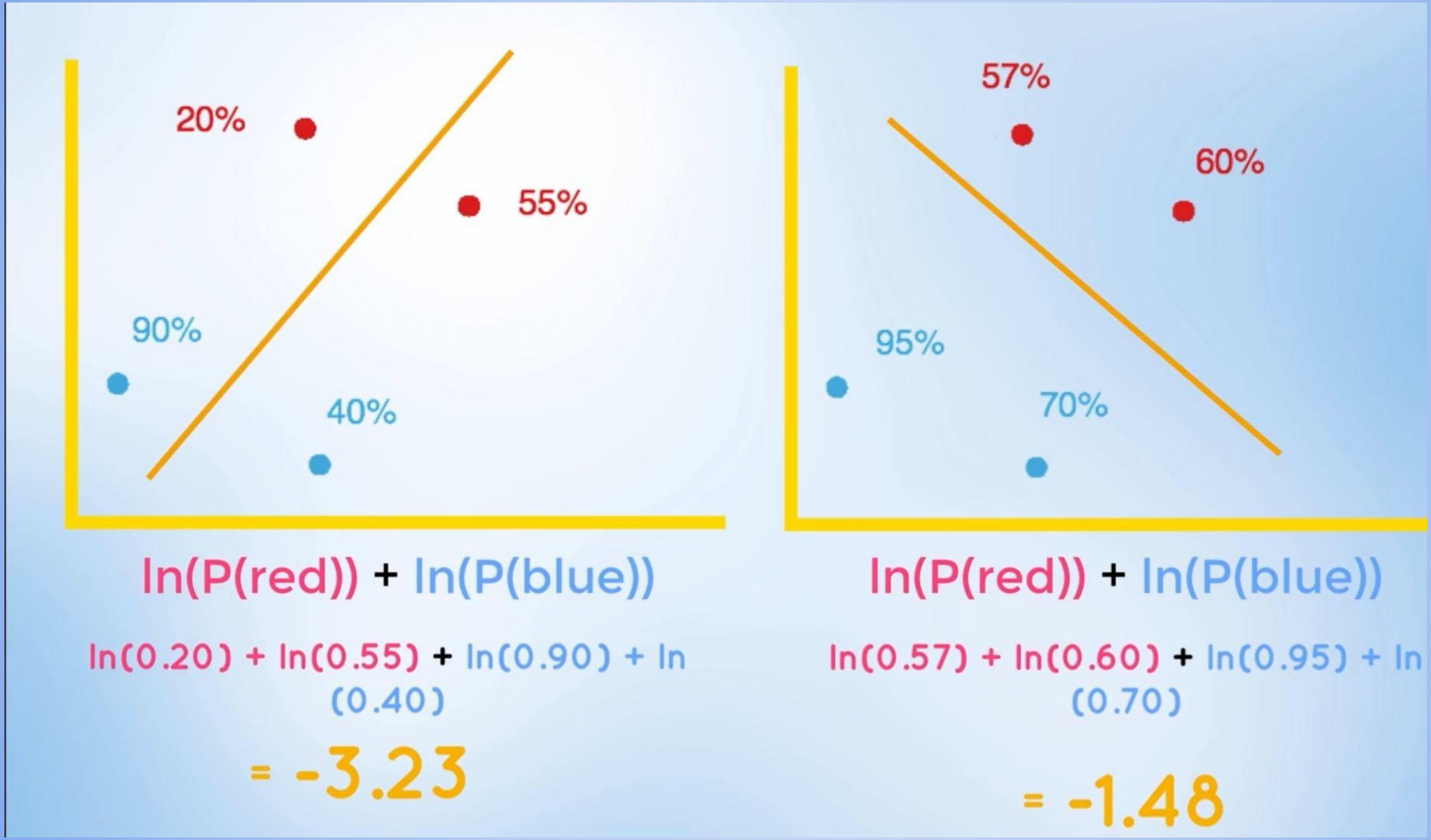
Cross Entropy
80%
45%
90%
40%
43%
40%
95%
70%



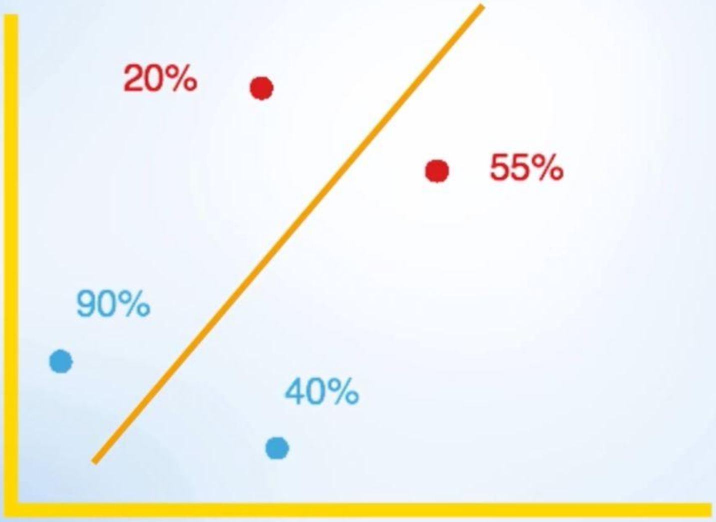
P(red) = 1 - P(blue)

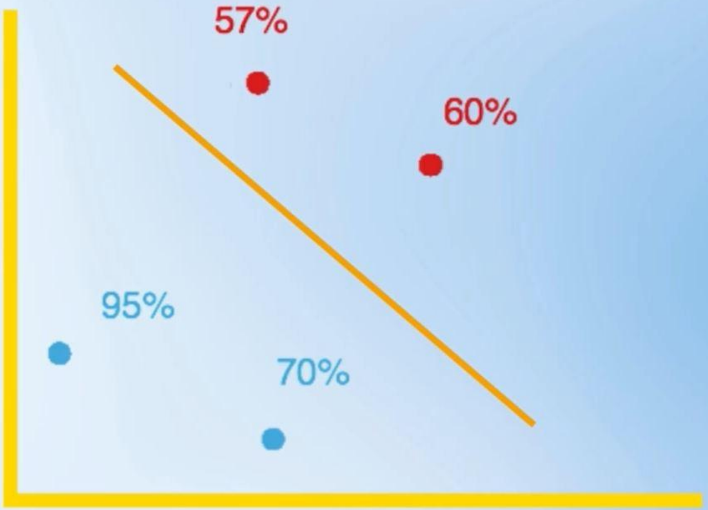


20%
55%
90%
40%
ln(P(red)) + ln(P(blue))
57%
60%
95%
70%
ln(P(red)) + ln(P(blue))



20%
55%
90%
40%
ln(P(red)) + ln(P(blue))
ln(0.20) + ln(0.55) + ln(0.90) + ln (0.40)
= -3.23
57%
60%
95%
70%
ln(P(red)) + ln(P(blue))
ln(0.57) + ln(0.60) + ln(0.95) + ln (0.70)
= -1.48





$\ln(P(\text{red})) + \ln(P(\text{blue}))$

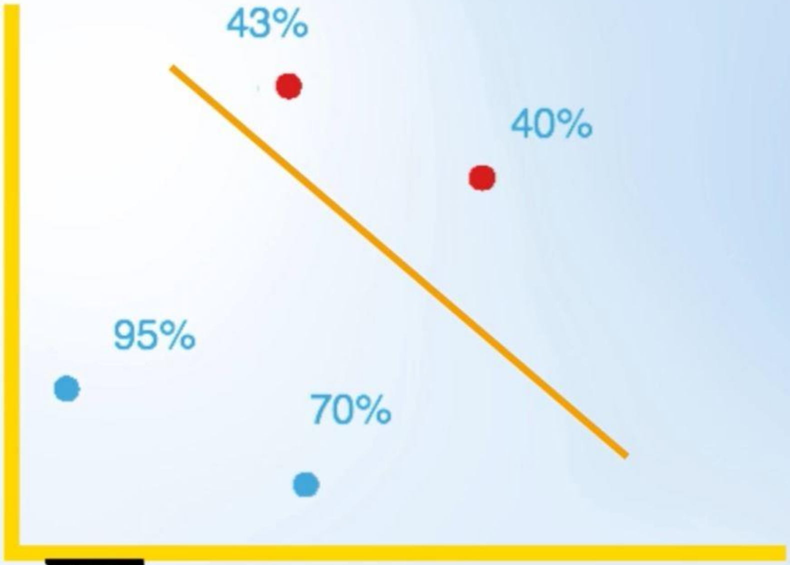
$-[\ln(0.20) + \ln(0.55) + \ln(0.90) + \ln(0.40)]$

$\boxed{= 3.23}$

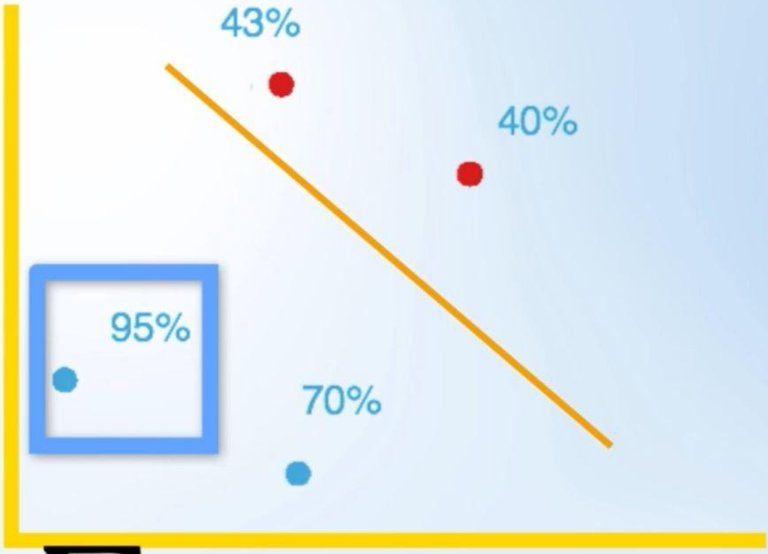
$\ln(P(\text{red})) + \ln(P(\text{blue}))$

$-[\ln(0.57) + \ln(0.60) + \ln(0.95) + \ln(0.70)]$

$= 1.48$

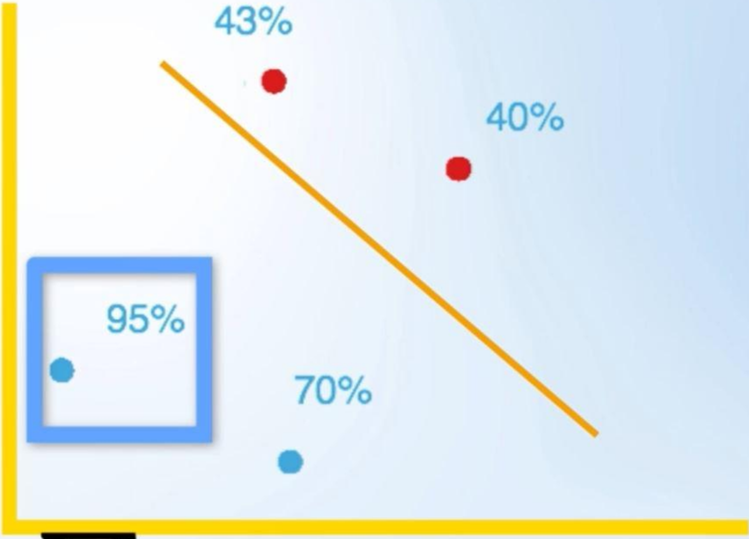


$$-\sum y\ln(p) + (1-y)(\ln(1-p))$$



$$-\sum y\ln(p) + (1-y)(\ln(1-p))$$

$$1*\ln(0.95) + (1-1)(\ln(0.05))$$

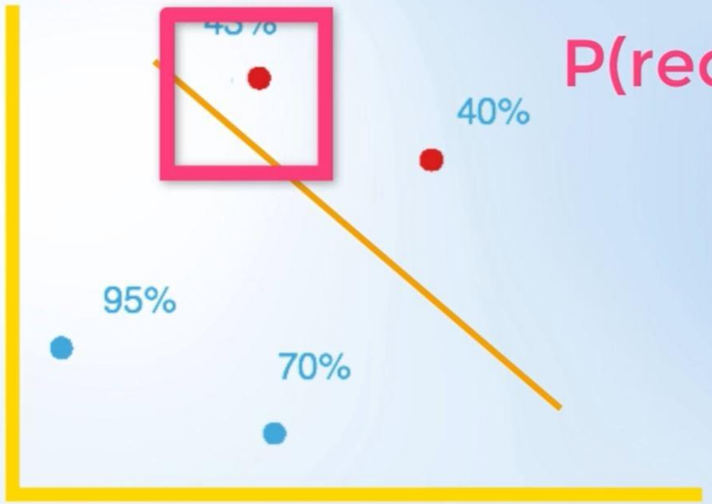


$$-\sum y\ln(p) + (1-y)(\ln(1-p))$$

$$1*\ln(0.95) + (1-1)(\ln(0.05))$$

$$\ln(0.95)$$

$P(red) = 1 - P(blue$

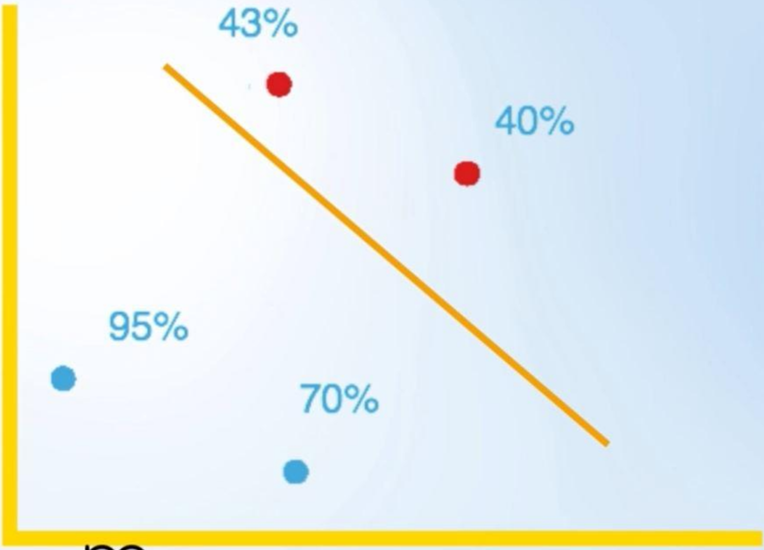


$$-\sum y\ln(p) + (1-y)(\ln(1-p))$$

$$0*\ln(0.43) + (1-0)(\ln(1-0.43))$$

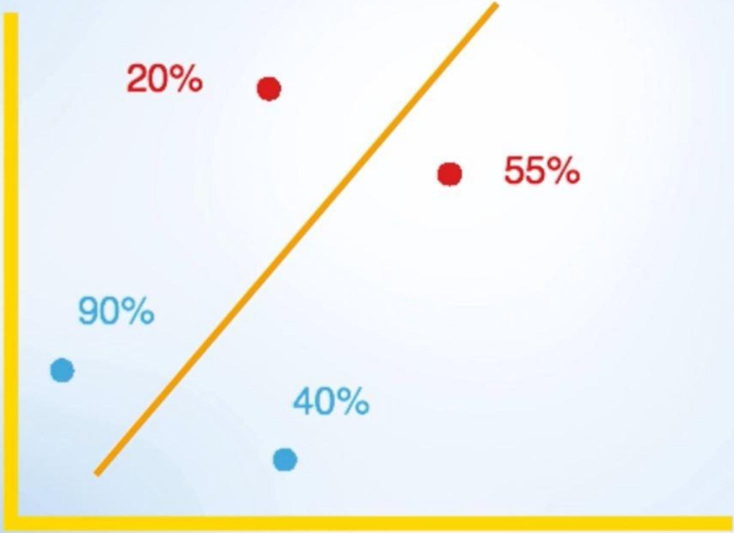
$\ln(0.95)$

$\ln(0.57)$



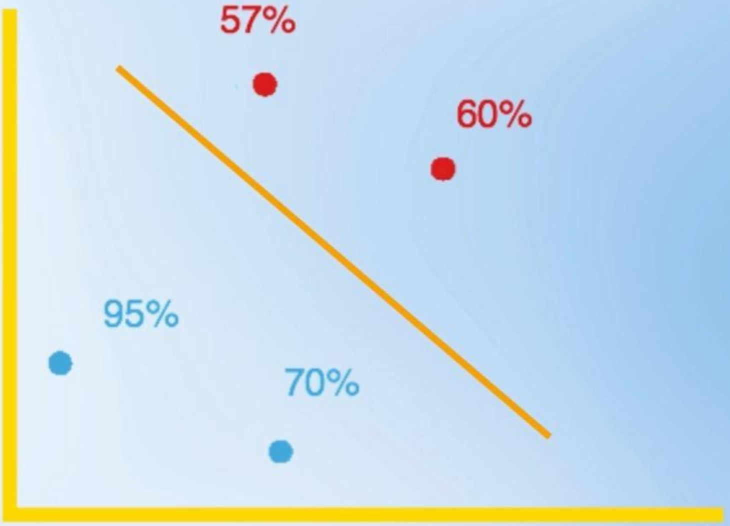
$$-\sum^{m} y\ln(p) + (1-y)(\ln(1-p))$$

$$-[\ln(0.95) + \ln(0.57) + \ln(0.70) + \ln(0.60)] = 1.48$$



$$-[\ln(0.20) + \ln(0.55) + \ln(0.90) + \ln(0.40)]$$

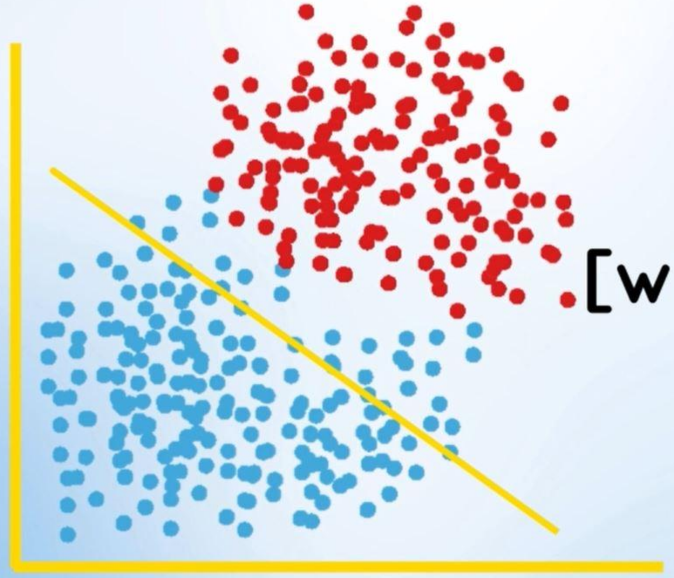
$$= 0.8$$



$$-[\ln(0.57) + \ln(0.60) + \ln(0.95) + \ln(0.70)]$$

$$= 0.37$$

Gradient Descent



$$[w1, w2, b] - \nabla E$$

Gradient Descent

variables

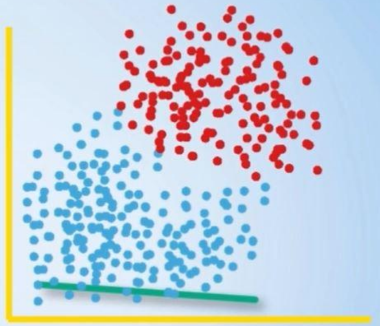
pts = points

p = probability

y = label

m = number of points

$$\nabla E = \frac{pts * (p - y)}{m}$$



Gradient Descent

variables

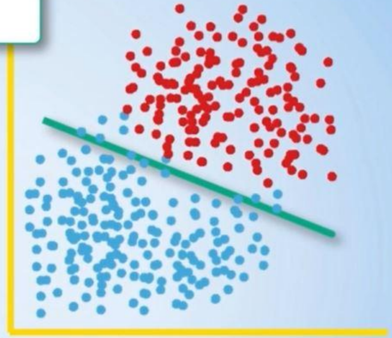
pts = points

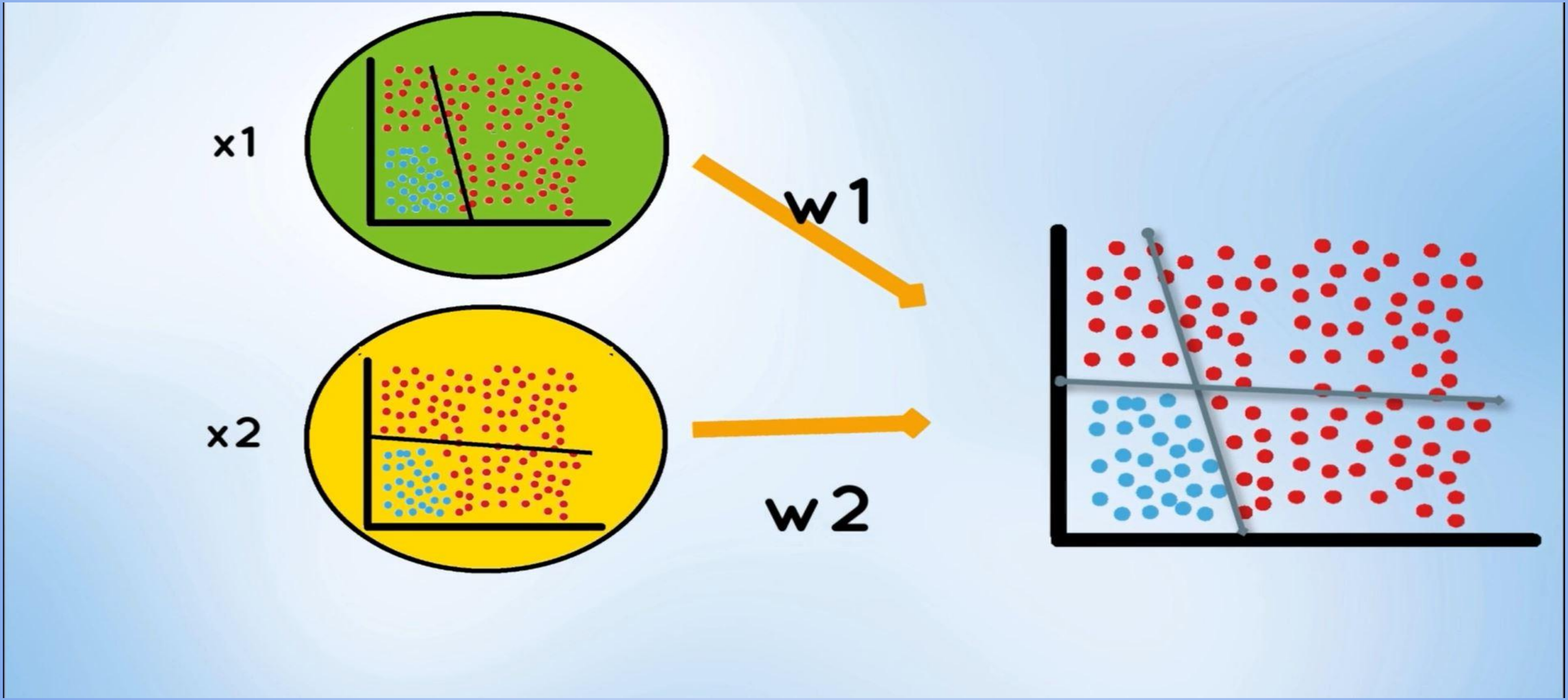
p = probability

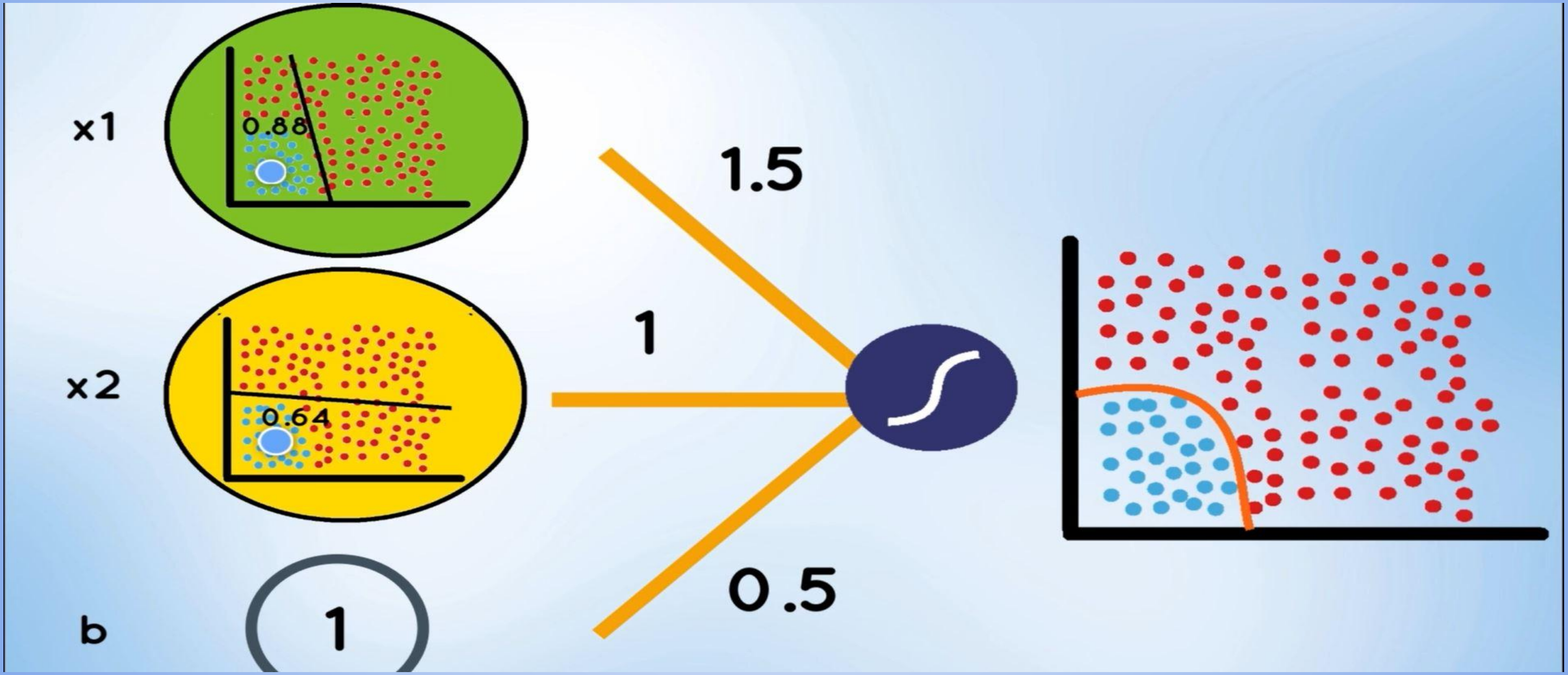
y = label

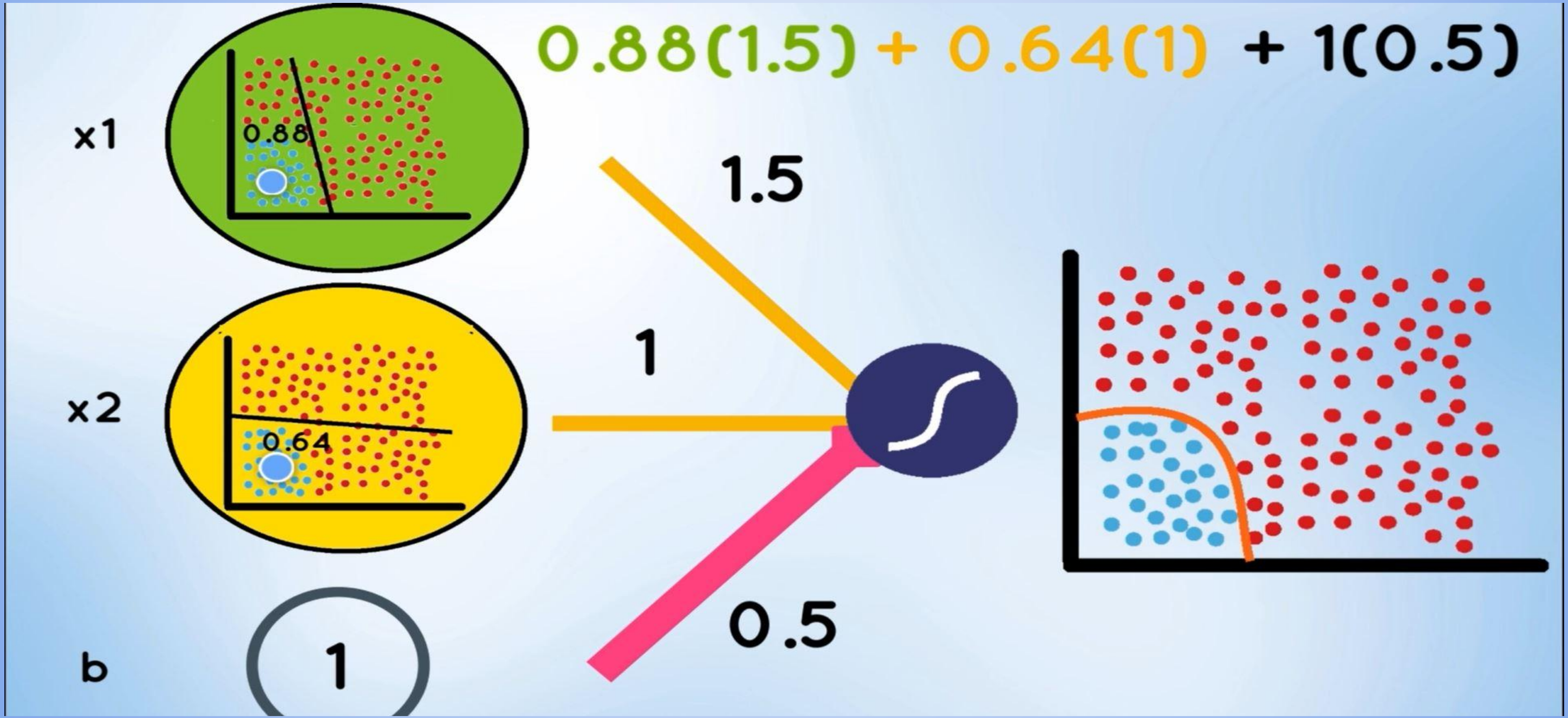
m = number of points

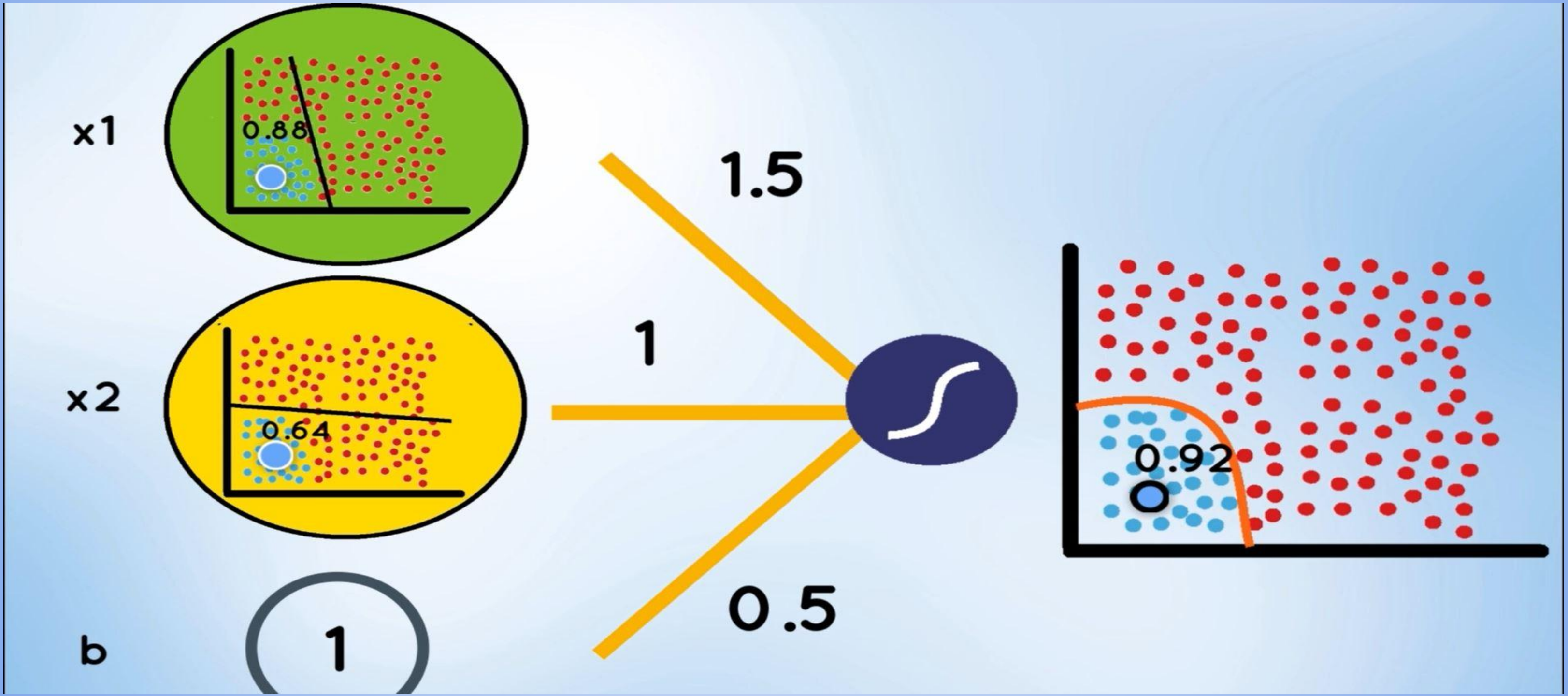
$$\nabla E = \frac{pts * (p - y)}{m} \boxed{*0.01}$$

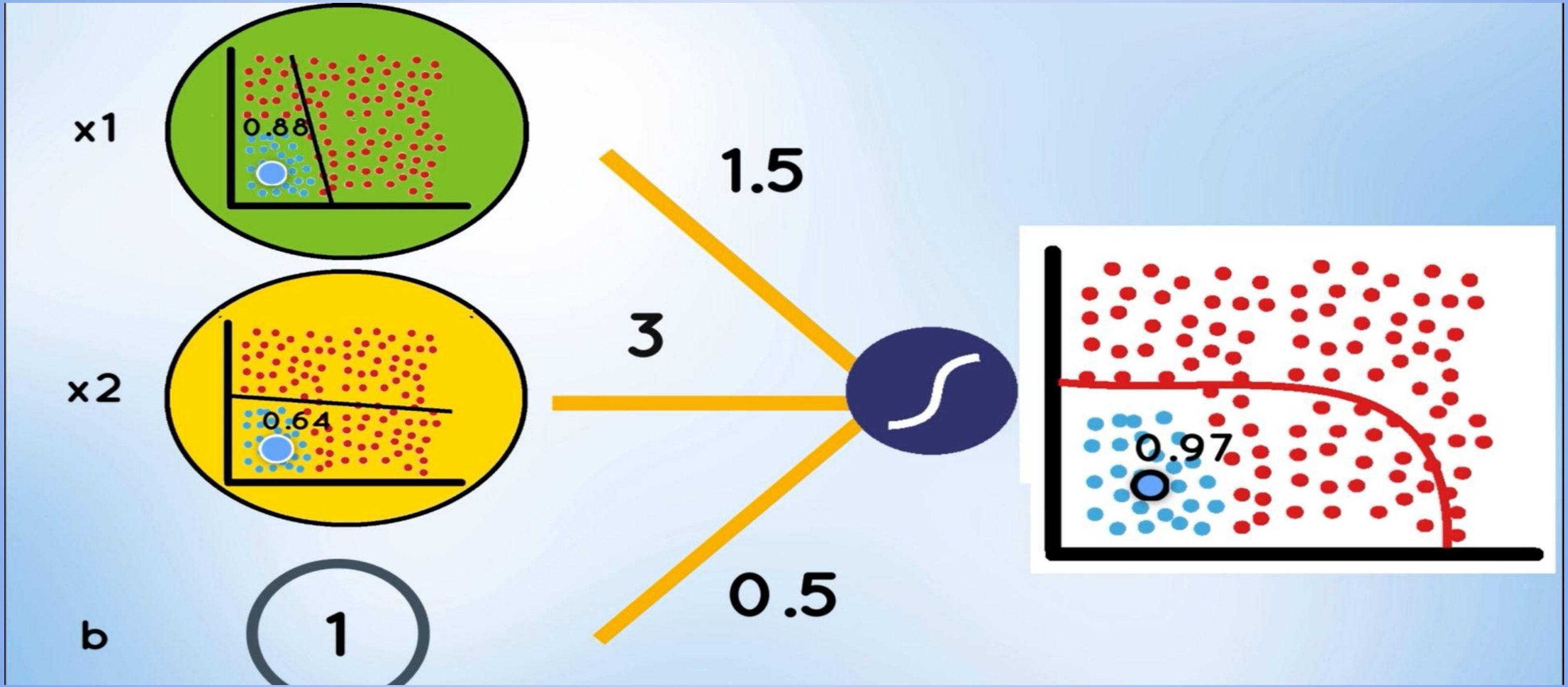


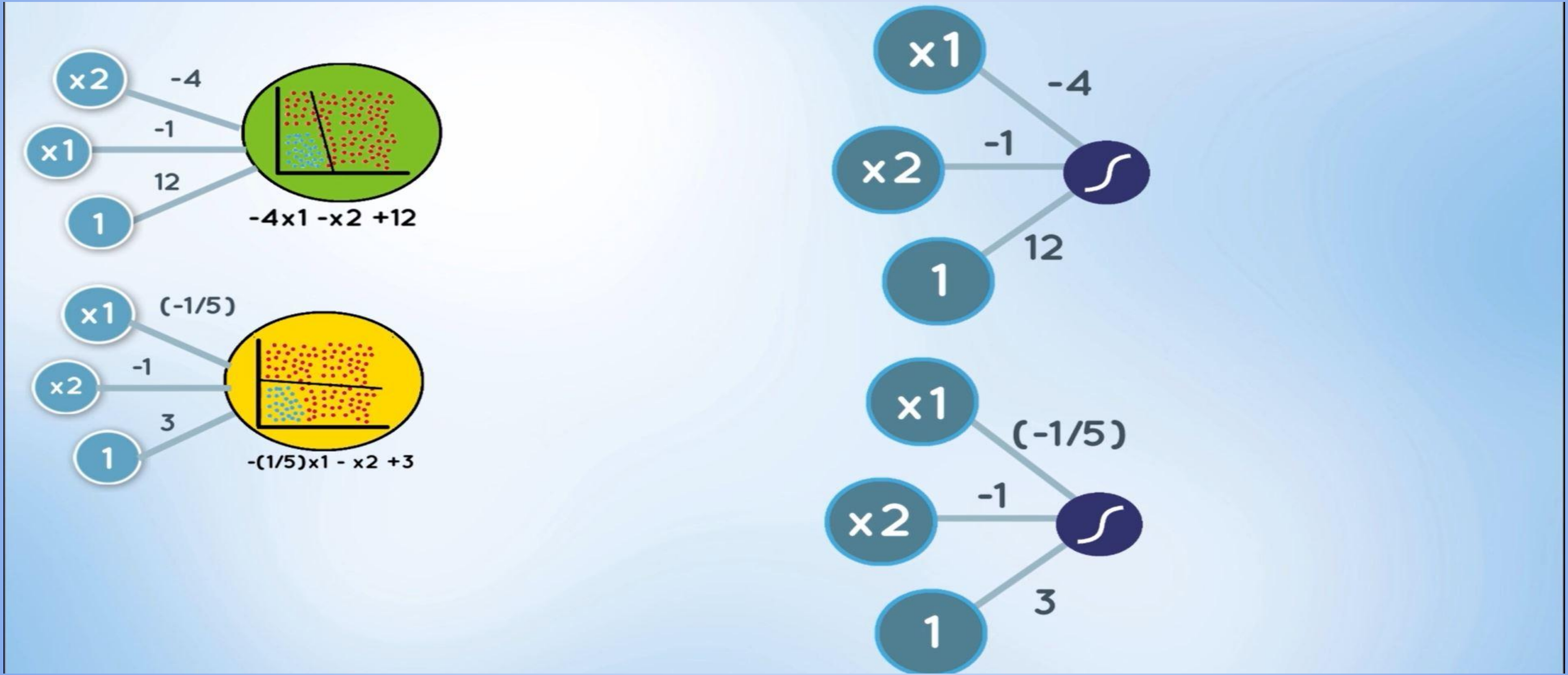

x1
x2
w1
w2

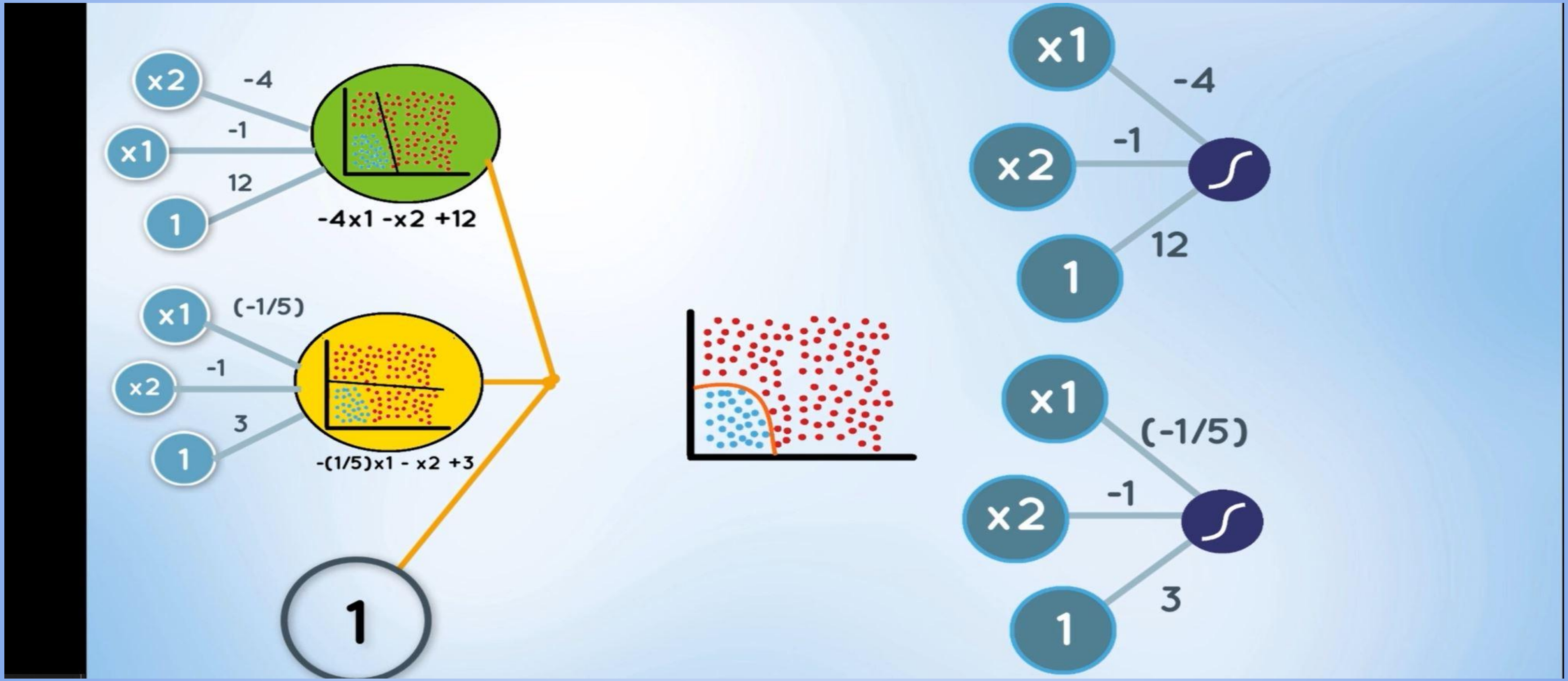

x1
0.88
x2
0.64
b
1
1.5
1
0.5

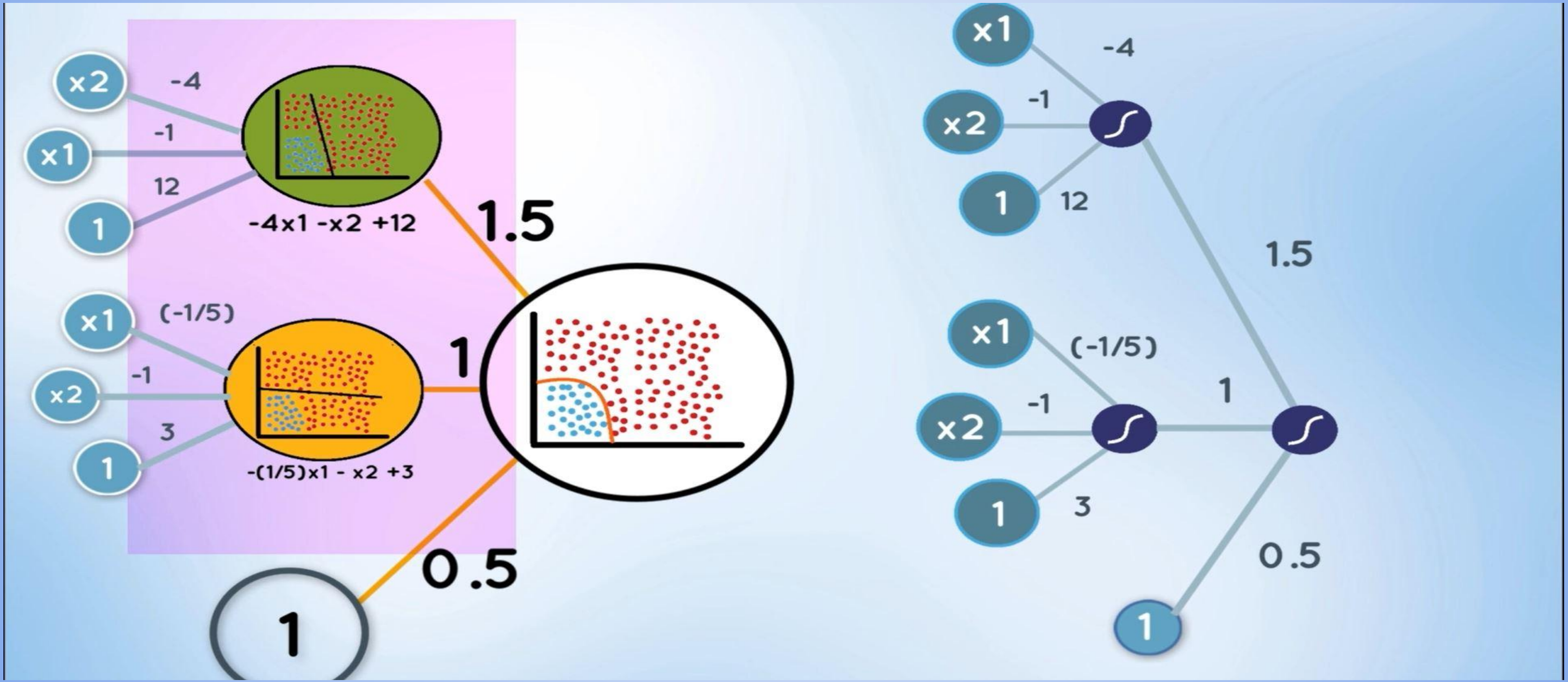

0.88(1.5) + 0.64(1) + 1(0.5)
x1
0.88
1.5
x2
0.64
1
b
1
0.5

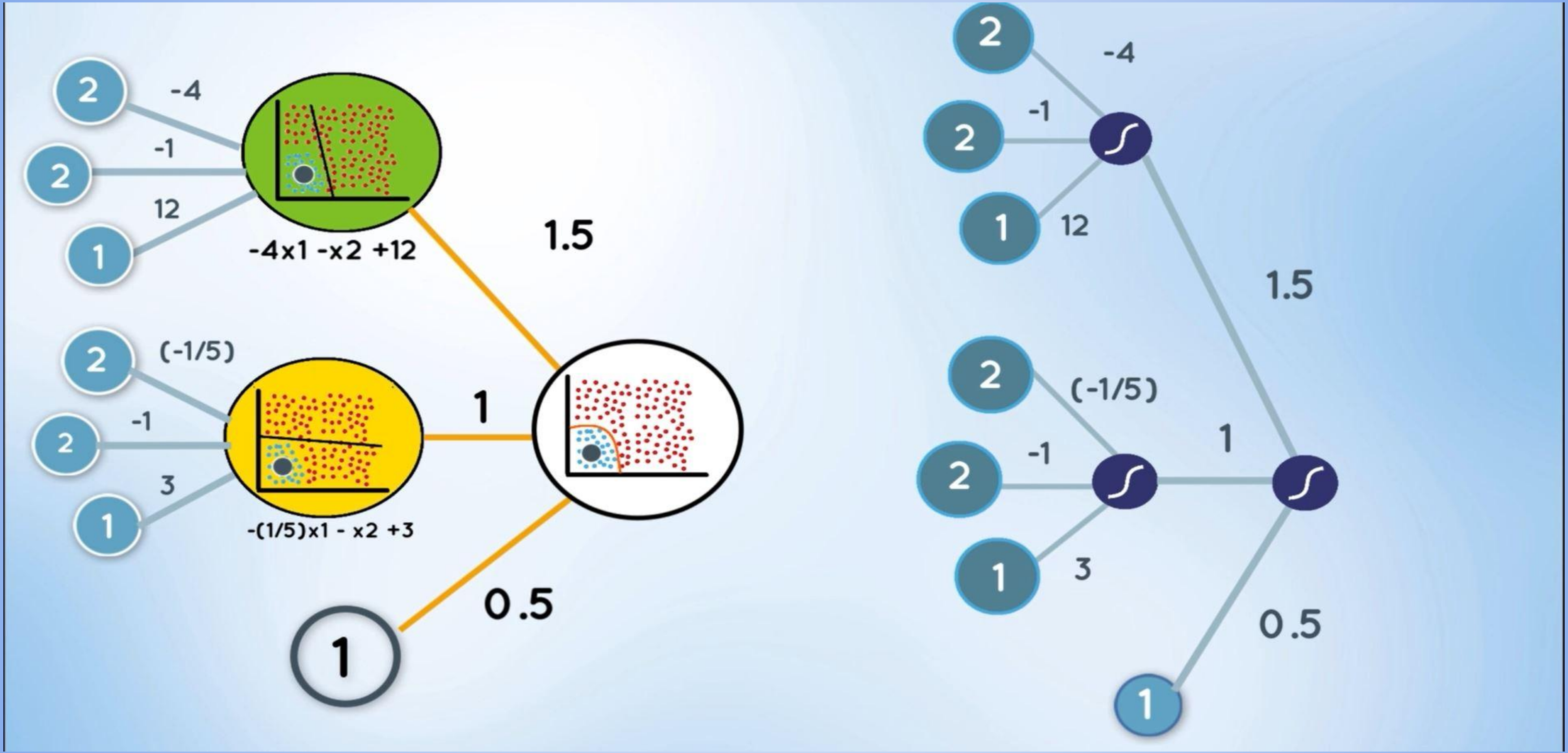

x1
0.88
x2
0.64
b
1
1.5
1
0.5
0.92


x1
0.88
x2
0.64
b
1
1.5
3
0.5
0.97

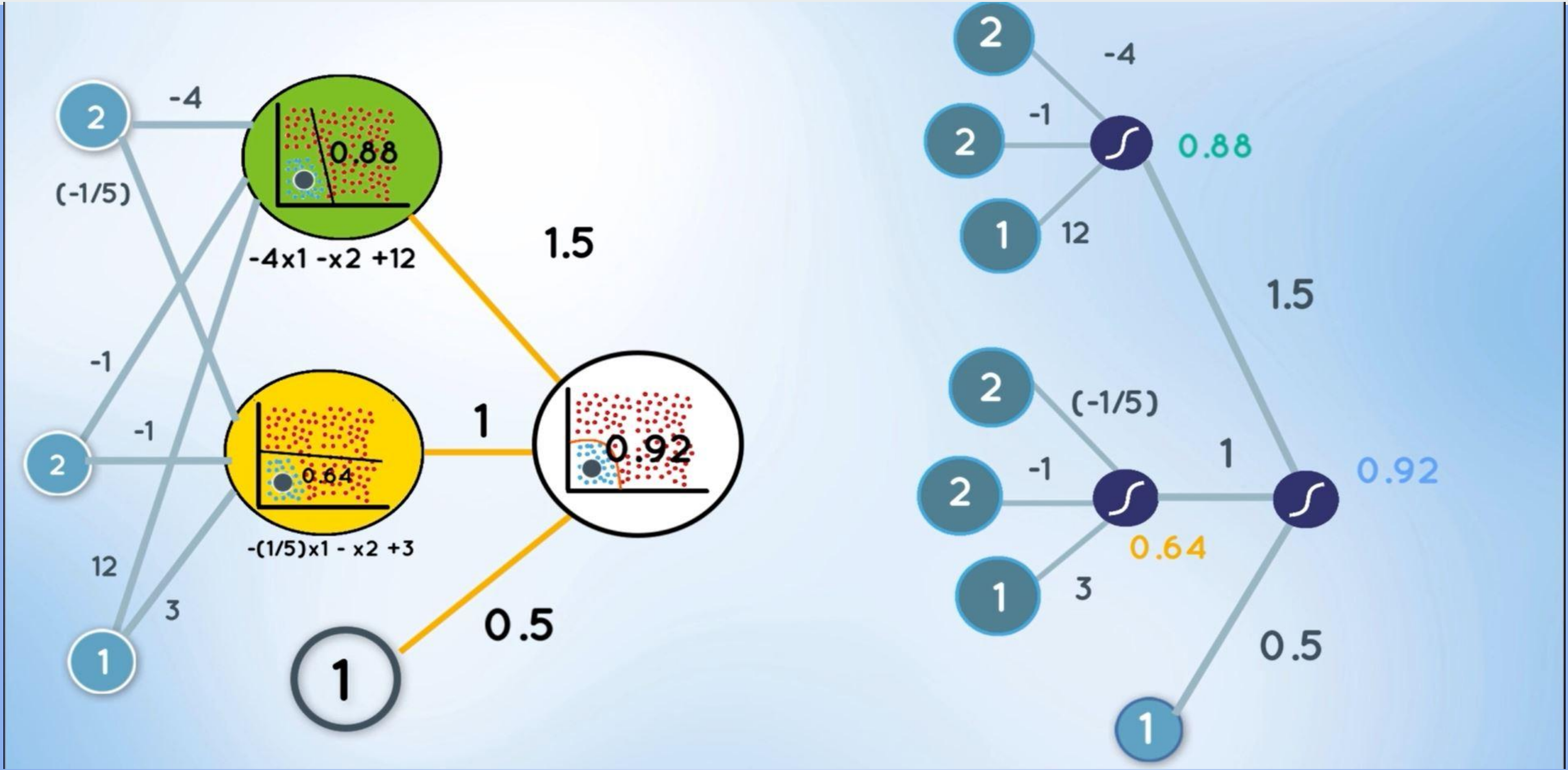

x2
-4
x1
-1
12
1
-4x1 -x2 +12
x1
(-1/5)
x2
-1
3
1
-(1/5)x1 - x2 +3
x1
-4
x2
-1
1
12
x1
(-1/5)
x2
-1
1
3


x2
-4
x1
-1
12
1
-4x1 -x2 +12
x1
(-1/5)
x2
-1
3
1
-(1/5)x1 - x2 +3
1
x1
-4
x2
-1
1
12
x1
(-1/5)
x2
-1
1
3

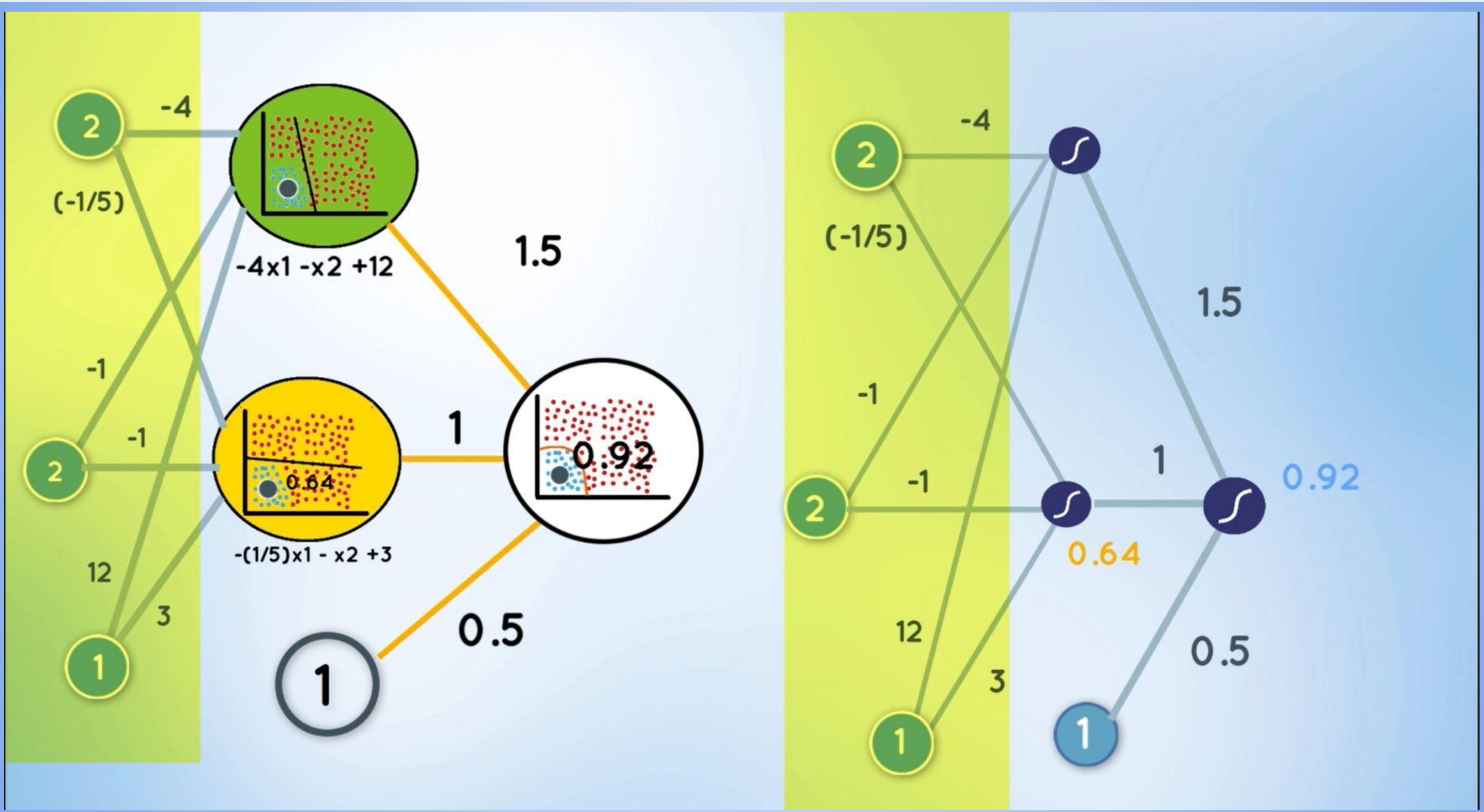

x2
-4
x1
-1
12
1
-4x1 -x2 +12
1.5
x1
(-1/5)
x2
-1
3
1
-(1/5)x1 - x2 +3
1
0.5
1
x1
-4
x2
-1
1
12
1.5
x1
(-1/5)
x2
-1
1
1
3
0.5
1

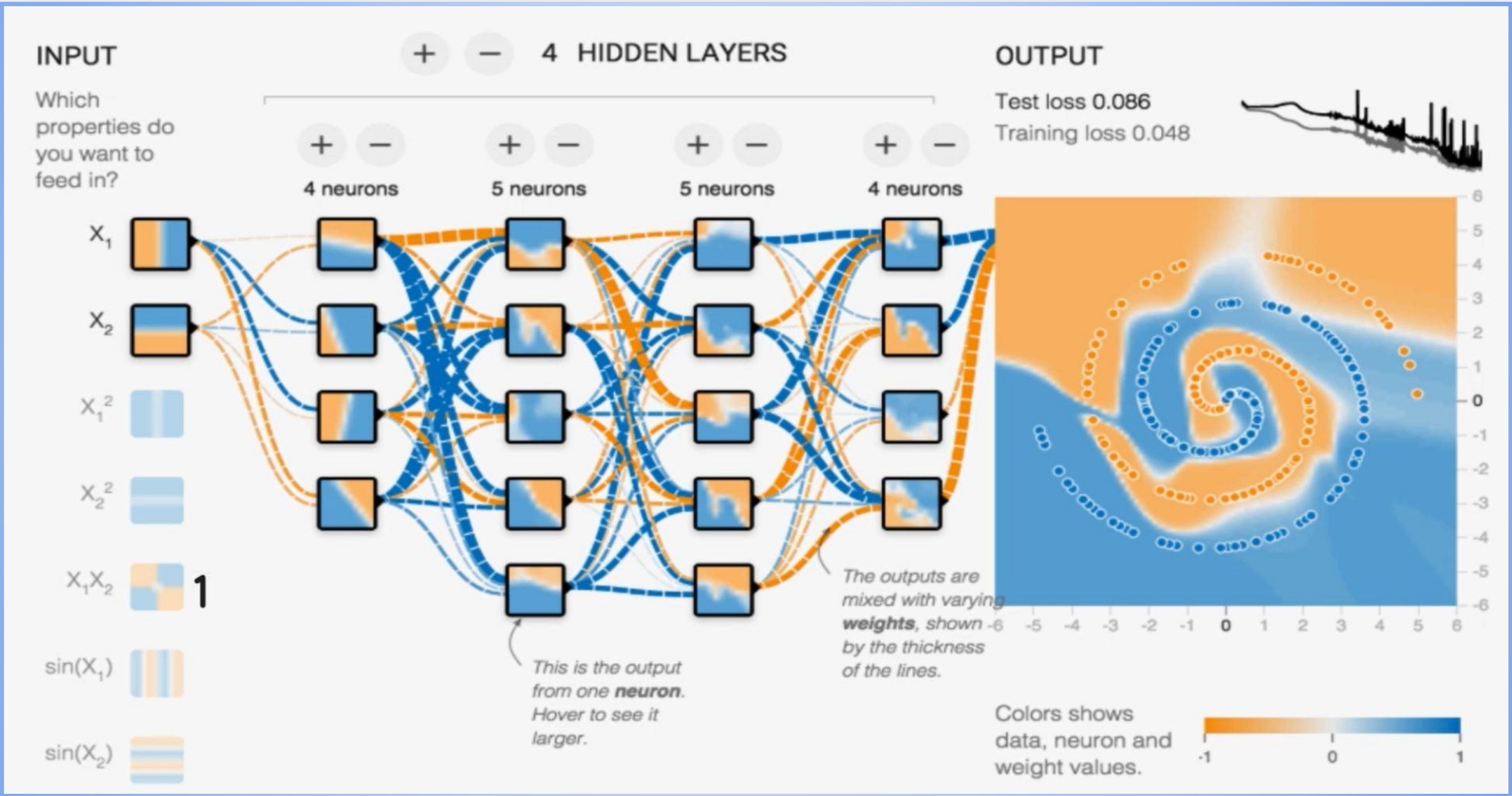


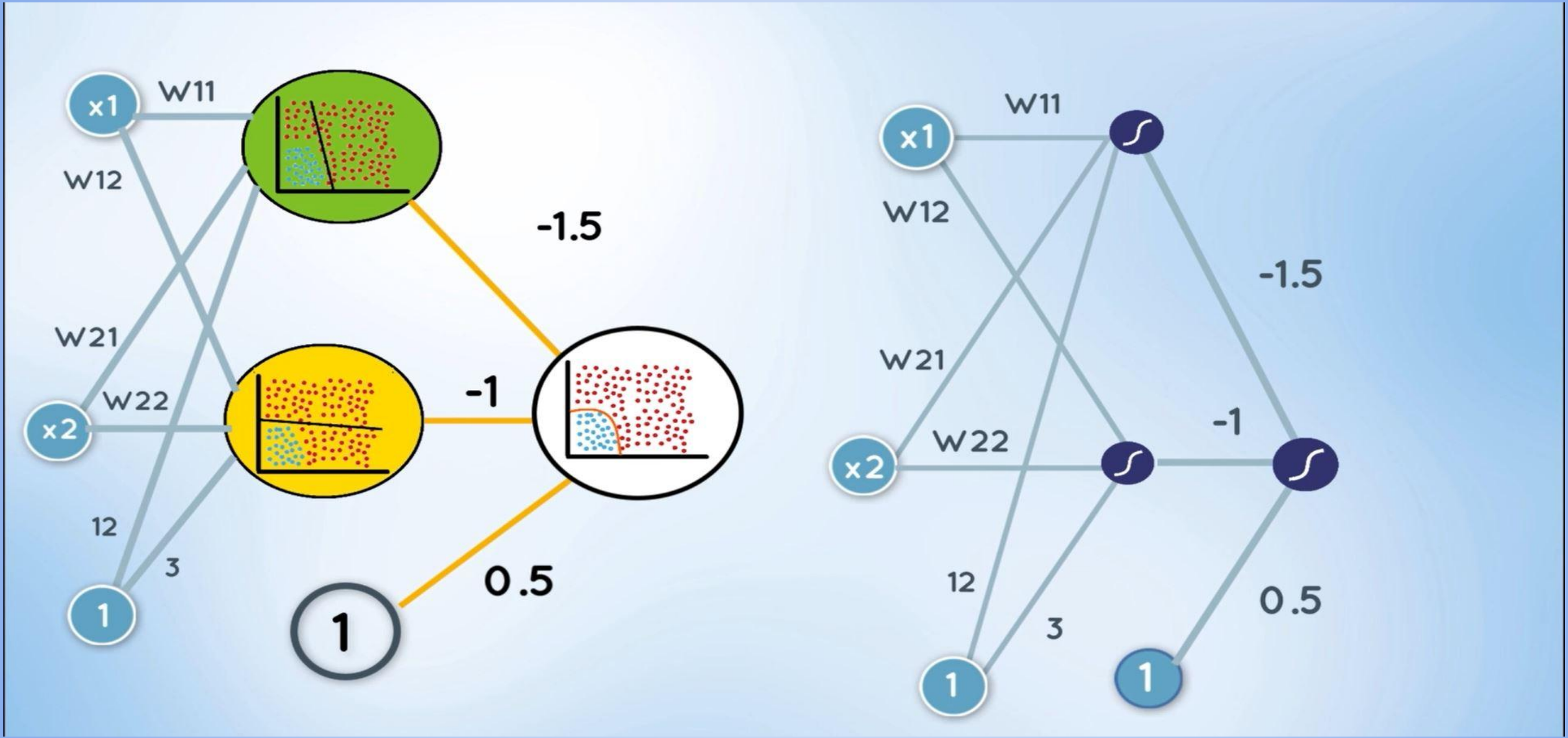
2
-4
2
-1
12
1
-4x1 -x2 +12
1.5
2
(-1/5)
2
-1
1
3
1
-(1/5)x1 - x2 +3
1
0.5
2
-4
2
-1
1
12
1.5
2
(-1/5)
2
-1
1
1
3
0.5
1



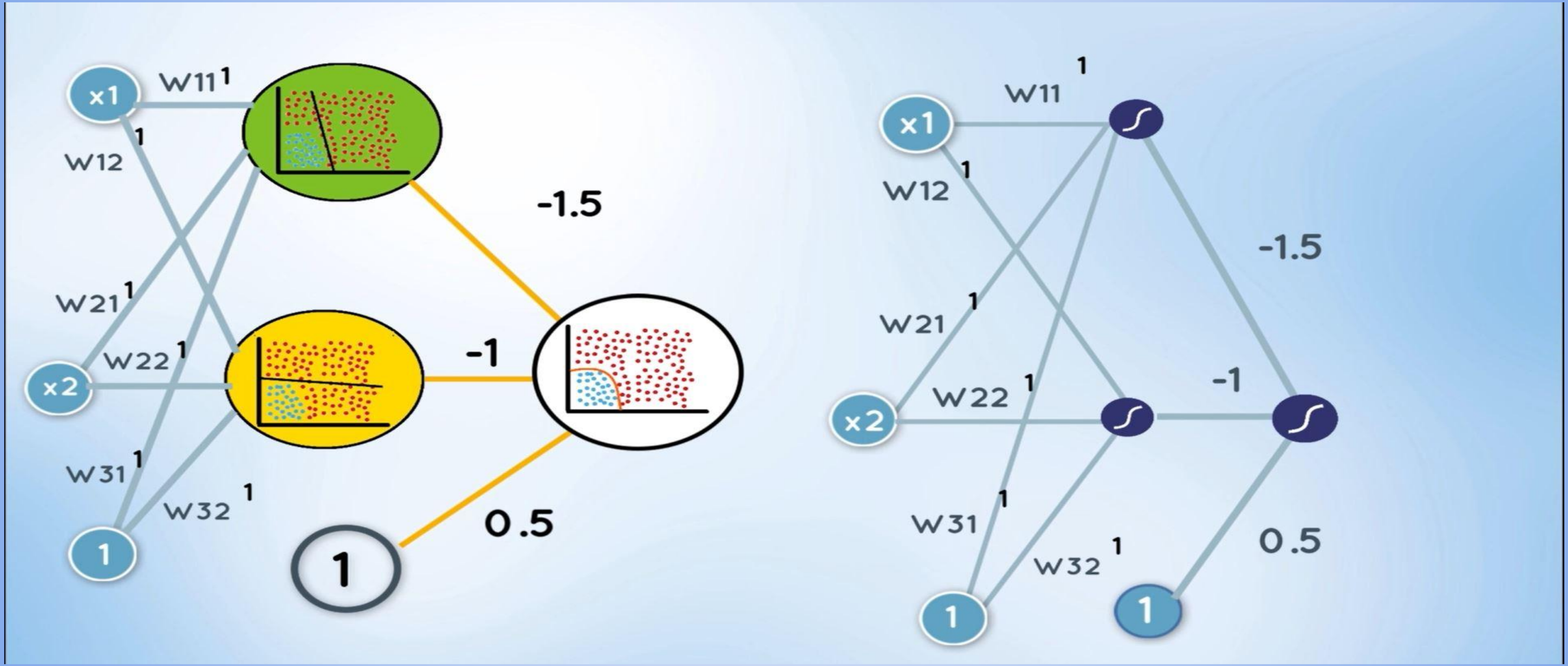
2
-4
(-1/5)
0.88
-4x1 -x2 +12
1.5
-1
2
-1
1
0.64
0.92
-(1/5)x1 - x2 +3
12
3
1
0.5
1
2
-4
2
-1
0.88
1
12
1.5
2
(-1/5)
2
-1
1
0.92
0.64
1
3
0.5
1


-4
(-1/5)
-4x1 -x2 +12
1.5
-1
-1
1
0.92
0.64
-(1/5)x1 - x2 +3
12
3
0.5
1
2
2
1
-4
(-1/5)
1.5
-1
1
-1
0.92
0.64
12
0.5
3
2
2
1
1

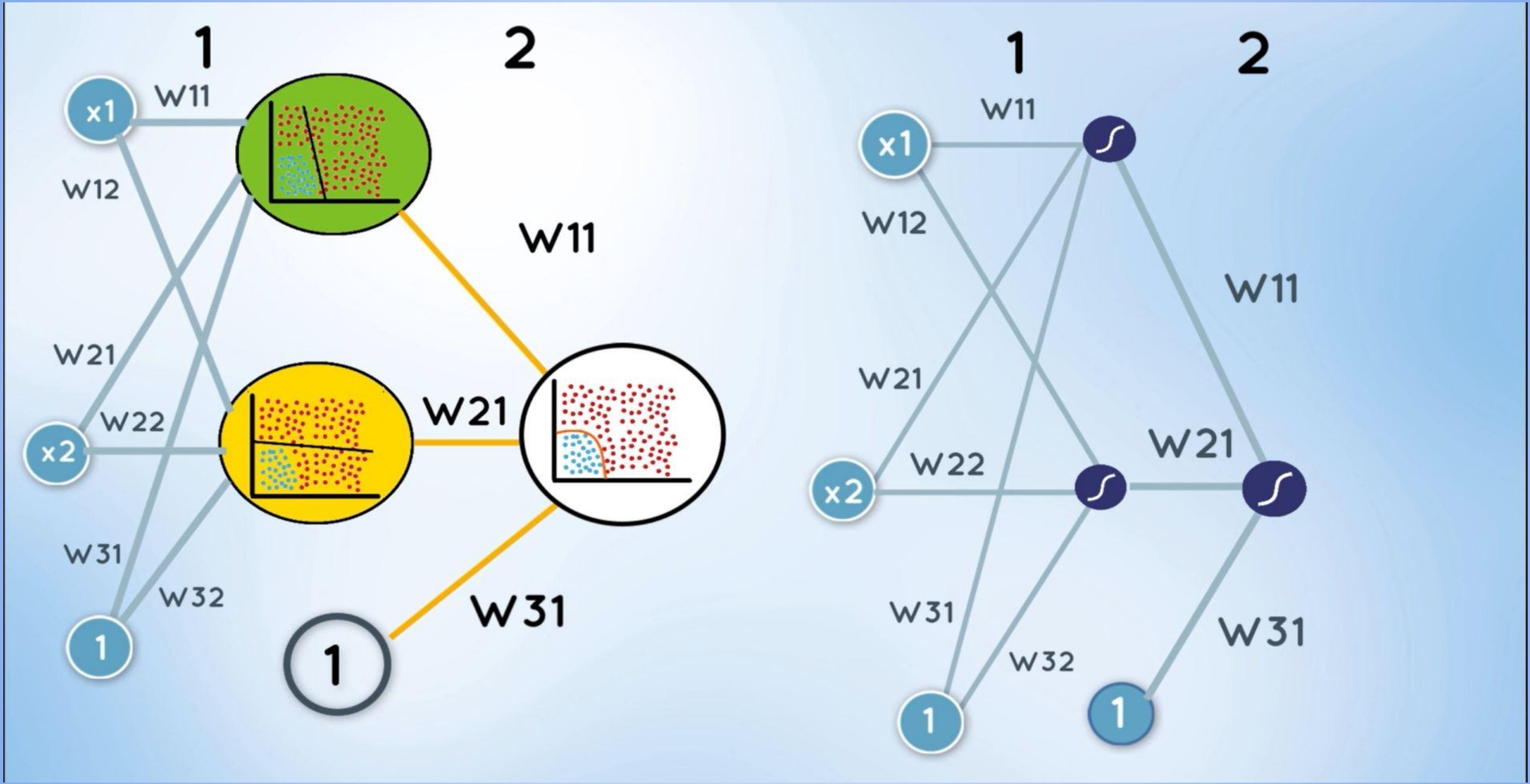

INPUT
Which properties do you want to feed in?
X1
X2
X1^2
X2^2
X1X2
sin(X1)
sin(X2)
4 HIDDEN LAYERS
4 neurons
5 neurons
5 neurons
4 neurons
This is the output from one **neuron**. Hover to see it larger.
The outputs are mixed with varying **weights**, shown by the thickness of the lines.
OUTPUT
Test loss 0.086
Training loss 0.048
Colors shows data, neuron and weight values.
-1
0
1

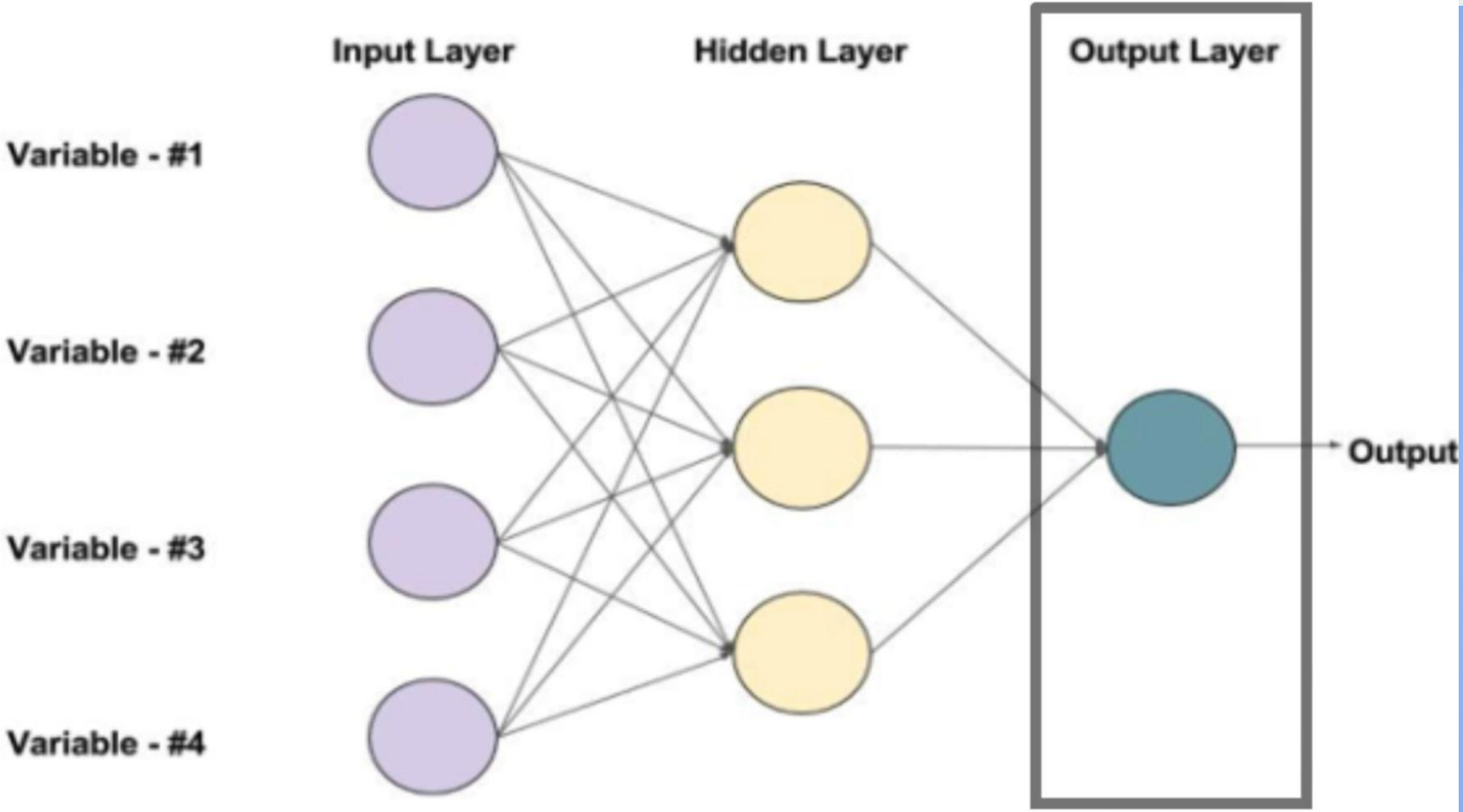


x1
W11
W12
W21
x2
W22
12
3
1
-1.5
-1
0.5
1
x1
W11
W12
W21
x2
W22
12
3
1
-1.5
-1
0.5
1

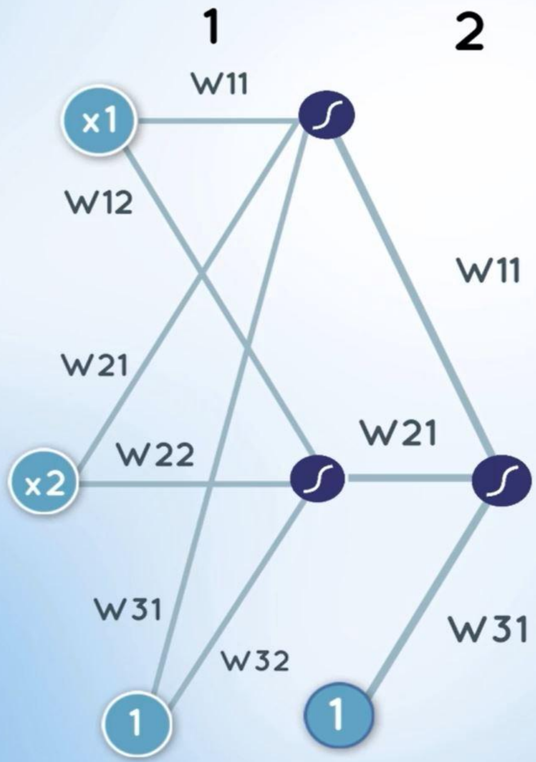


x1
W11 1
W12 1
W21 1
x2
W22 1
W31 1
W32 1
1
-1.5
-1
0.5
1
x1
W11 1
W12 1
W21 1
W22 1
x2
W31 1
W32 1
1
-1.5
-1
0.5
1

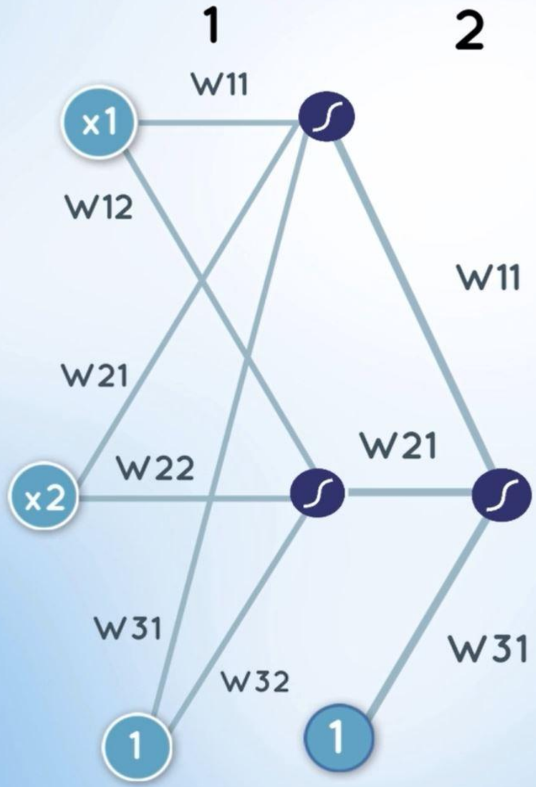

1
2
x1
W11
W12
W21
x2
W22
W31
W32
1
W11
W21
W31
1
1
2
x1
W11
W12
W21
x2
W22
W31
W32
1
W11
W21
W31
1



An example of a Feed-forward Neural Network with one hidden layer (with 3 neurons)



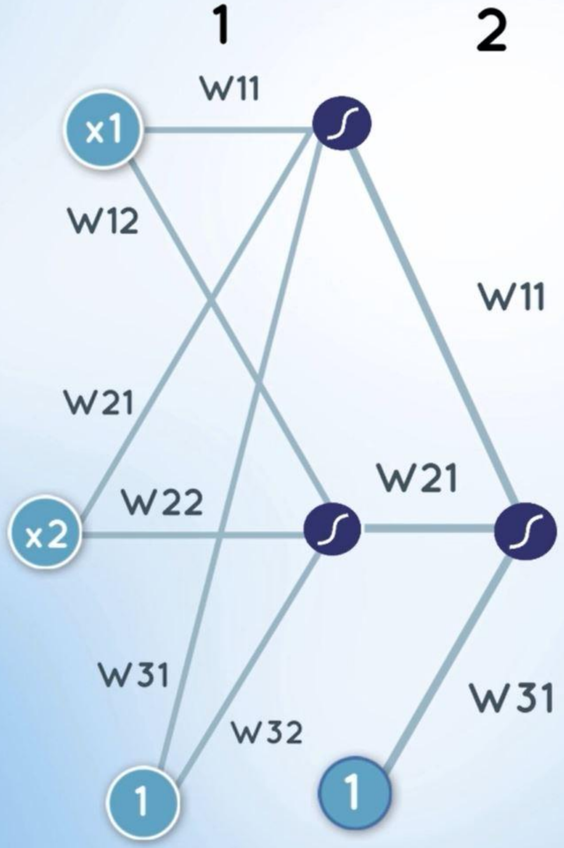
$$[x1\ x2\ 1]\begin{bmatrix} w11 & w12 \\ w21 & w22 \\ w31 & w32 \end{bmatrix}^{1} = [\text{score score}]$$



$$[x1\ x2\ 1]\ \begin{bmatrix} w11 & w12 \\ w21 & w22 \\ w31 & w32 \end{bmatrix}^{1} = [\text{score score}]$$

$$\frac{1}{1+e^{-x}}\ [\text{score score}] = [\text{probability probability}]$$

$$[\text{probability probability } 1]\ \begin{bmatrix} w11 \\ w21 \\ w31 \end{bmatrix}^{2}$$

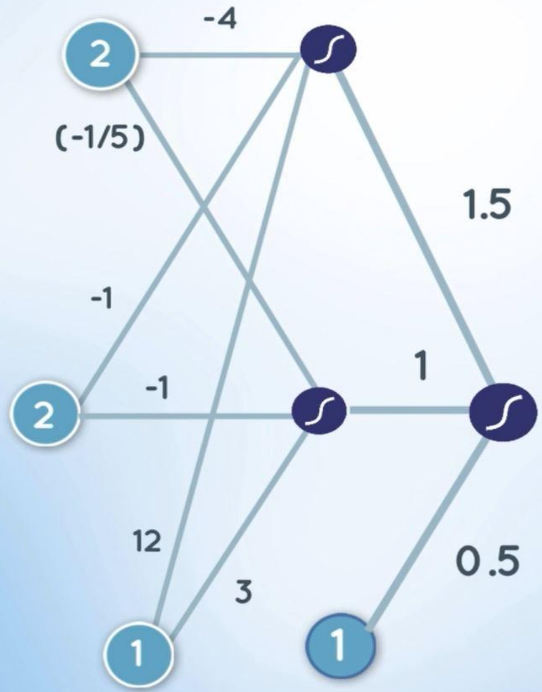


$$[x1\ x2\ 1] \begin{bmatrix} w11 & w12 \\ w21 & w22 \\ w31 & w32 \end{bmatrix}^{1} = [\text{score score}]$$

$$\frac{1}{1+e^{-x}} [\text{score score}] = [\text{probability probability}]$$

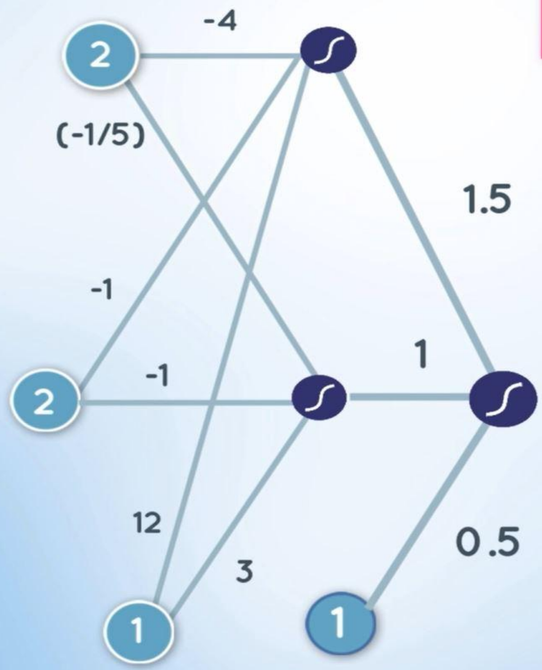
$$[\text{probability probability } 1] \begin{bmatrix} w11 \\ w21 \\ w31 \end{bmatrix}^{2} = [\text{score}]$$

$$\frac{1}{1+e^{-x}} [\text{score}] = [\text{probability}]$$



[2 2 1]

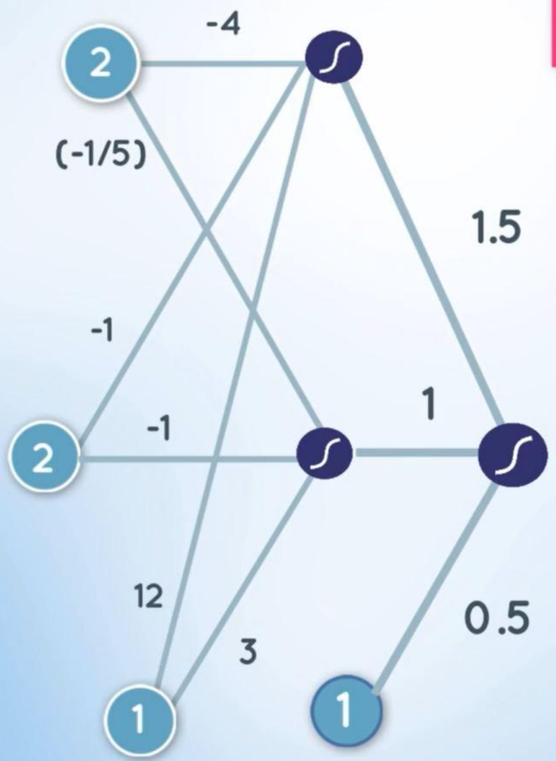
[-4 w12]
[-1 w22]
[12 w32]



[2 2 1]

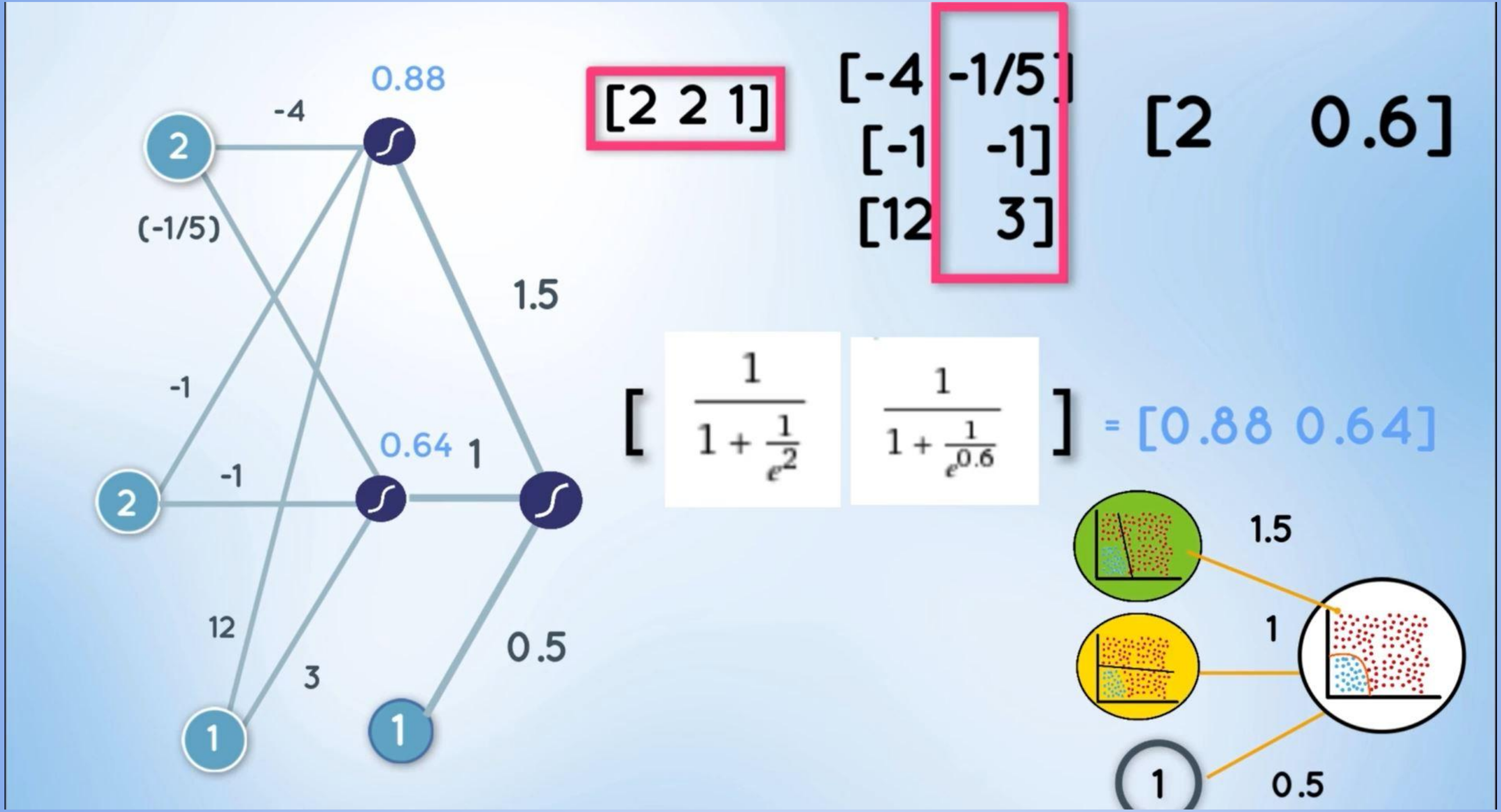
[-4 -1/5]
[-1 -1]
[12 3]

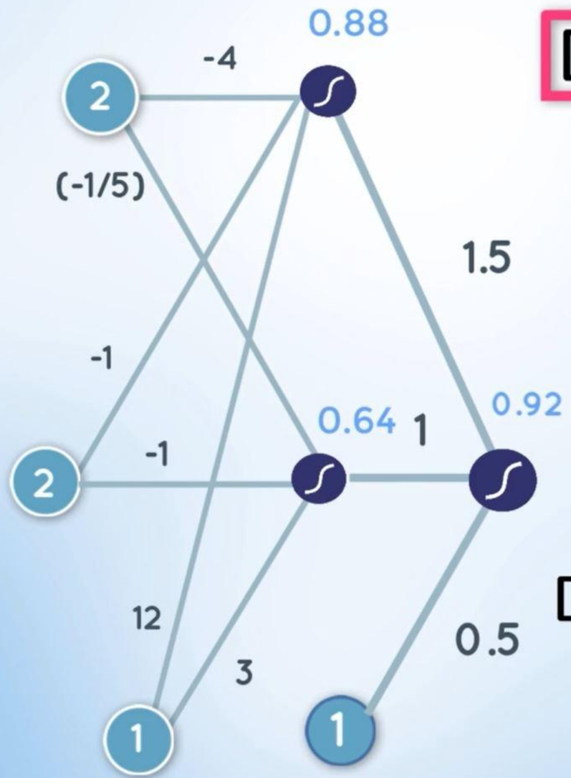
[2 0.6]



$$\boxed{[2\ 2\ 1]} \quad \begin{bmatrix} -4 & -1/5 \\ -1 & -1 \\ 12 & 3 \end{bmatrix} \quad [2 \quad 0.6]$$

$$\left[\frac{1}{1+\frac{1}{e^{2}}} \quad \frac{1}{1+\frac{1}{e^{0.6}}} \right] = [0.88\ 0.64]$$


0.88
-4
2
(-1/5)
1.5
-1
0.64
1
-1
2
12
3
0.5
1
1
[2 2 1]
[-4 -1/5]
[-1 -1]
[12 3]
[2 0.6]
= [0.88 0.64]
1.5
1
1
0.5

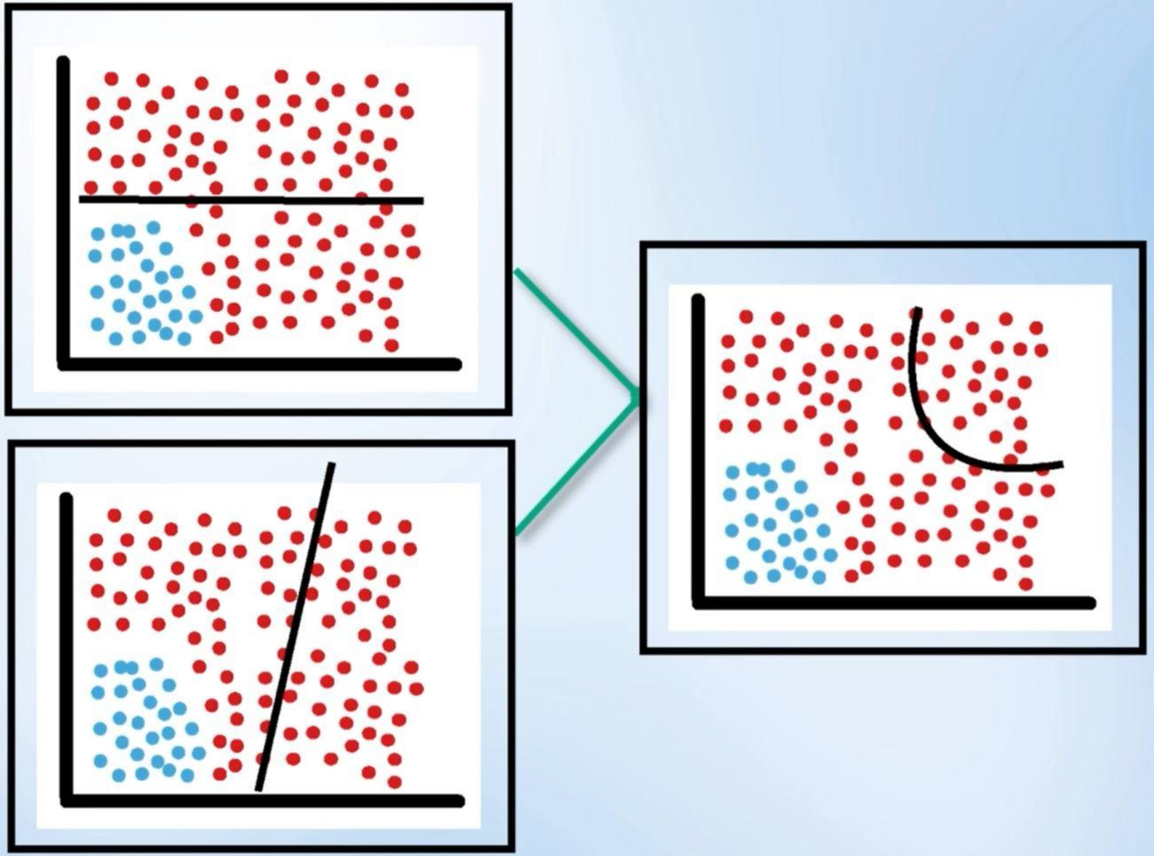


$$\boxed{[2\ 2\ 1]} \quad \begin{bmatrix} -4 & -1/5 \\ -1 & -1 \\ 12 & 3 \end{bmatrix} \quad [2 \quad 0.6]$$

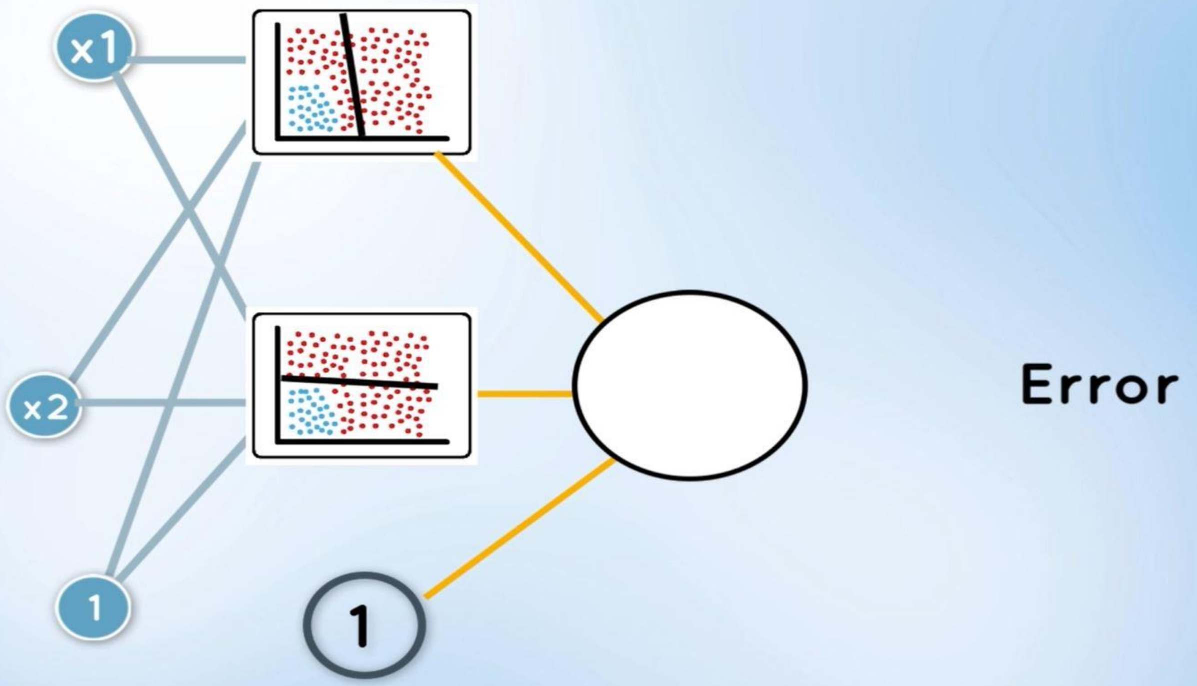
$$\left[\frac{1}{1+\frac{1}{e^{2}}} \quad \frac{1}{1+\frac{1}{e^{0.6}}} \right] = [0.88\ 0.64]$$

$$[0.88\ 0.64\ 1] \quad \begin{bmatrix} 1.5 \\ 1 \\ 0.5 \end{bmatrix} \quad [2.46]$$

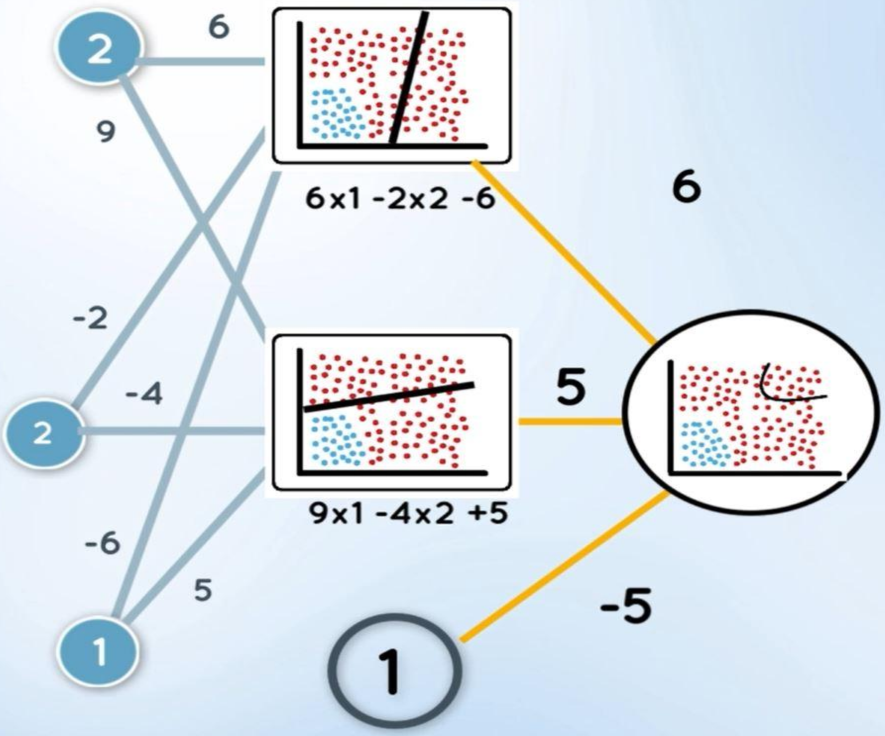
$$\frac{1}{1+\frac{1}{e^{2.46}}} = [0.92]$$



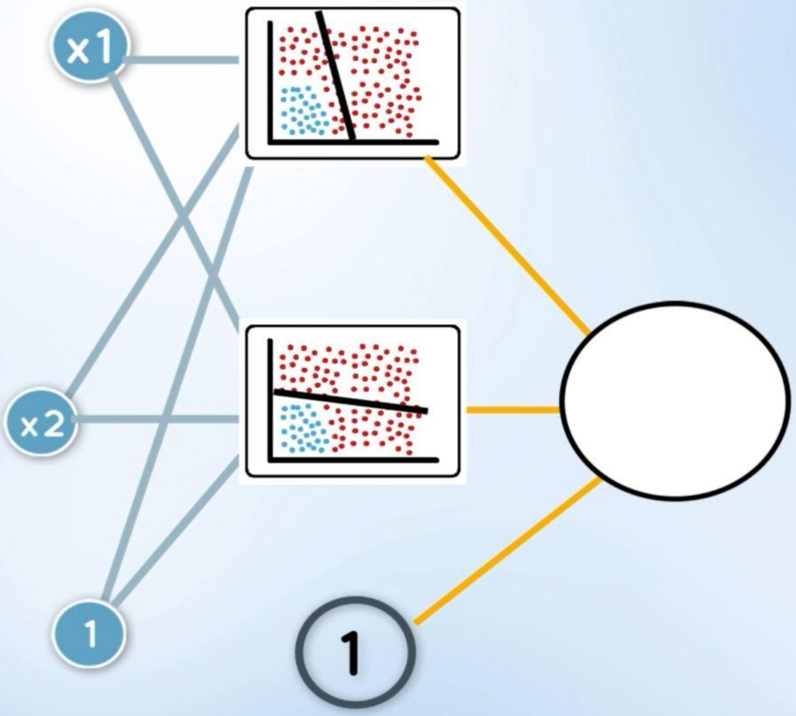
Error function



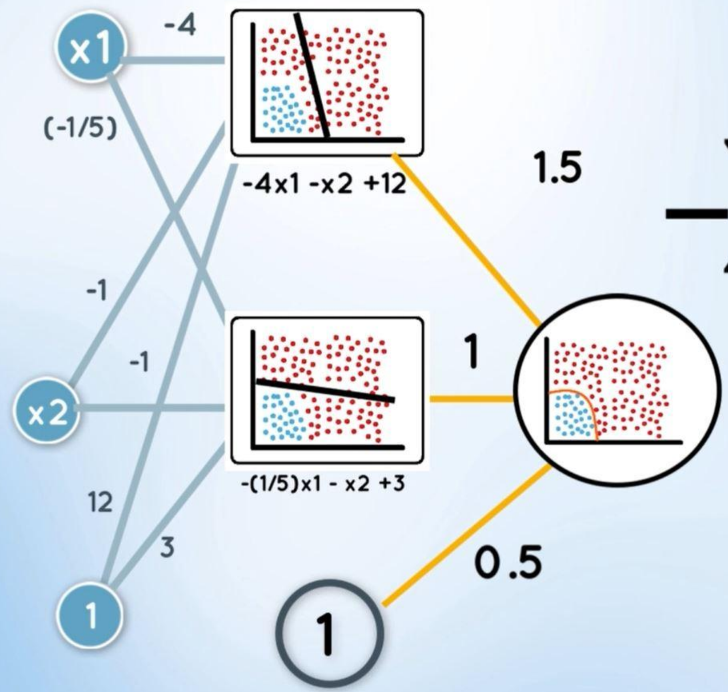
Error function



Error function

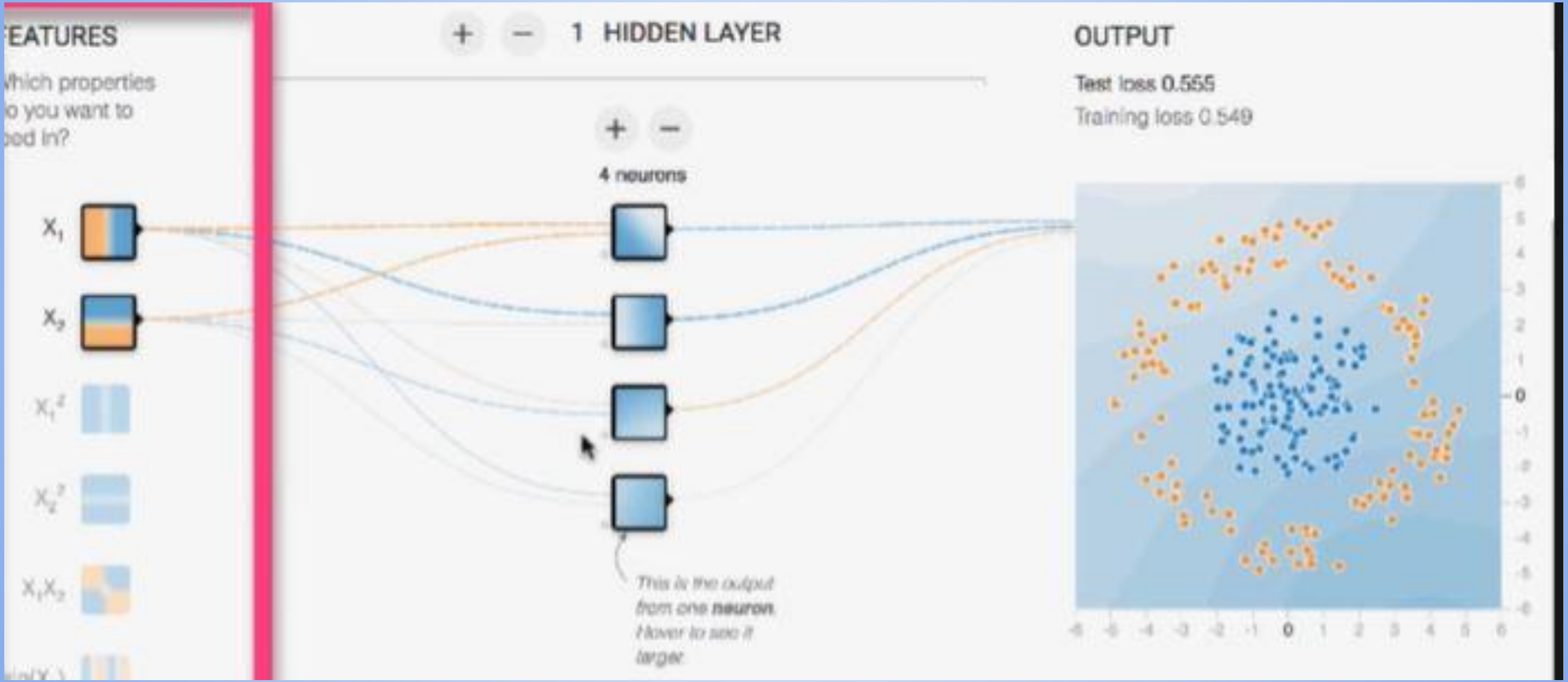


Cross Entropy

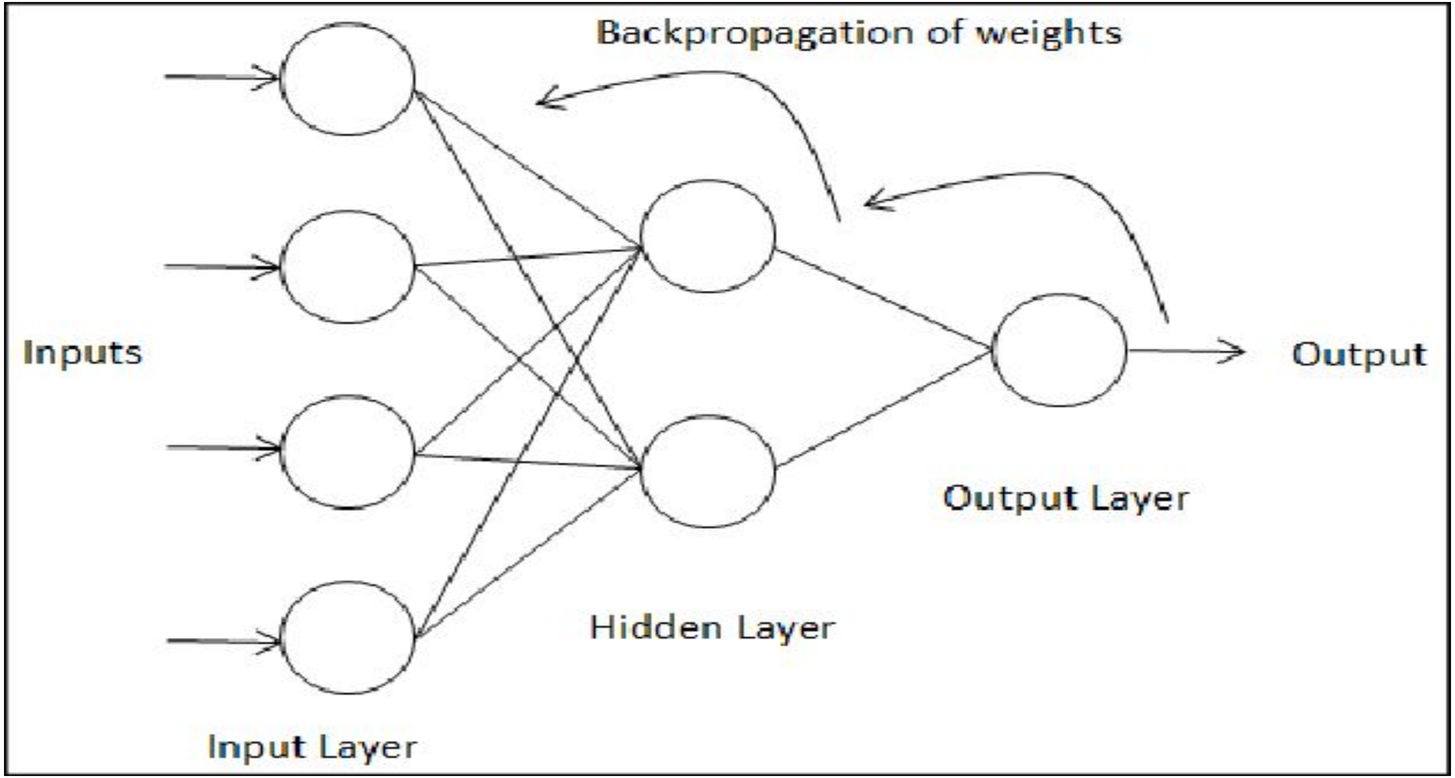


$$-\sum y\ln(p) + (1-y)(\ln(1-p))$$

1. Feedword to predict all outputs
2. Determine Total Error with Cross_Entropy
3. Backpropagation
4. Repeat at some learning rate



FEATURES
Which properties do you want to feed in?
X1
X2
1 HIDDEN LAYER
4 neurons
This is the output from one neuron. Hover to see it larger.
OUTPUT
Test loss 0.555
Training loss 0.549


Backpropagation of weights
Inputs
Output
Output Layer
Hidden Layer
Input Layer