①

This color is a comment

1. logical OR

$z_{in} = (0\,1\,1\,0\,1)$ for example

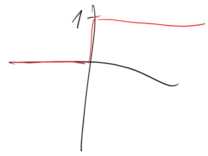
$\beta = (1\,1\,1\,1\,1) = \{1\}^D$

$b = -0{,}5$ (to avoid exactly 0)

activation function ρ = step function Θ

e.g. $z' = z_{in} \cdot \beta + b = (0+1+1+0+1) - 0{,}5 = 2.5$

$\rho(z') = 1$



2. masked logical OR

$z_{in} = (1\,0\,1)$ $\beta = c$

$c = (1\,0\,1)$ $b = -0{,}5$

activation function $\rho = \Theta$

$z' = z_{in}\beta + b = (1+0+1) - 0{,}5 = 1{,}5$ $\Theta(1.5) = 1$

3. perfect match

$z_{in} = (1\,1\,0)$ for example $\beta_i = \begin{cases} -1 & \text{if } c_i = 0 \\ +1 & \text{if } c_i = 1 \end{cases} = (1\,1\,-1)$

$c = (1\,1\,0)$ for example $b = -\sum_{i=1}^{D} c_i + 1$

activation function ρ = ReLu

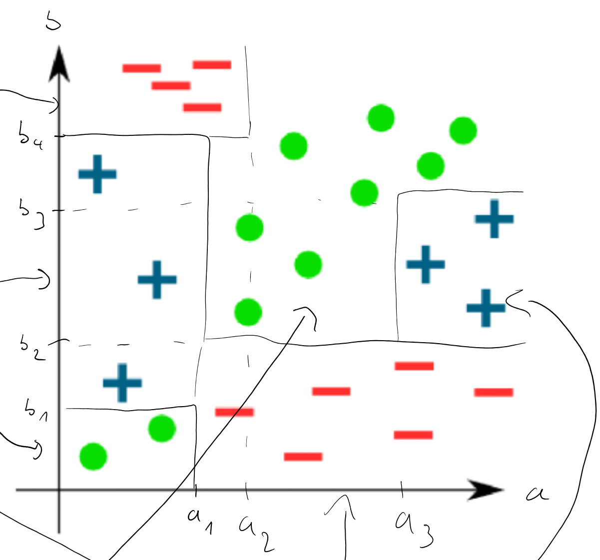
$z' = z_{in}\beta + b = (1+1+0) - 1 = 1$ $\rho(1) = 1$

I think this solution should work as well.

$X = (a_i, b_i)$ $M = 6$

first layer $f(x) \to \{0,1\}^6$

$$f(a,b) = \begin{cases} (100\,000) & \text{if } a < a_1 \text{ and } b < b_1 \\ (010\,000) & \text{if } a > a_1 \text{ and } b < b_2 \\ (001\,000) & \text{if } a > a_1 \text{ and } b > b_1 \text{ and } b < b_4 \\ (000\,100) & \text{if } a < a_2 \text{ and } b > b_4 \\ (000\,010) & \text{if } a > a_3 \text{ and } b > b_2 \text{ and } b < b_3 \\ (000\,001) & \text{else} \end{cases}$$



M=3 is enough since a Hypercube with M=3 has 8 corners. Theoretically $f(x) \to \{0,1\}^8$, however appending 00 to each vector is redundant

Second layer not necessary.

third layer has 3 Neurons

1. Z_{out_1} = masked OR (Z_{in}, 010100) red minus
2. Z_{out_2} = masked OR (Z_{in}, 001010) blue plus
3. Z_{out_3} = masked OR (Z_{in}, 100001) green circle

Given the input vectors above this should work

---

other option $M = 7$ 7 neurons: one for $a > a_1$, one for $a > a_2$ ...

$\rho_1(a,b) = \Theta(a - a_1)$ so $\beta = (1,0)$: $Z_{in} = X = (a,b)$ $Z' = Z_{in}\beta + b = a - a_1$

$b = -a_1$ $\rho = \Theta$

$\rho_2(a,b) = \Theta(a - a_2)$

$\rho_3(a,b) = \Theta(a - a_3)$

$\rho_4(a,b) = \Theta(b - b_1)$

$\rho_5(a,b) = \Theta(b - b_2)$

$\rho_6(a,b) = \Theta(b - b_3)$

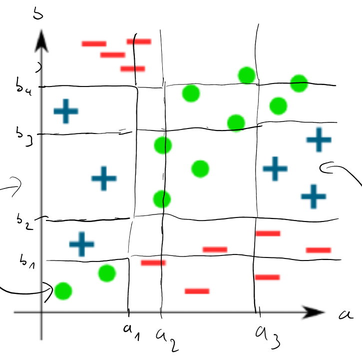
$\rho_7(a,b) = \Theta(b - b_4)$

examples:

this would be (0000000)

this would be (1111100)

this would be (0001100)



Second layer: as many (perfect match) Neurons as regions ($4 \cdot 5 = 20$ here)

each neuron represents one Region

1, Z_{out} = perfect match (Z_{in}, (0000000))

2. ...

3. ...

third layer: 3 masked OR Neurons:

Z_{out_1} = masked OR (Z_{in}, $\{0,1\}^{20}$) with 1 where region contains red minus

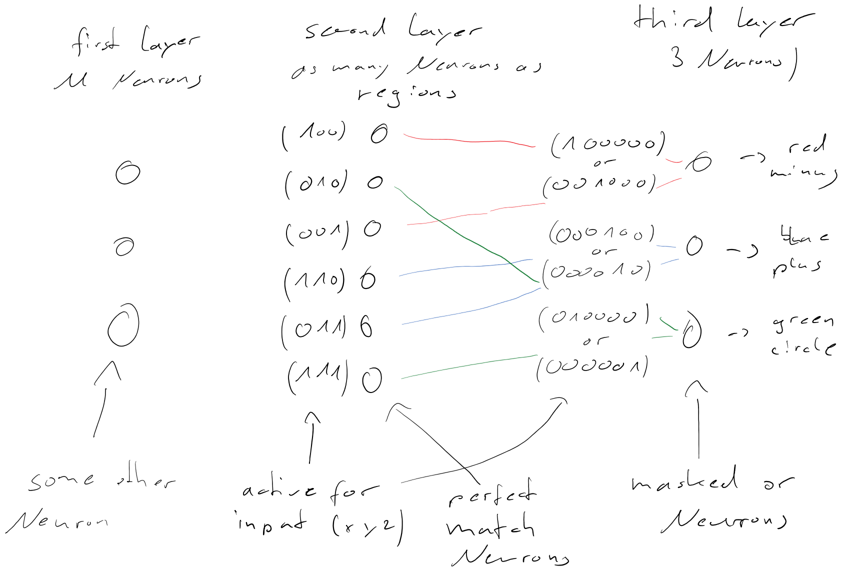
Z_{out_2} = masked OR (Z_{in}, $\{0,1\}^{20}$) with 1 where region contains blue plus

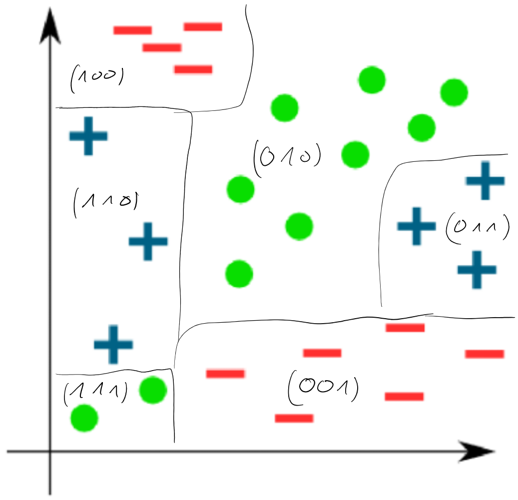
Z_{out_3} = masked OR (Z_{in}, $\{0,1\}^{20}$) with 1 where region contains green circle

---

third option

$M = 3 = \text{ceil}(\log_2(\#\text{Regions}))$





For more dimensions you need more decision Regions and M gets larger. You always need M Neurons in the first layer mapping Features to distinct M-dimensional vectors. The second layer has as many Neurons as decision Regions and each M-dim vector gets mapped to a specific Neuron representing that region. The last layer has as many Neurons as there are classes. Using the masked OR all Neurons corresponding to a class are mapped to one Neuron representing that class.

This gets difficult, because there will be a lot of decision Regions and a lot of necessary Neurons. Also overfitting will be a problem.

2 a network with L layers

$$
\begin{aligned}
Z_0 &= X \\
\tilde{Z}_l &= Z_{l-1} \cdot B_l + b_l \\
Z_l &= \phi_l(\tilde{Z}_l)
\end{aligned}
$$

$$Z_L = Z_{L-1} \cdot B_L + b_L$$

$$= (Z_{L-2} \cdot B_{L-1} + b_{L-1}) \cdot B_L + b_L$$

...

$$= Z_{L-2} \cdot \underbrace{B_{L-1} \cdot B_L}_{\text{new } B} + \underbrace{b_{L-1} \cdot B_L + b_L}_{\text{new } b}$$

$$= Z_{L-2} \, B + b$$

... repeat for all L layers

$$= Z_0 \, B' + b'$$

same as one layer