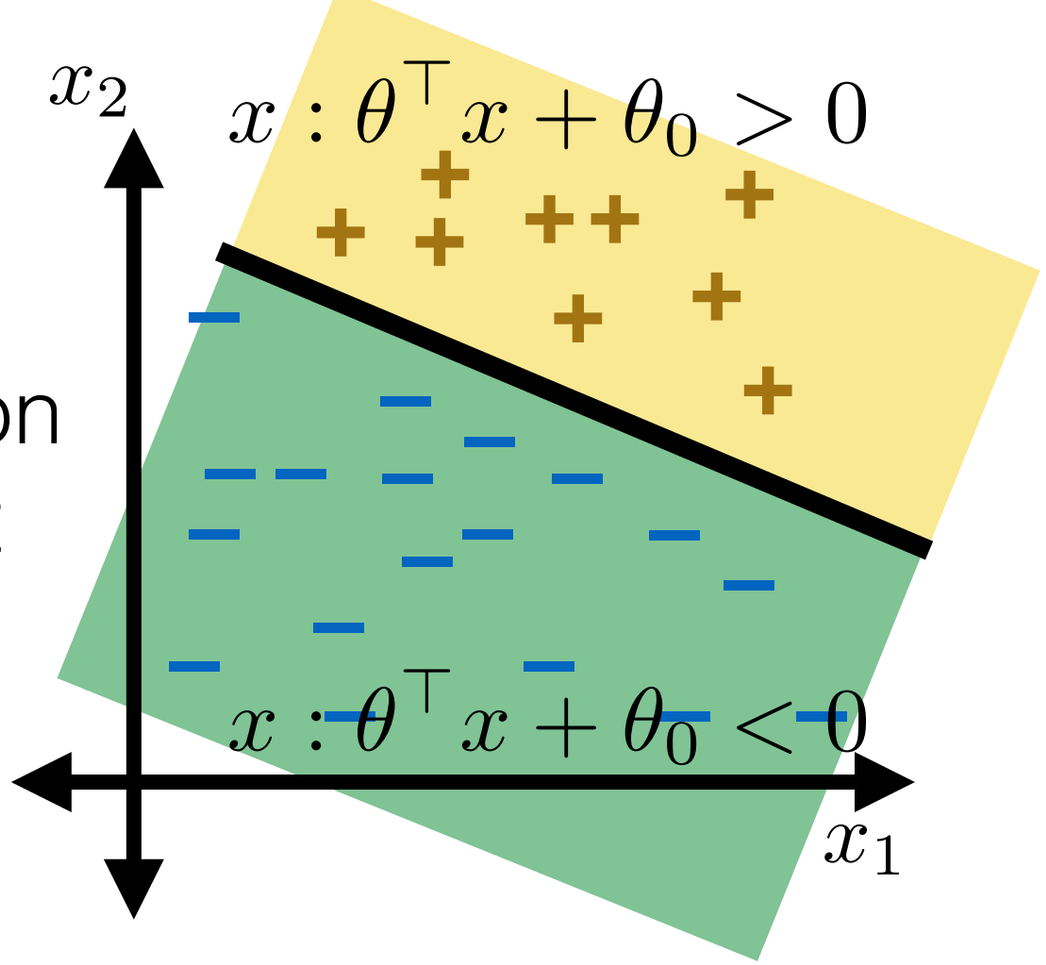
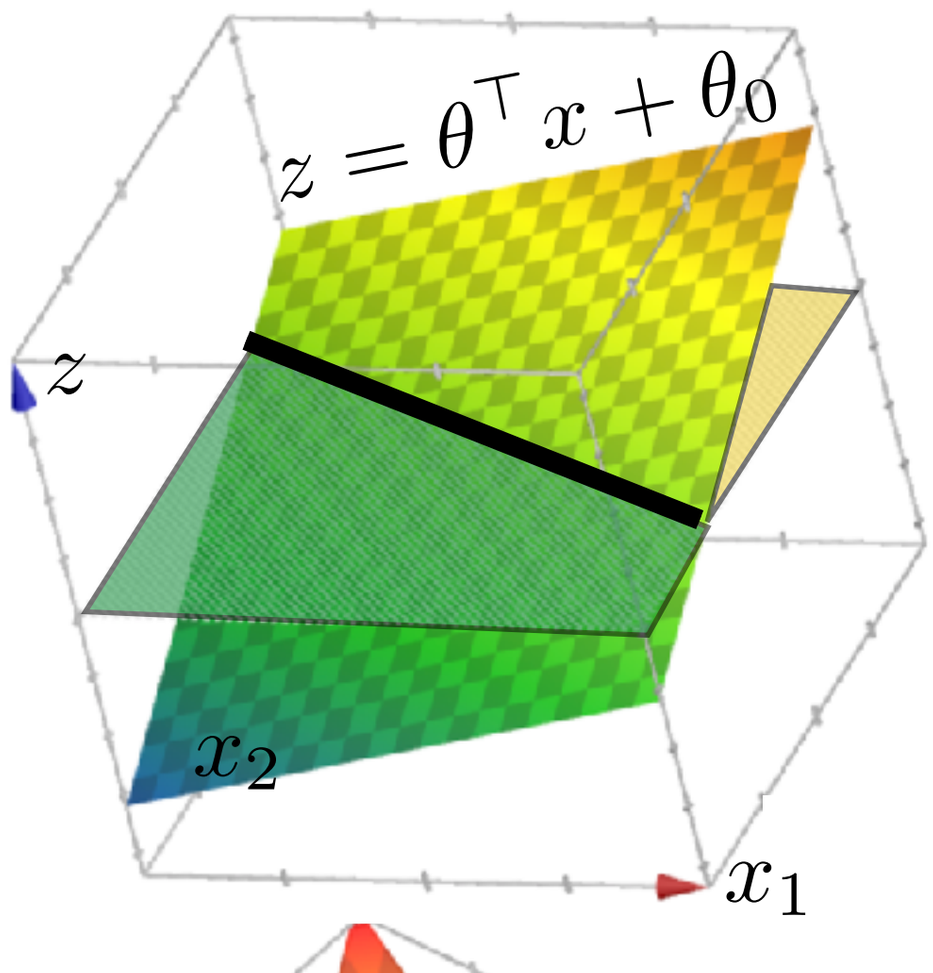
Recall

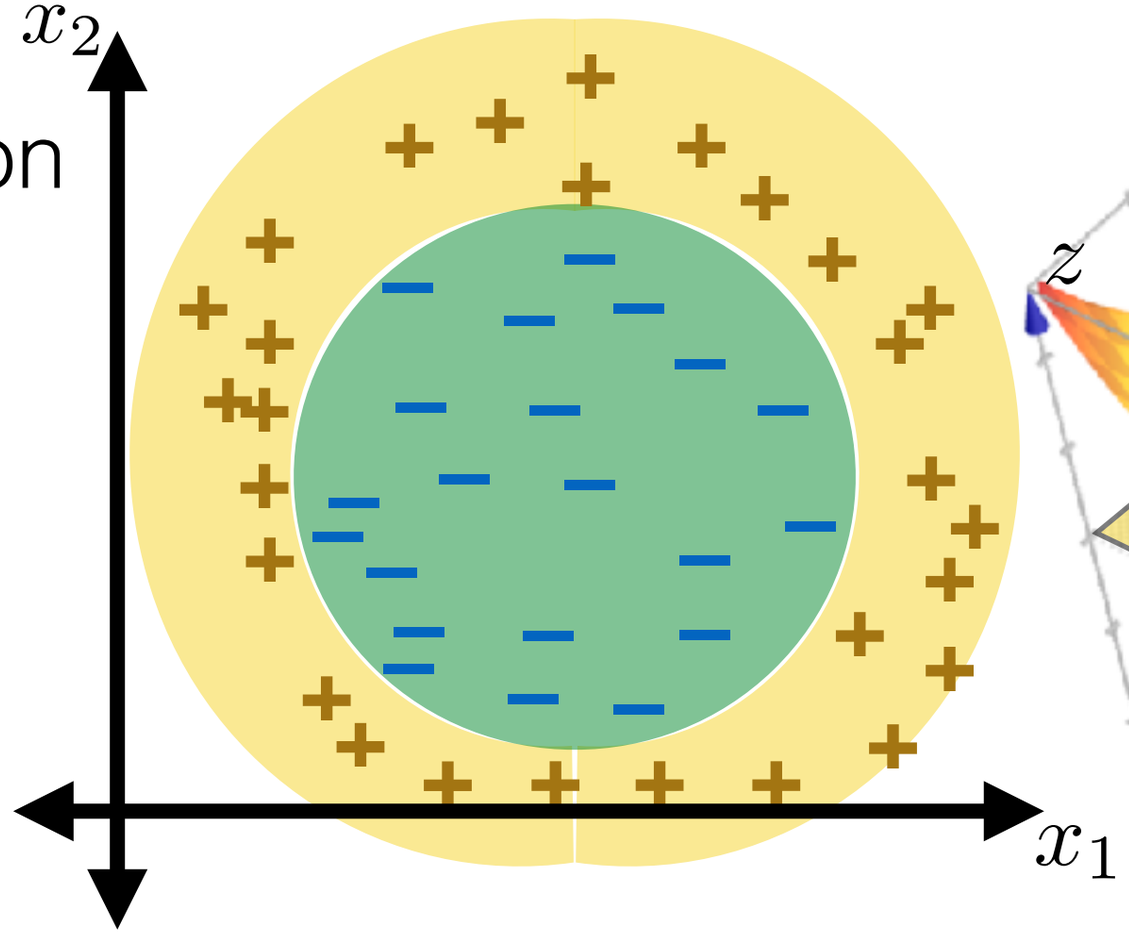
- Linear classification with default features:

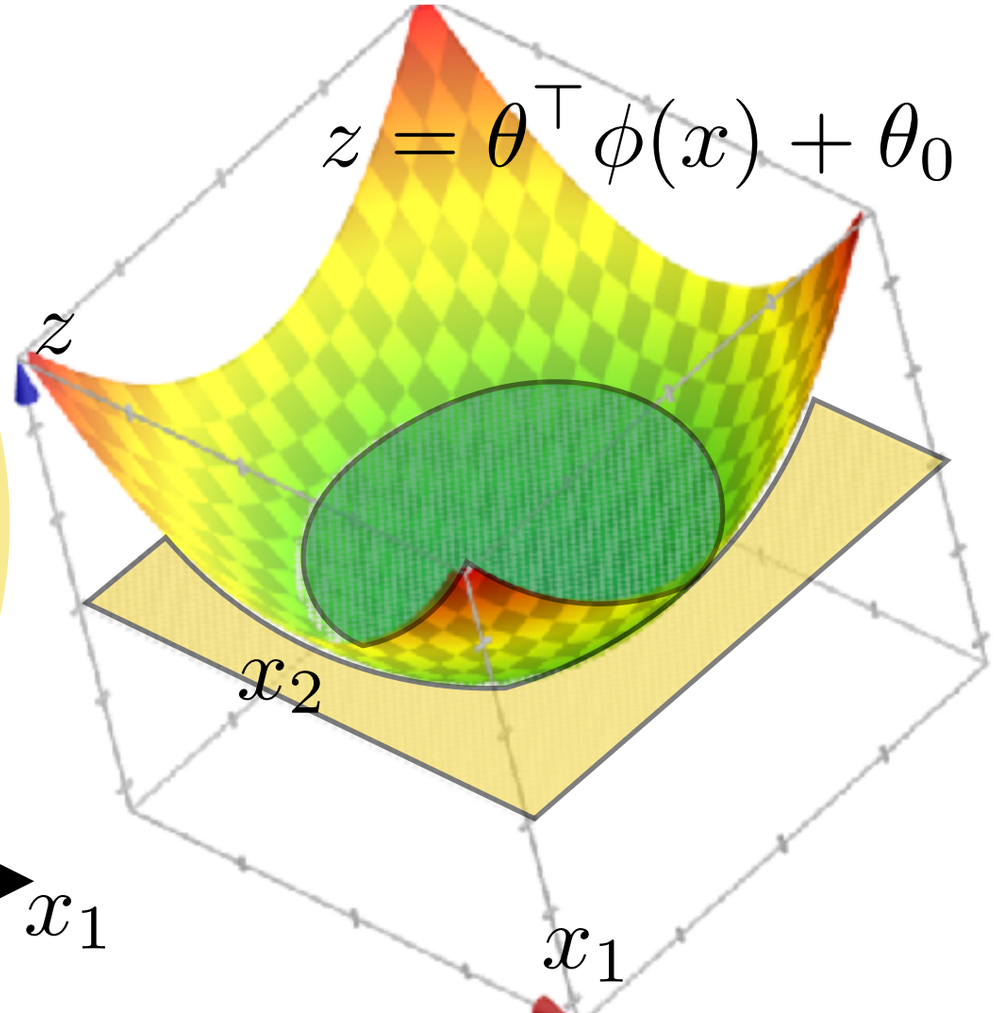




- Linear classification with polynomial features:

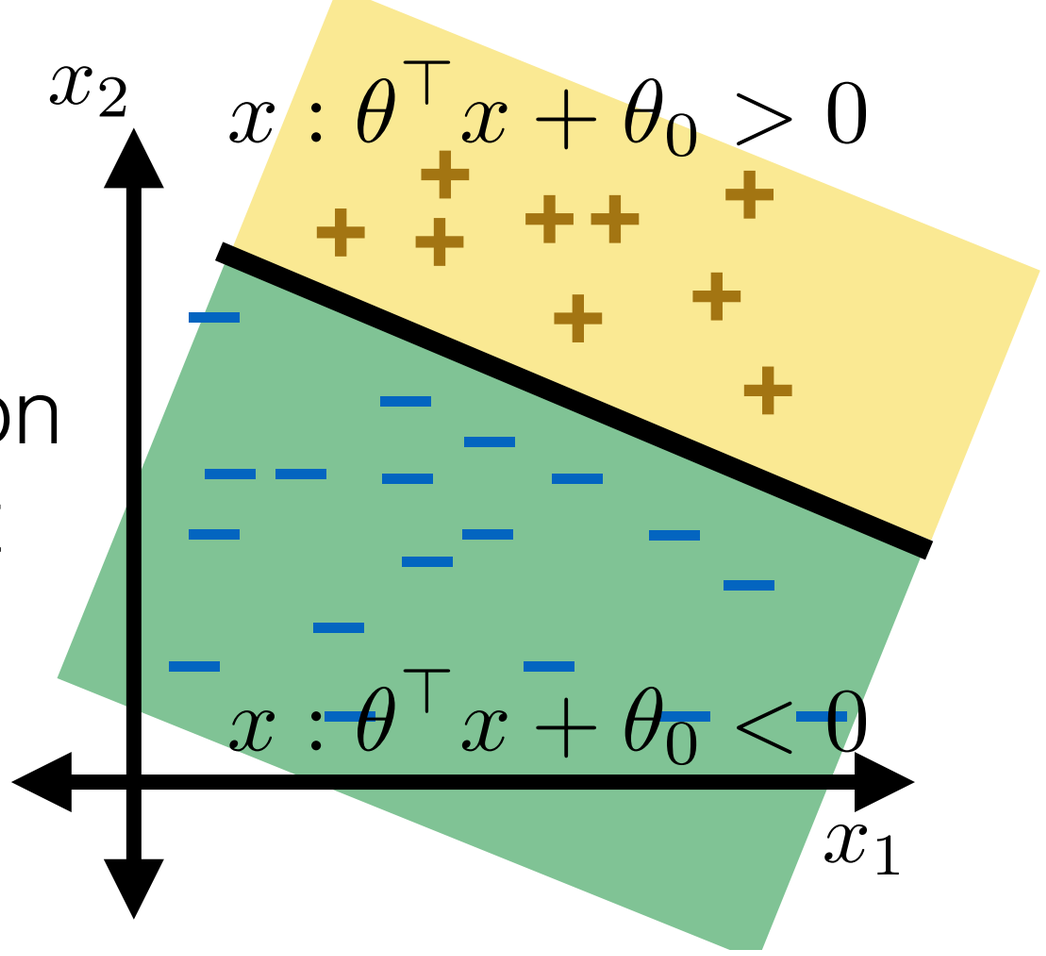
$$\phi(x) = [x_1, x_2, x_1^2, x_1 x_2, x_2^2]^\top$$

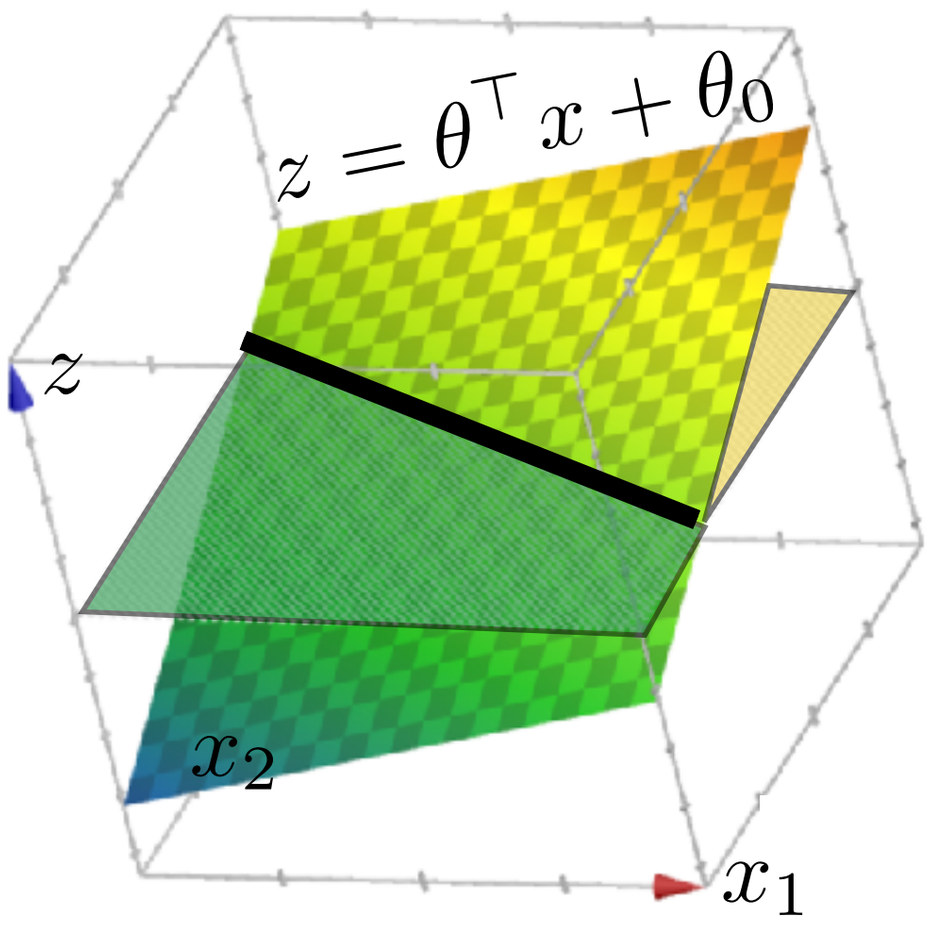


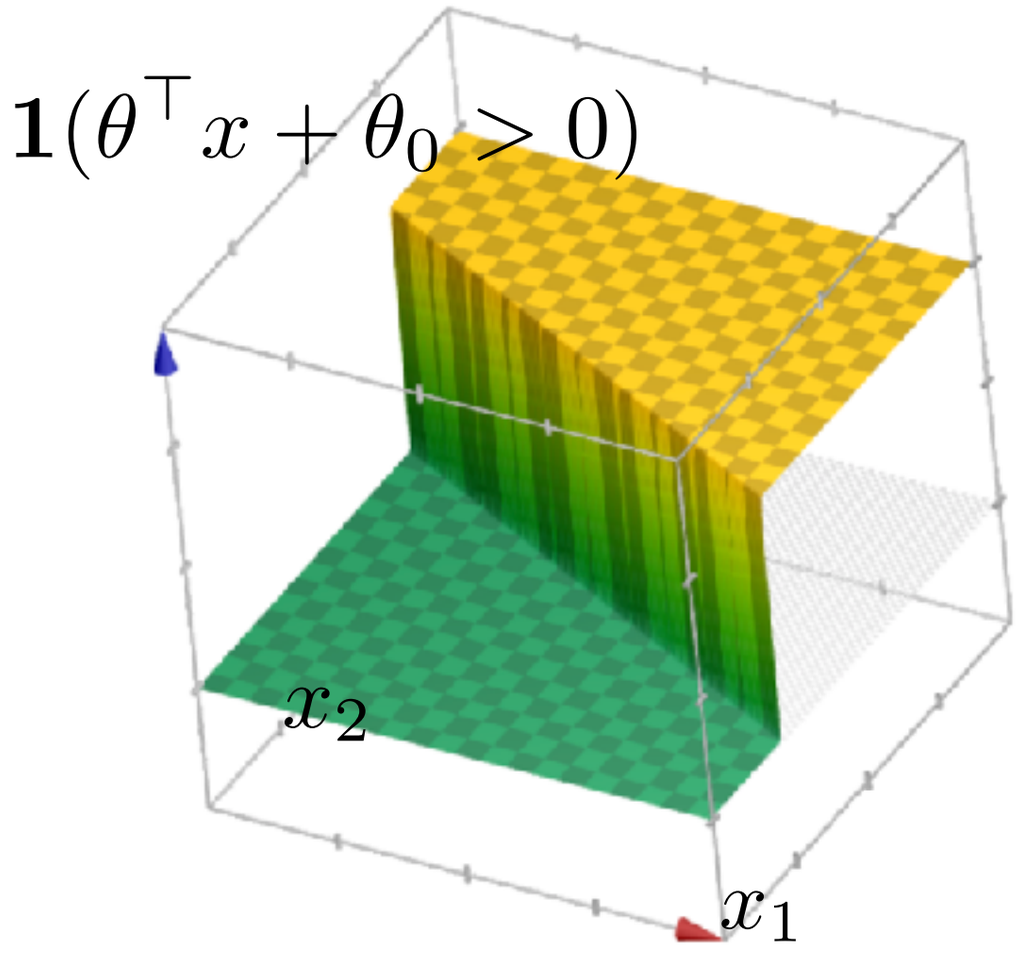


Recall

- Linear classification with default features:



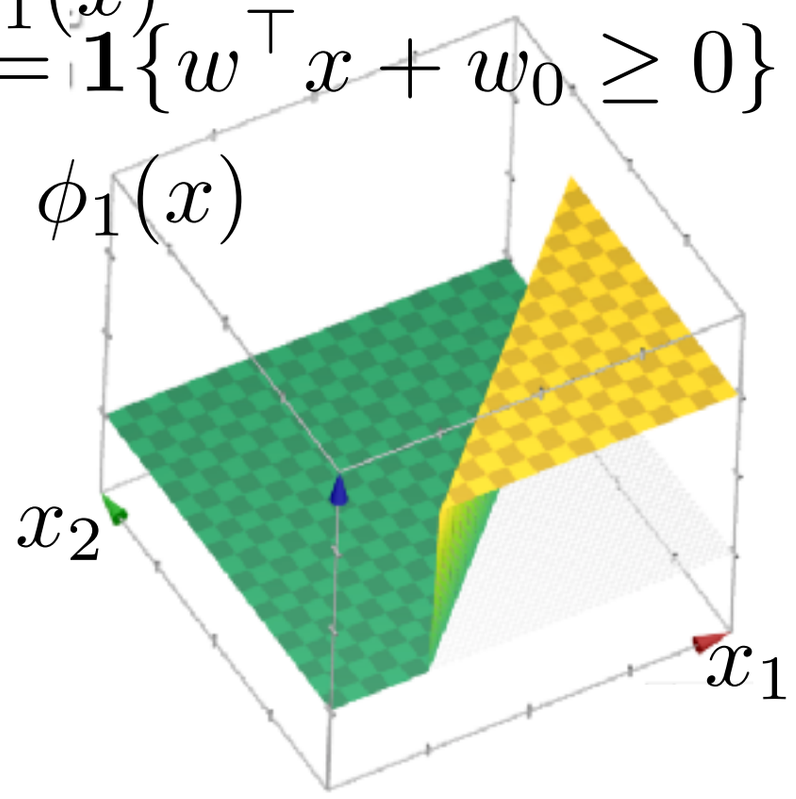




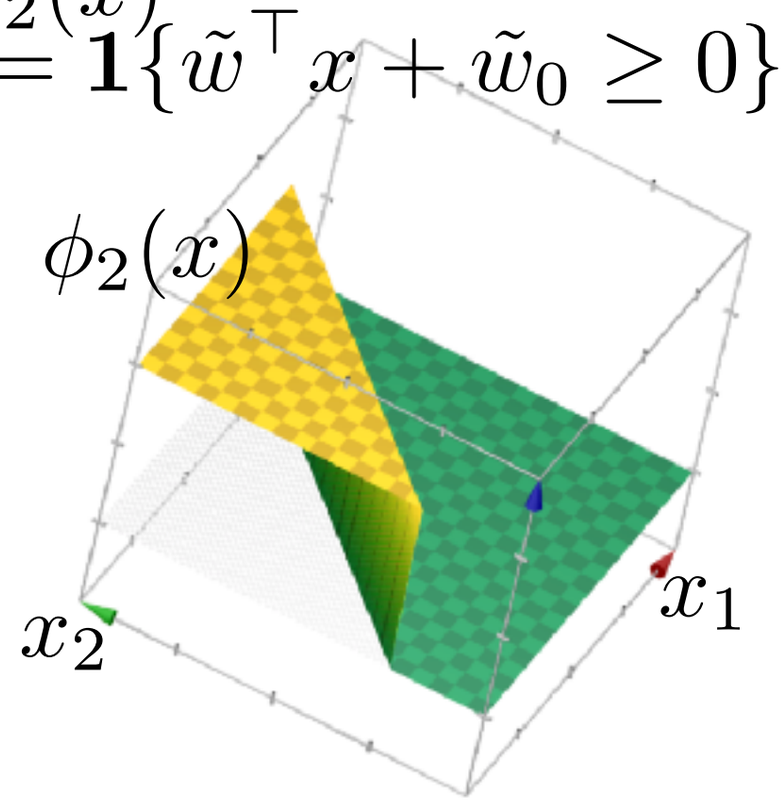
- We're used to using step functions to classify
- New idea today: we'll use step functions as *features*, with their own parameters

New features: step functions!

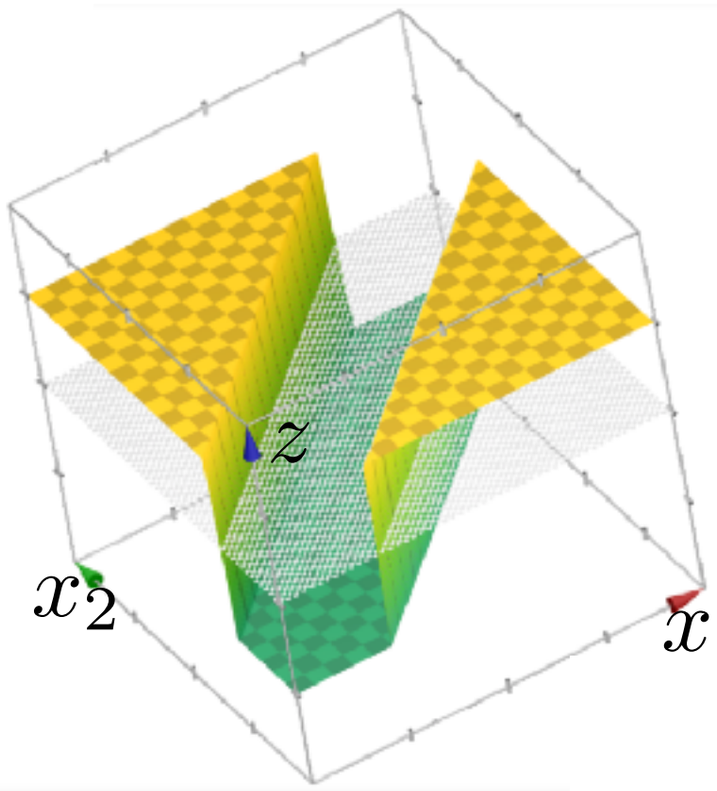
$$\phi_1(x) = \mathbf{1}\{w^\top x + w_0 \geq 0\}$$



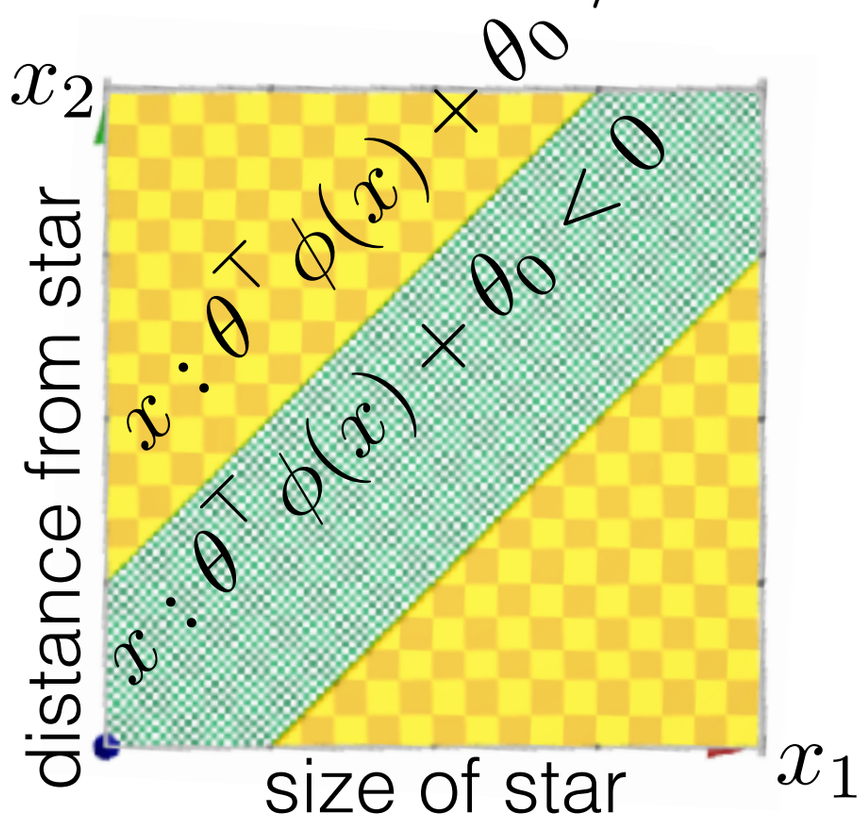
$$\phi_2(x) = \mathbf{1}\{\tilde{w}^\top x + \tilde{w}_0 \geq 0\}$$



$$z = \theta^\top \phi(x) + \theta_0$$
$$= \theta_1 \phi_1(x) + \theta_2 \phi_2(x) + \theta_0$$
$$= 1 \cdot \phi_1(x) + 1 \cdot \phi_2(x) + (-0.5)$$

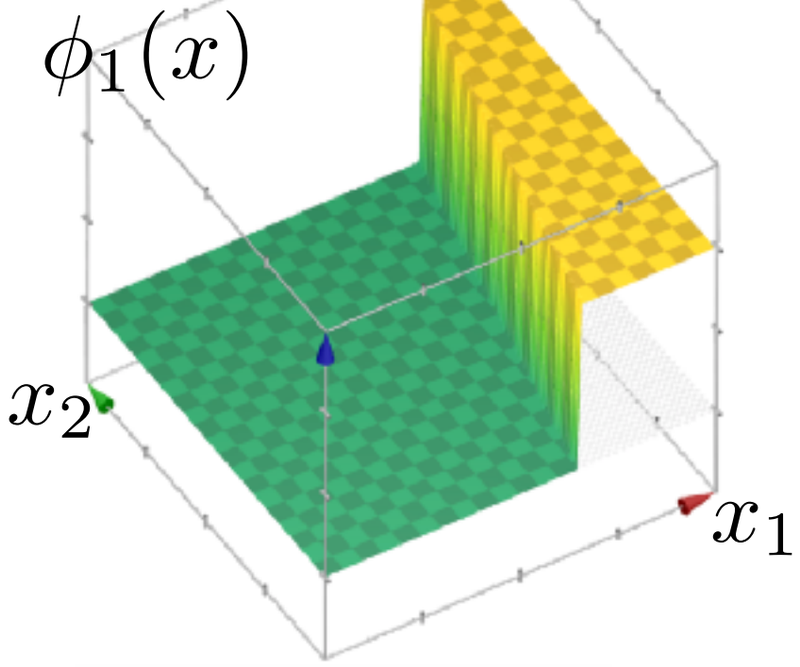


Is an exoplanet in the habitable zone?

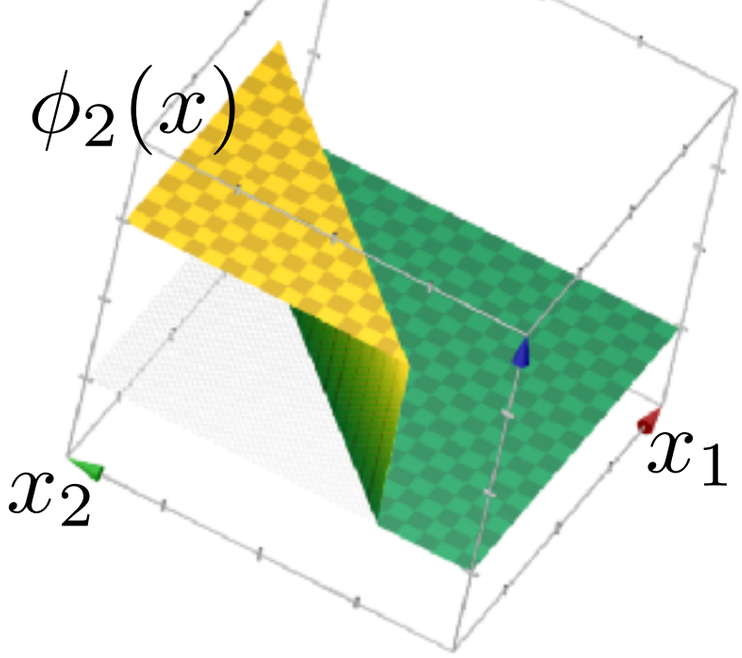


New features: step functions!

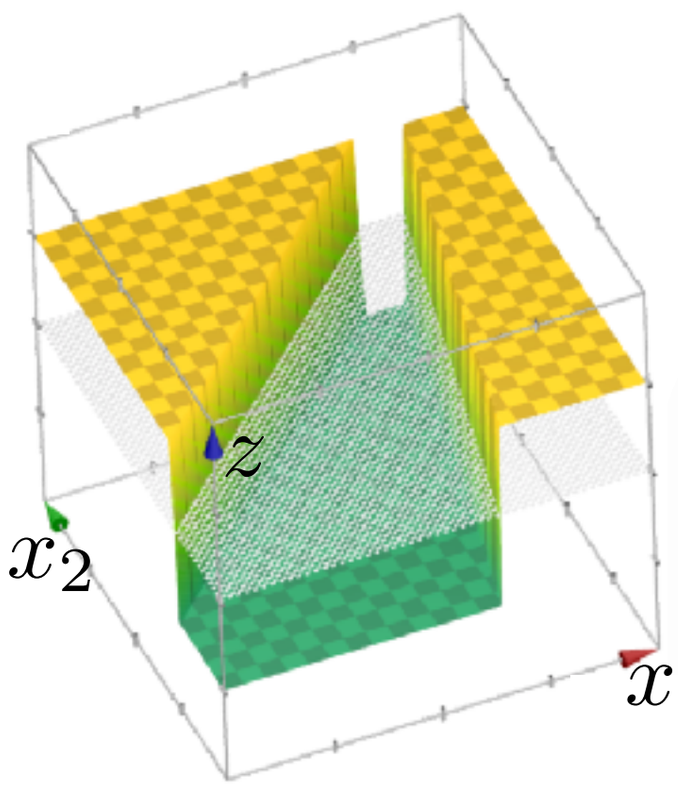
$$\phi_1(x) = \mathbf{1}\{w^\top x + w_0 \geq 0\}$$

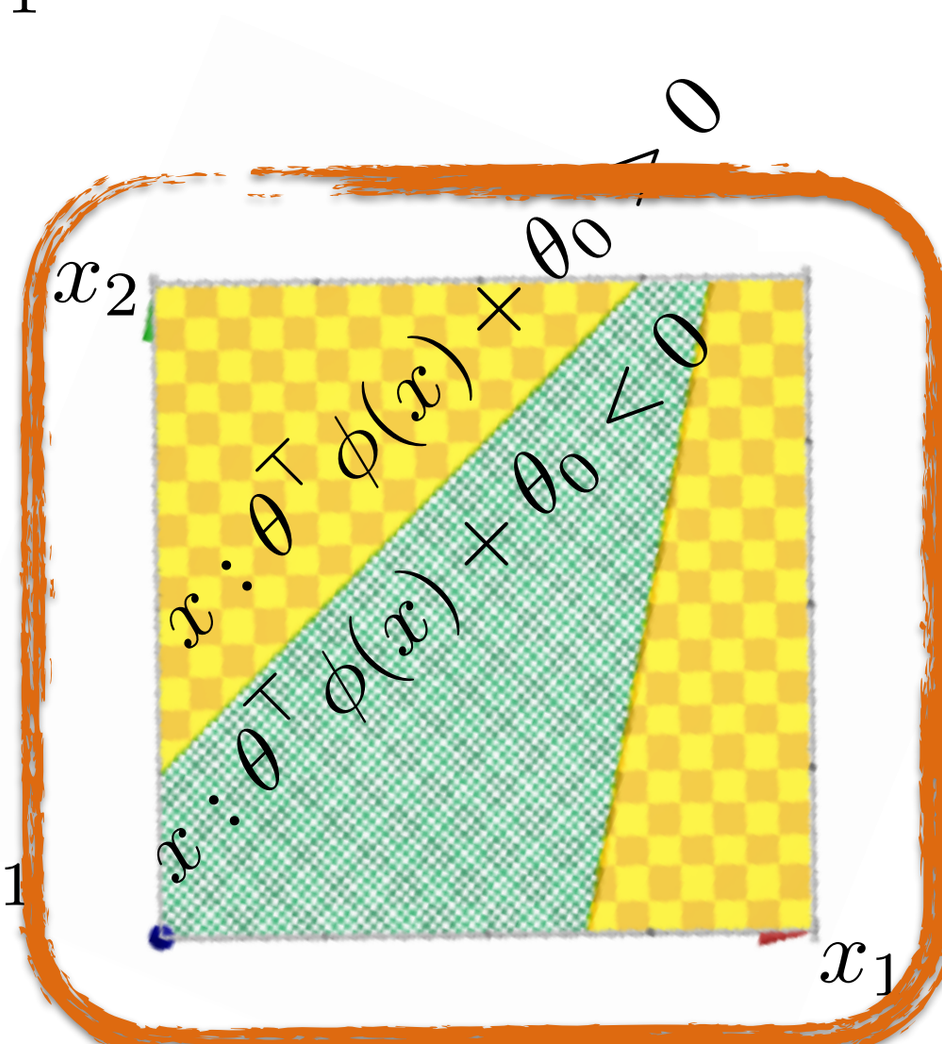


$$\phi_2(x) = \mathbf{1}\{\tilde{w}^\top x + \tilde{w}_0 \geq 0\}$$



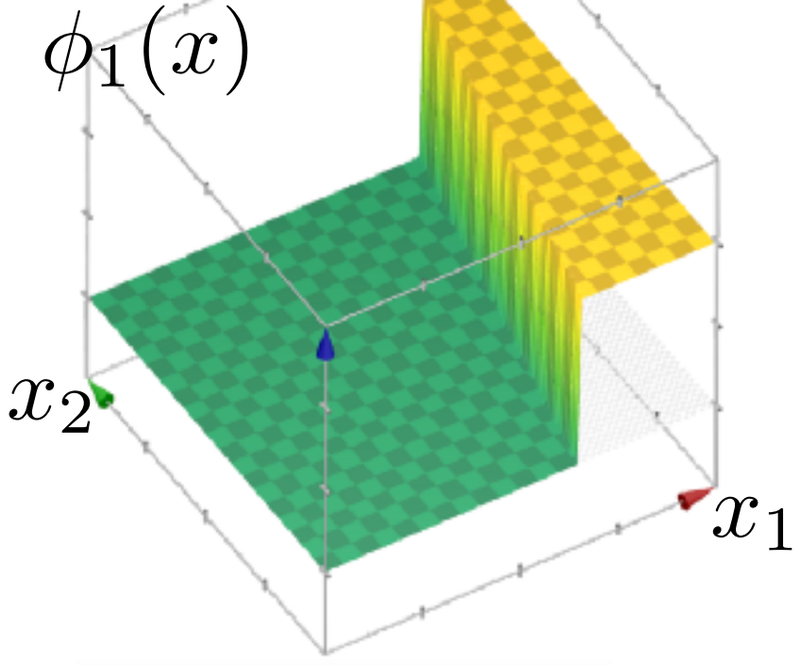
$$z = \theta^\top \phi(x) + \theta_0$$
$$= \theta_1 \phi_1(x) + \theta_2 \phi_2(x) + \theta_0$$
$$= 1 \cdot \phi_1(x) + 1 \cdot \phi_2(x) + (-0.5)$$



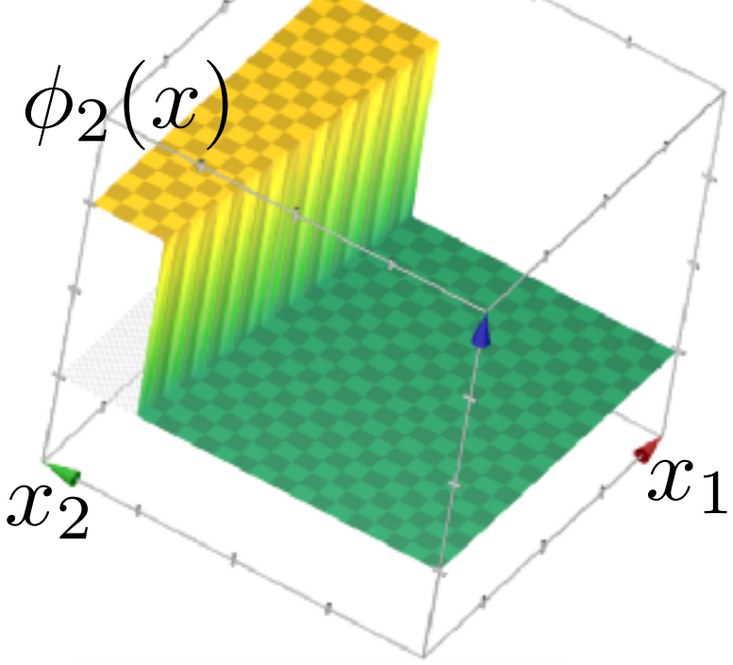


New features: step functions!

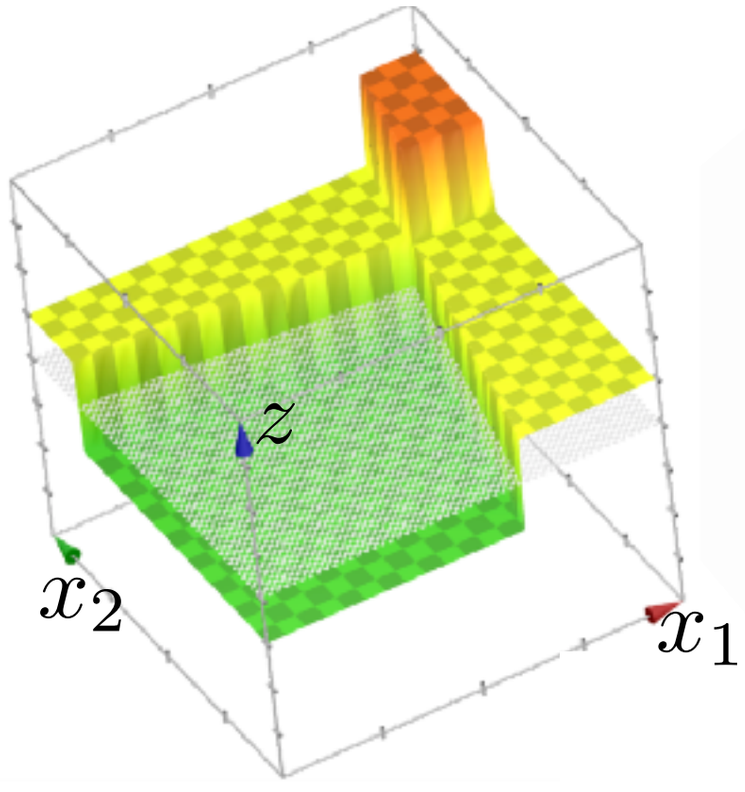
$$\phi_1(x) = \mathbf{1}\{w^\top x + w_0 \geq 0\}$$

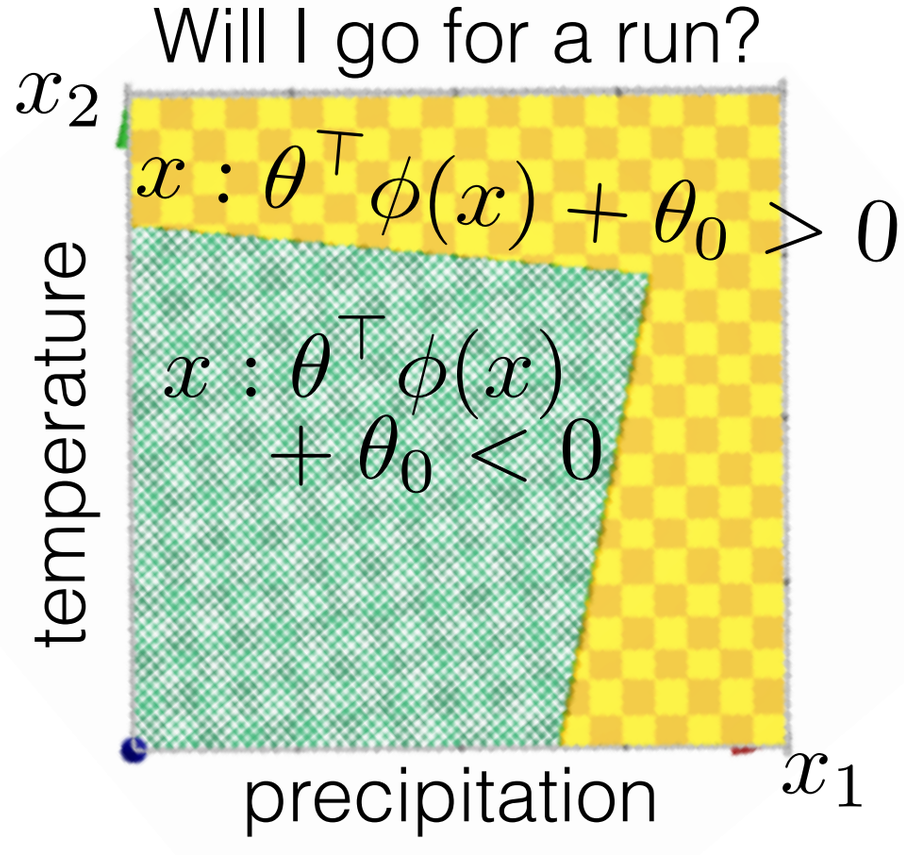


$$\phi_2(x) = \mathbf{1}\{\tilde{w}^\top x + \tilde{w}_0 \geq 0\}$$



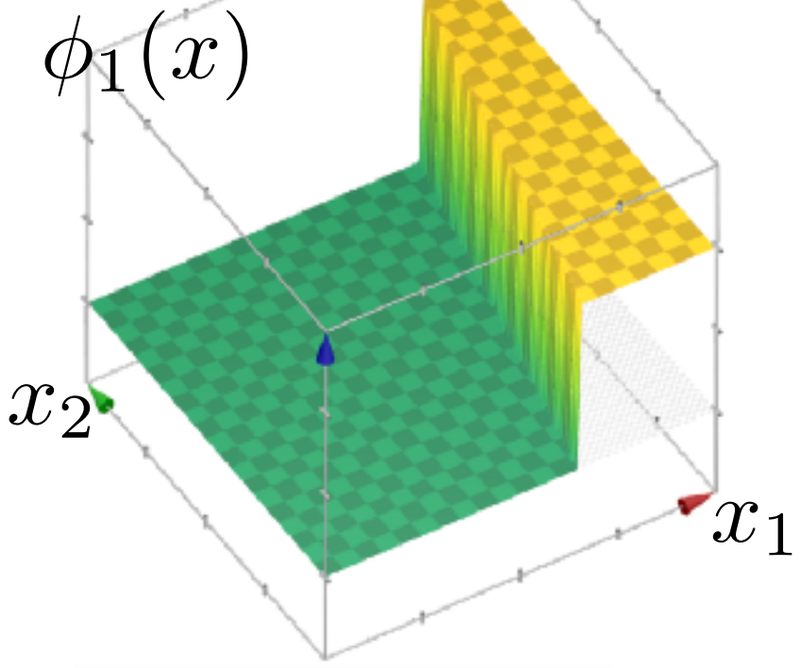
$$z = \theta^\top \phi(x) + \theta_0$$
$$= \theta_1 \phi_1(x) + \theta_2 \phi_2(x) + \theta_0$$
$$= 1 \cdot \phi_1(x) + 1 \cdot \phi_2(x) + (-0.5)$$



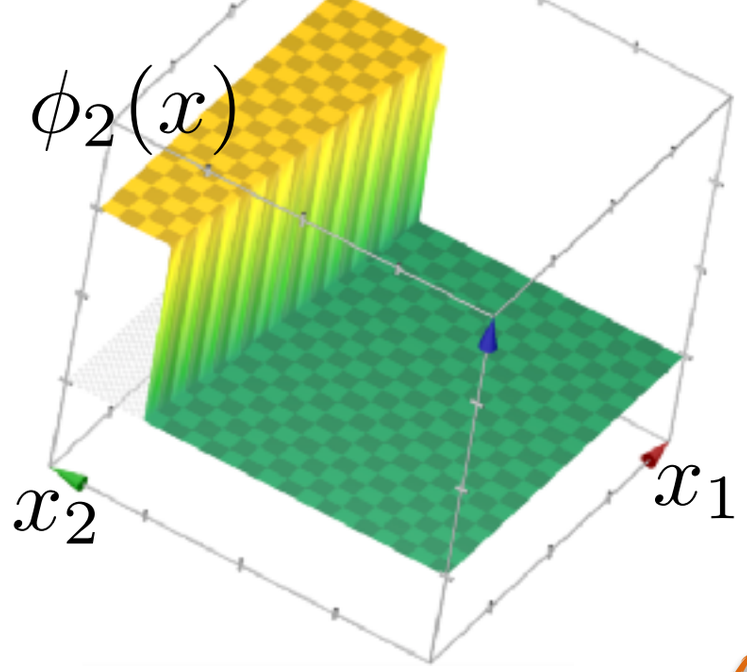


New features: step functions!

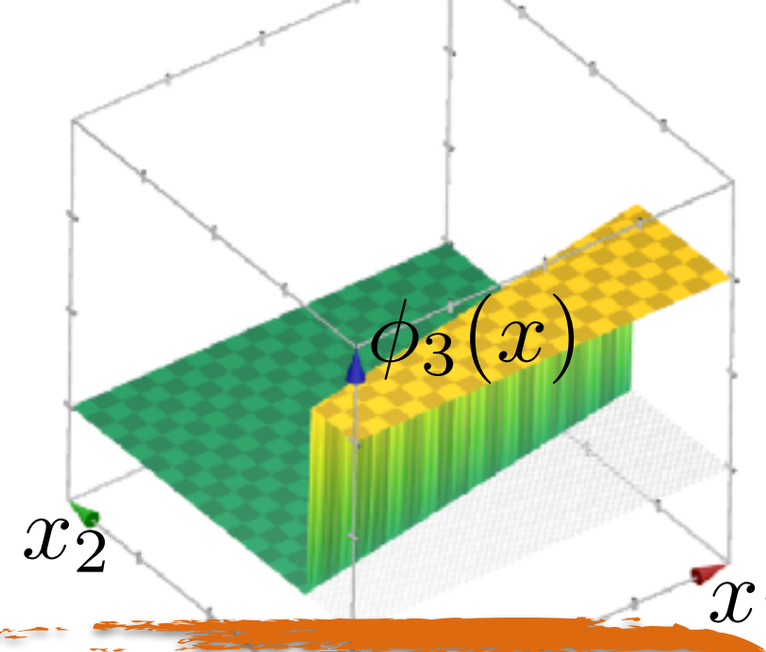
$$\phi_1(x) = \mathbf{1}\{w^\top x + w_0 \geq 0\}$$



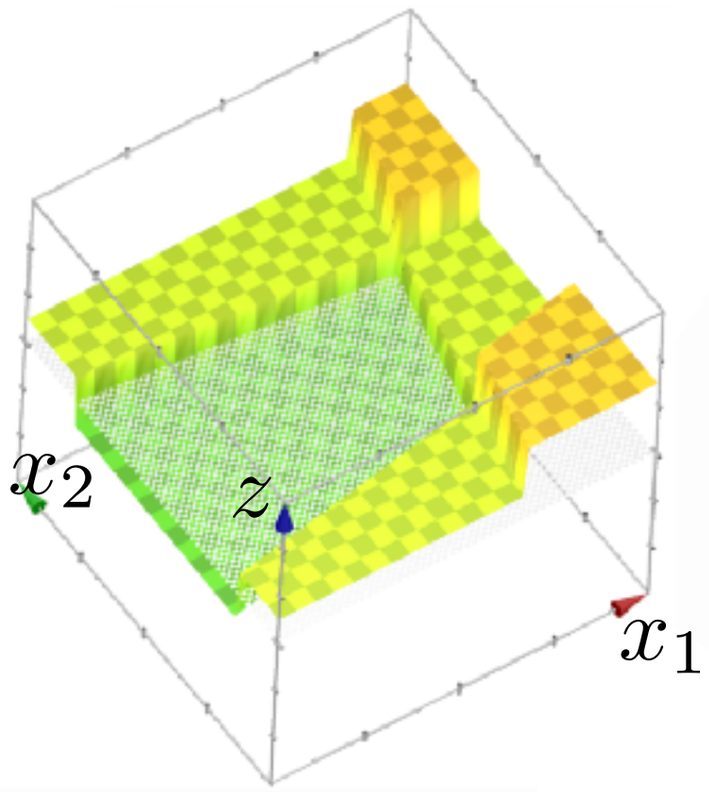
$$\phi_2(x) = \mathbf{1}\{\tilde{w}^\top x + \tilde{w}_0 \geq 0\}$$

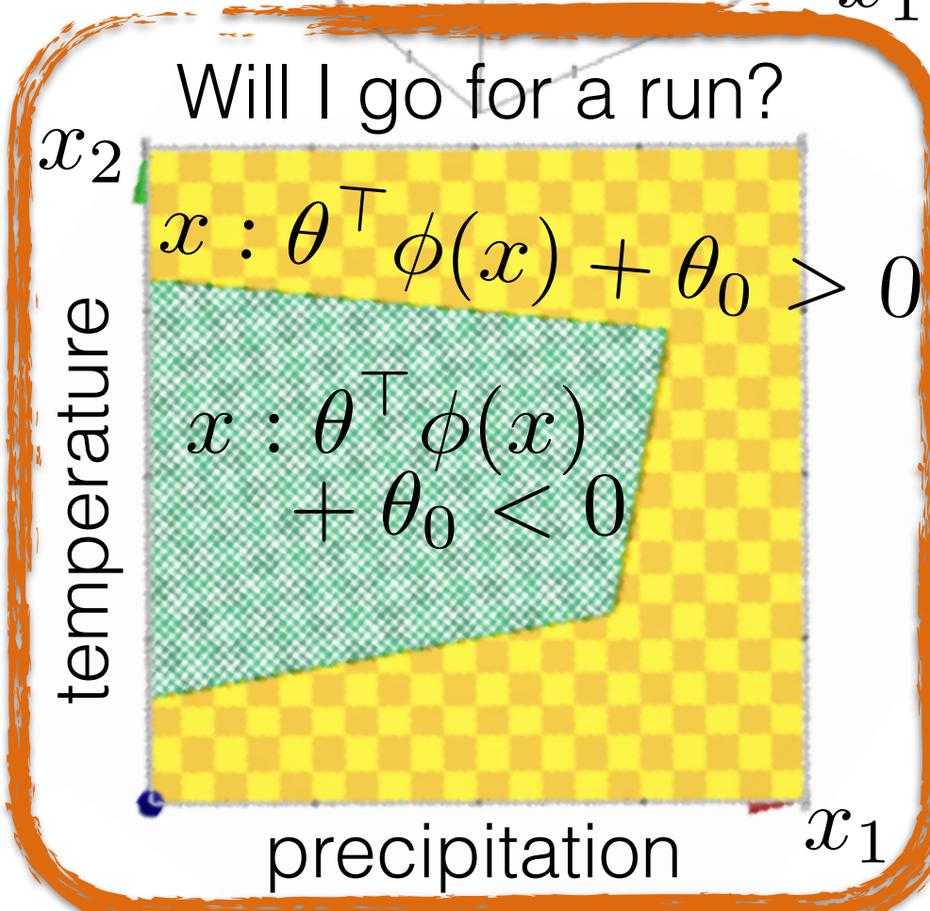


$$\phi_3(x) = \mathbf{1}\{\tilde{\tilde{w}}^\top x + \tilde{\tilde{w}}_0 \geq 0\}$$



$$\begin{aligned} z &= \theta^\top \phi(x) + \theta_0 \\ &= \theta_1 \phi_1(x) + \theta_2 \phi_2(x) \\ &\quad + \theta_3 \phi_3(x) + \theta_0 \\ &= 1 \cdot \phi_1(x) + 1 \cdot \phi_2(x) \\ &\quad + 1 \cdot \phi_3(x) + (-0.5) \end{aligned}$$

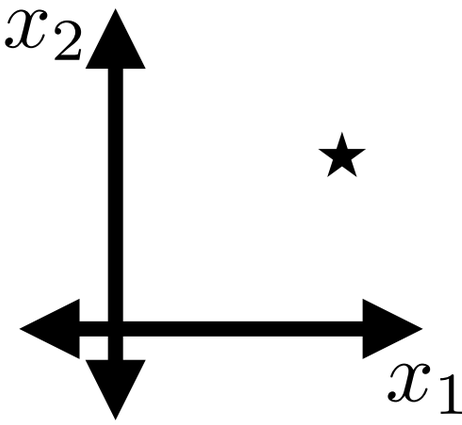


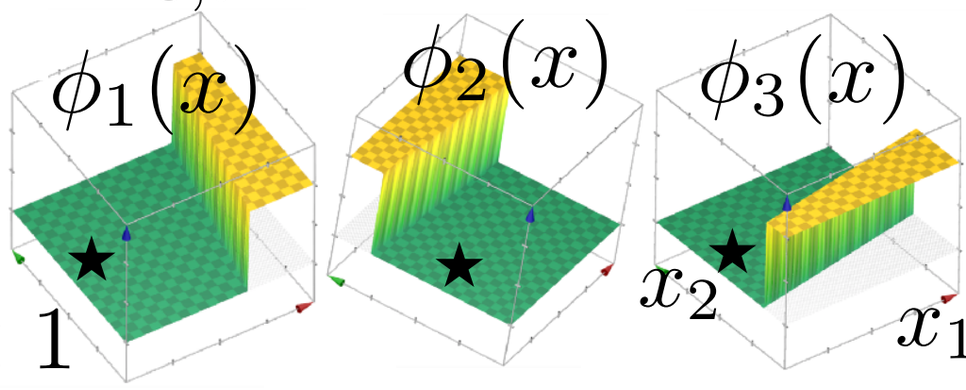


Let's get some new notation

- 1st layer, constructing the features:
  - Input x (a data point): size $m^{(1)} \times 1$
  - Output $A^{(1)}$ (vector of features): size $n^{(1)} \times 1$
  - The ith feature: $A_i^{(1)} = f^{(1)}(w_i^{(1)\top} x + w_{0,i}^{(1)})$
    - All the features at once:
      - $A^{(1)} = f^{(1)}(W^{(1)\top} x + W_0^{(1)})$
      - $W^{(1)} : m^{(1)} \times n^{(1)}; W_0^{(1)} : n^{(1)} \times 1$

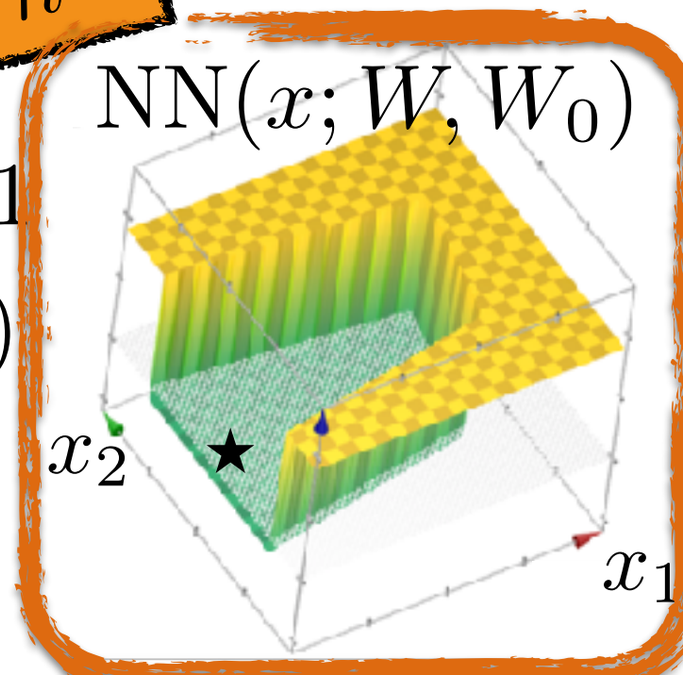
$m^{(1)} = d$





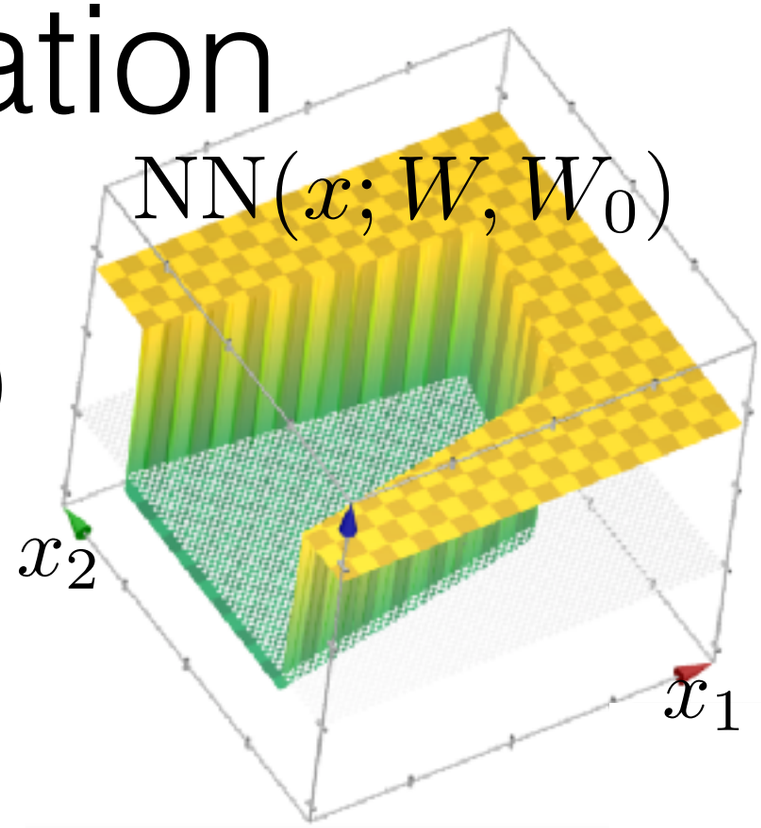
- 2nd layer, assigning a label (or labels):
  - Input (the features): size $m^{(2)} \times 1$
  - Output $A^{(2)}$(vector of labels): size $n^{(2)} \times 1$
  - The ith label: $A_i^{(2)} = f^{(2)}(w_i^{(2)\top} A^{(1)} + w_{0,i}^{(2)})$
    - All: $A^{(2)} = f^{(2)}(W^{(2)\top} A^{(1)} + W_0^{(2)})$
    - $W^{(2)} : m^{(2)} \times n^{(2)}; W_0^{(2)} : n^{(2)} \times 1$

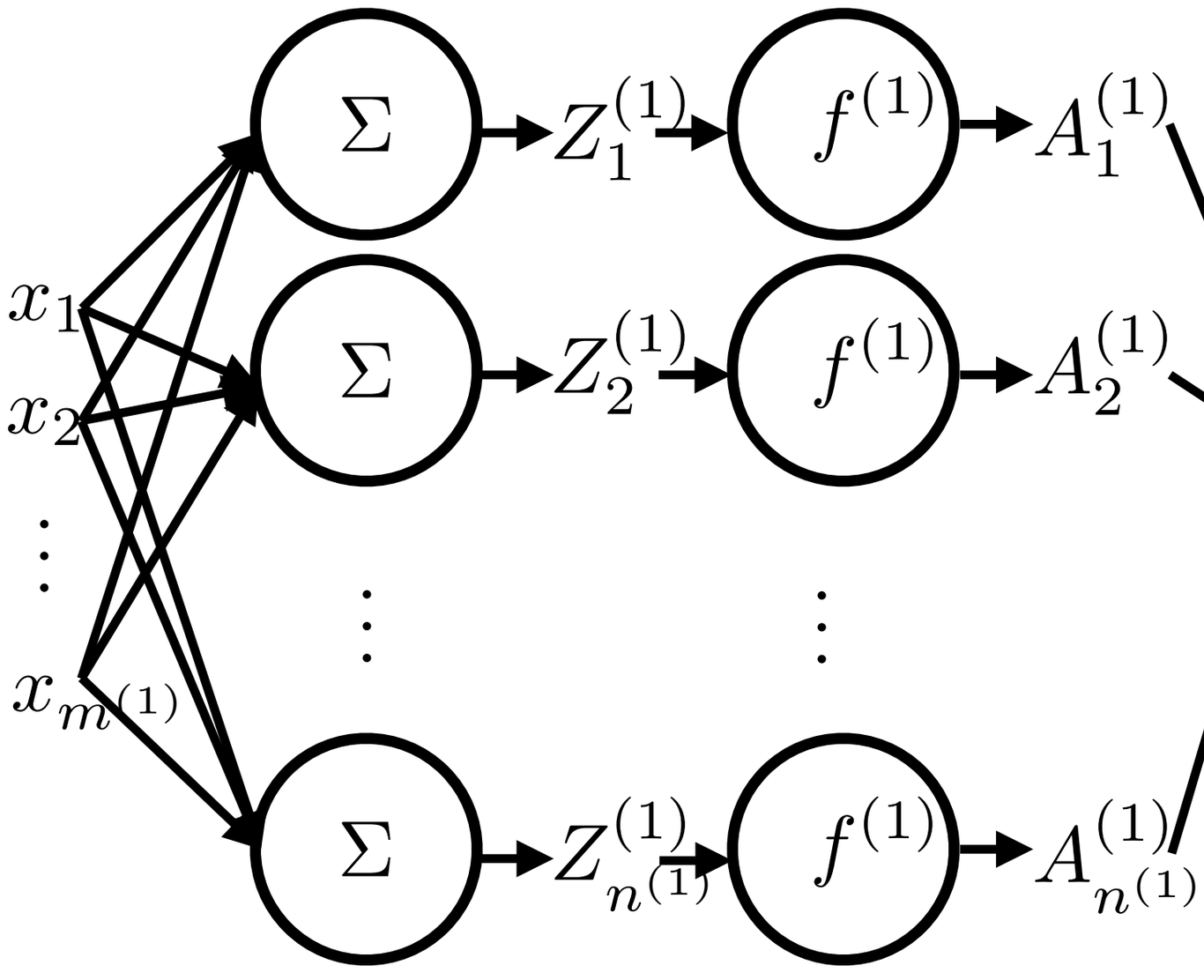
$m^{(2)} = n^{(1)}$



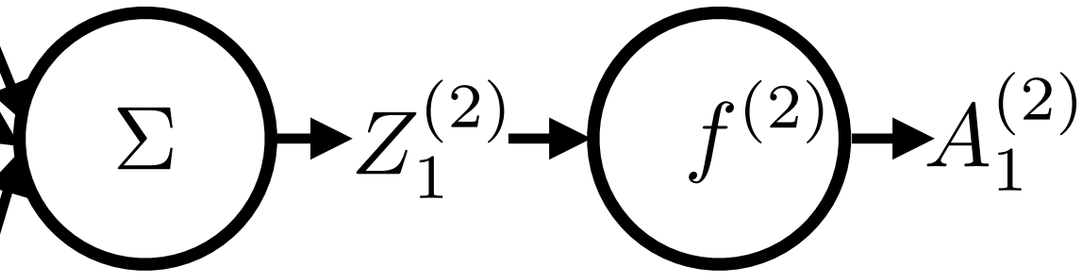
Function graph representation

- 1st layer: $A^{(1)} = f^{(1)}(W^{(1)\top} x + W_0^{(1)})$
- 2nd layer: $A^{(2)} = f^{(2)}(W^{(2)\top} A^{(1)} + W_0^{(2)})$
- Whole thing: $A^{(2)} = \mathrm{NN}(x; W, W_0)$





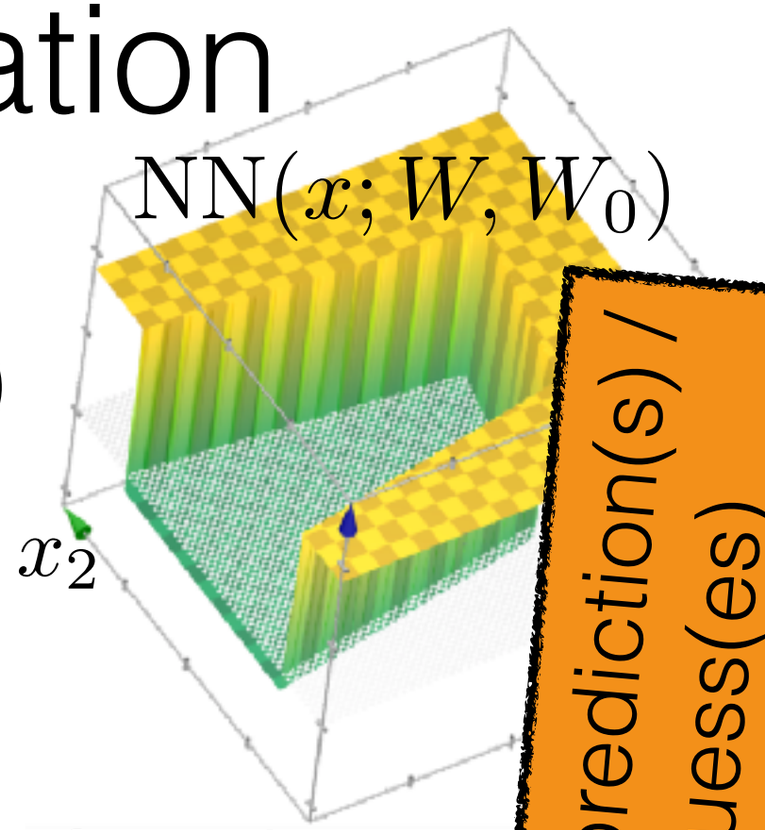
- Circle: function evaluation

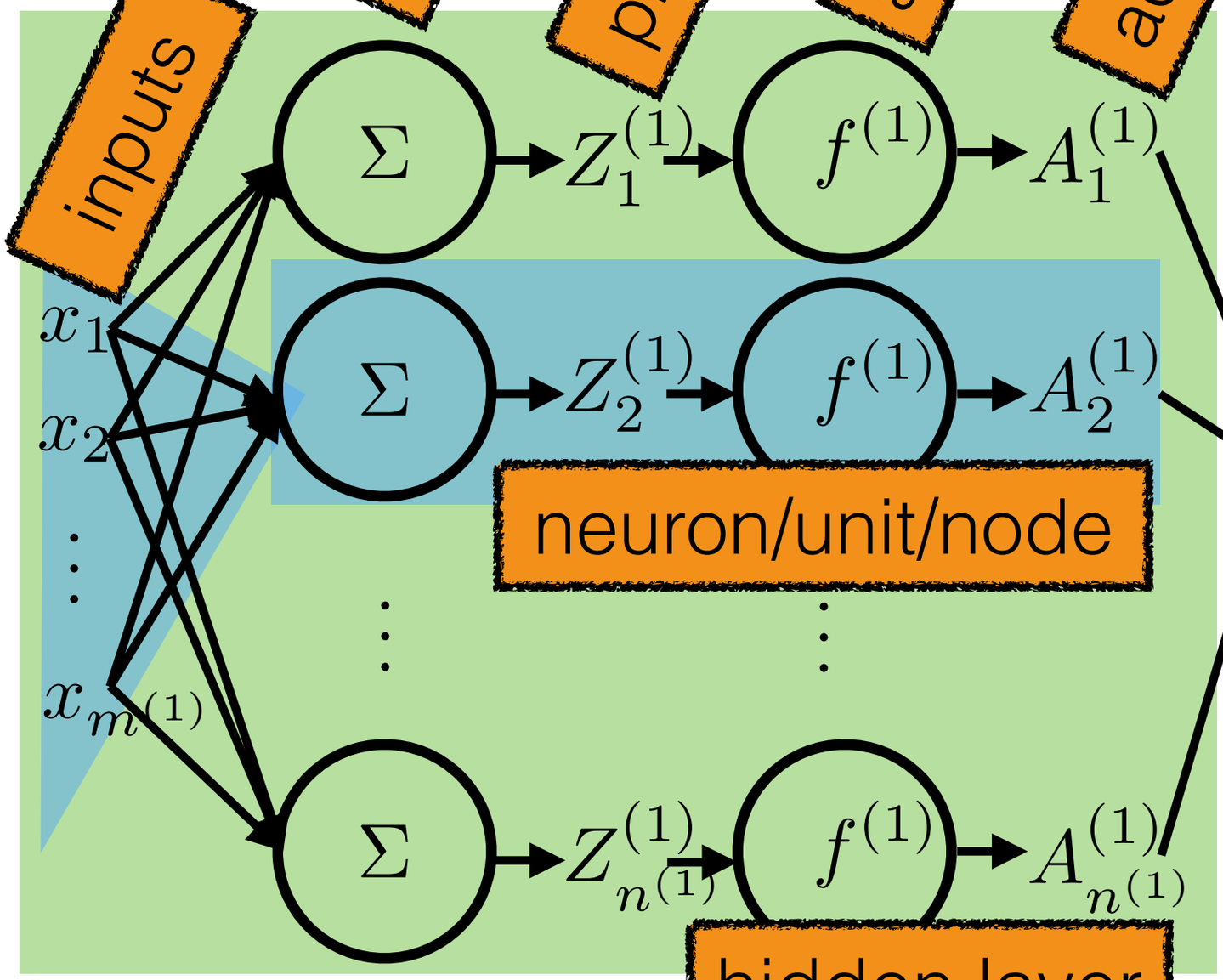


- Forward vs. backward
- A feed-forward neural network

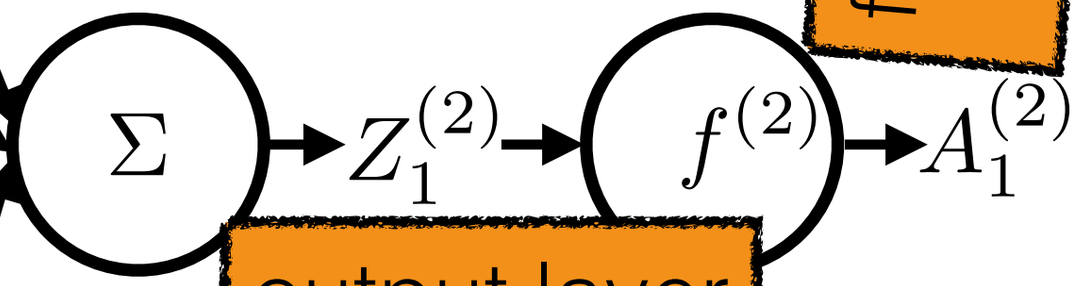
Function graph representation

- 1st layer: $A^{(1)} = f^{(1)}(W^{(1)\top} x + W_0^{(1)})$
- 2nd layer: $A^{(2)} = f^{(2)}(W^{(2)\top} A^{(1)} + W_0^{(2)})$
- Whole thing: $A^{(2)} = \mathrm{NN}(x; W, W_0)$





- Circle: function evaluation



- Forward vs. backward
- A feed-forward neural network
- Fully connected

Problem setup

1. Choose a hypothesis class. E.g.,

$$h(x; W, W_0) = \mathrm{NN}(x; W, W_0)$$

   - 1st layer: $A^{(1)} = f^{(1)}(W^{(1)\top} x + W_0^{(1)})$
   - 2nd layer: $A^{(2)} = f^{(2)}(W^{(2)\top} A^{(1)} + W_0^{(2)})$

2. Choose a loss. E.g. for classification: 0-1 loss, asymmetric, negative log likelihood
3. Learn the parameters. E.g. gradient descent or SGD
4. Predict on new data using these parameters

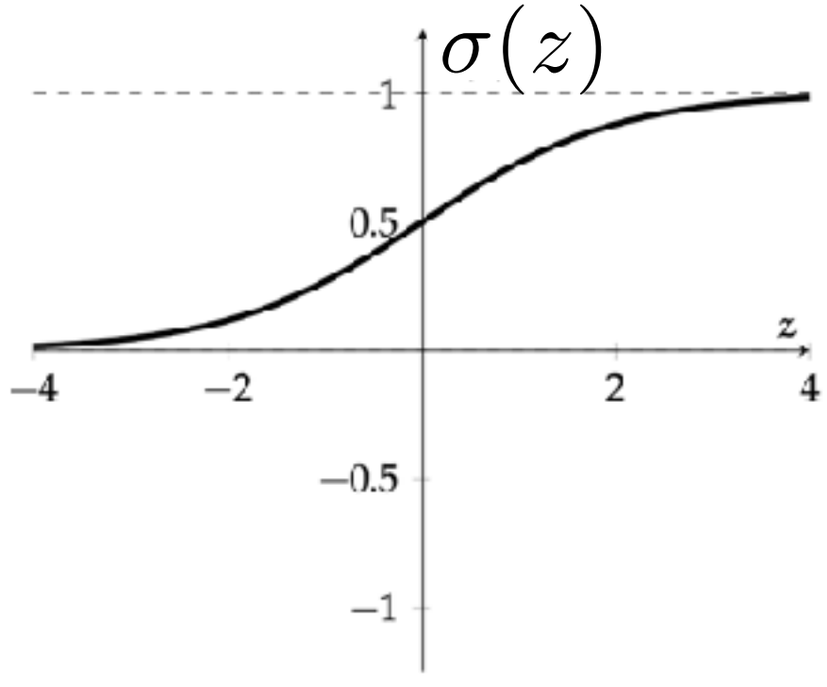
Issues:

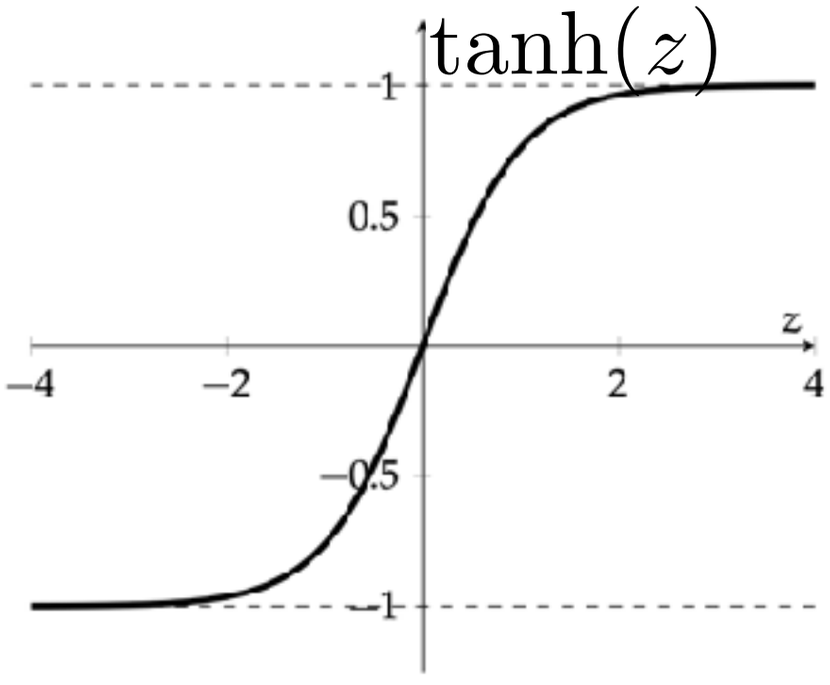
- Derivatives are zero (or undefined) if we use the step function activation, so (S)GD won't do what we want
- What if I want to do regression?
- What if I want to use NLL loss?

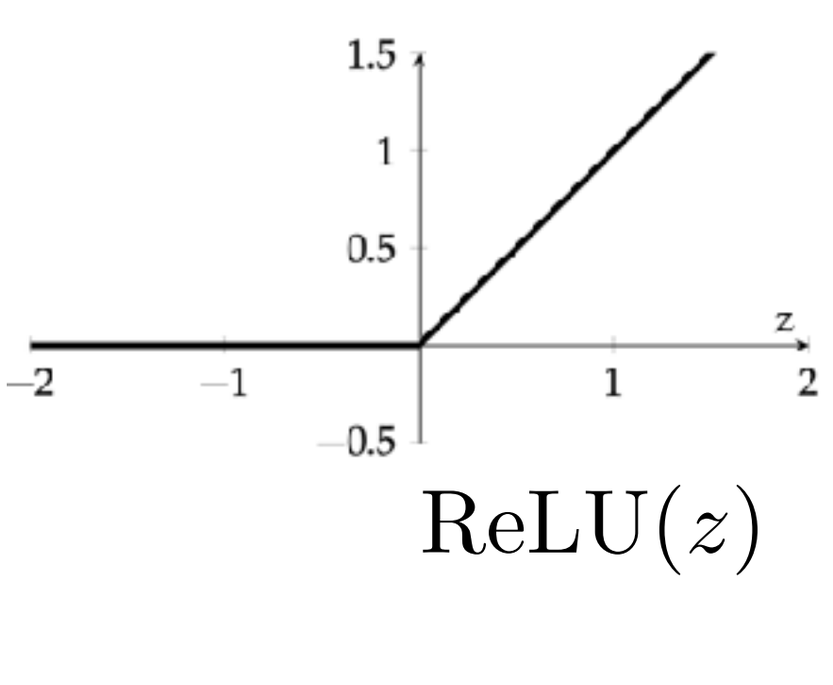
Different activation functions

1. Hypotheses $h(x; W, W_0) = \mathrm{NN}(x; W, W_0)$
   - 1st layer: $A^{(1)} = f^{(1)}(W^{(1)\top} x + W_0^{(1)})$
   - 2nd layer: $A^{(2)} = f^{(2)}(W^{(2)\top} A^{(1)} + W_0^{(2)})$

- What if I want to do regression? $f^{(2)}(z) = z$
- What if I want to use NLL loss? $f^{(2)}(z) = \sigma(z)$
- Need non-zero derivatives for (S)GD: Above & $f^{(1)}(z) =$

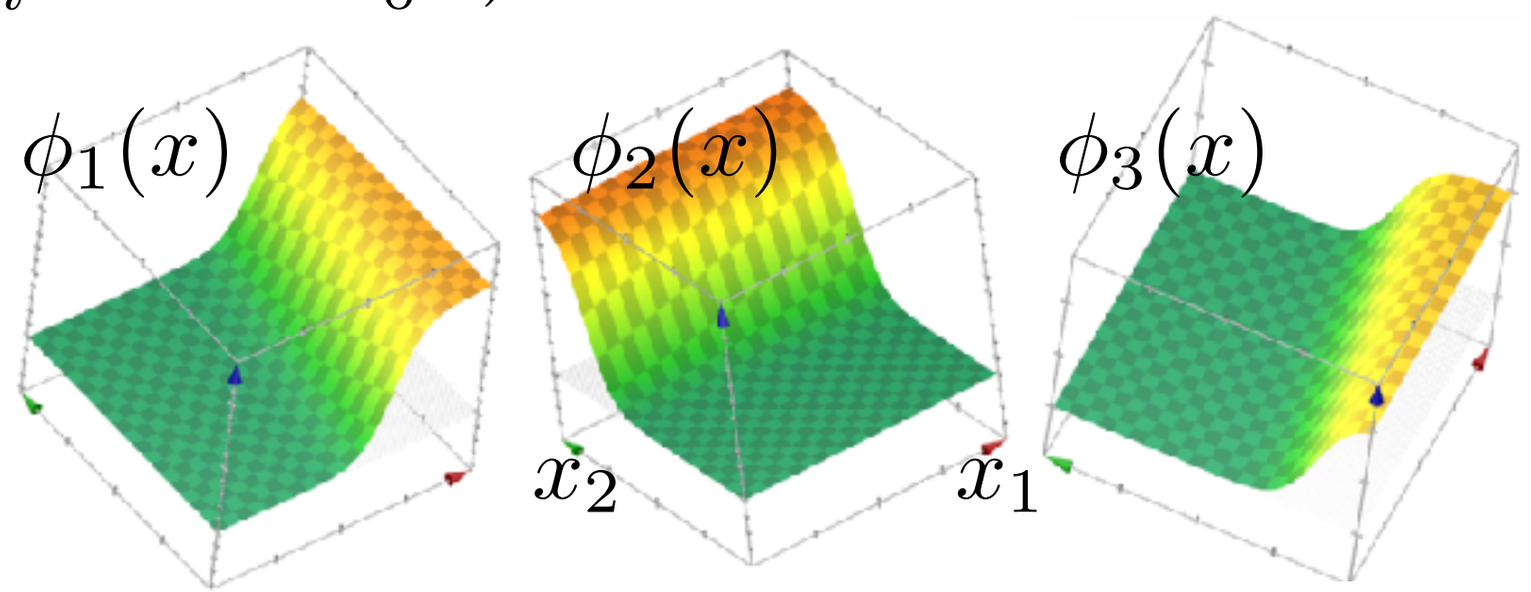




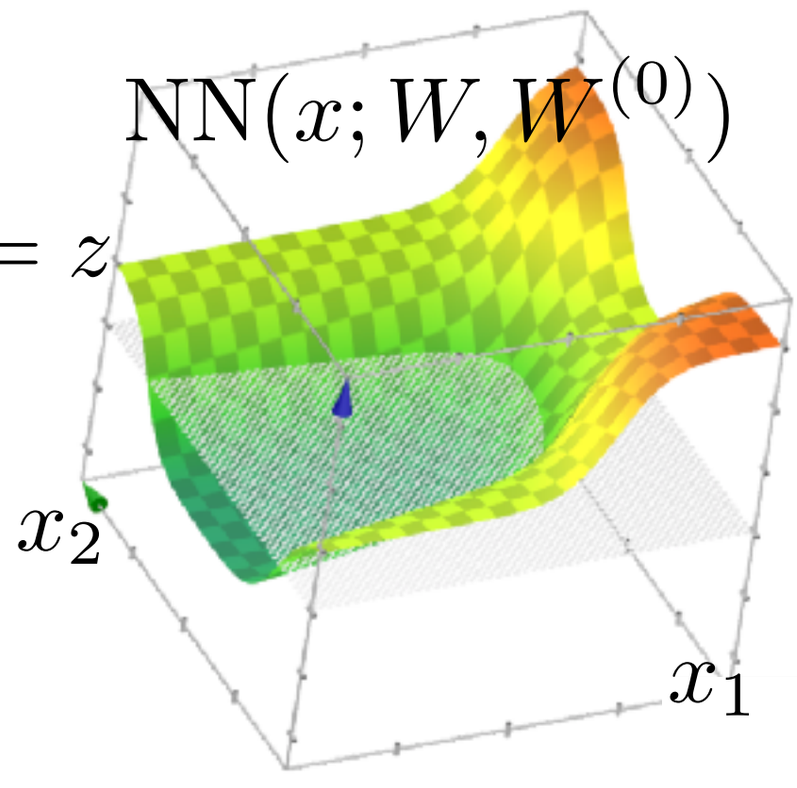


Choices of activation function

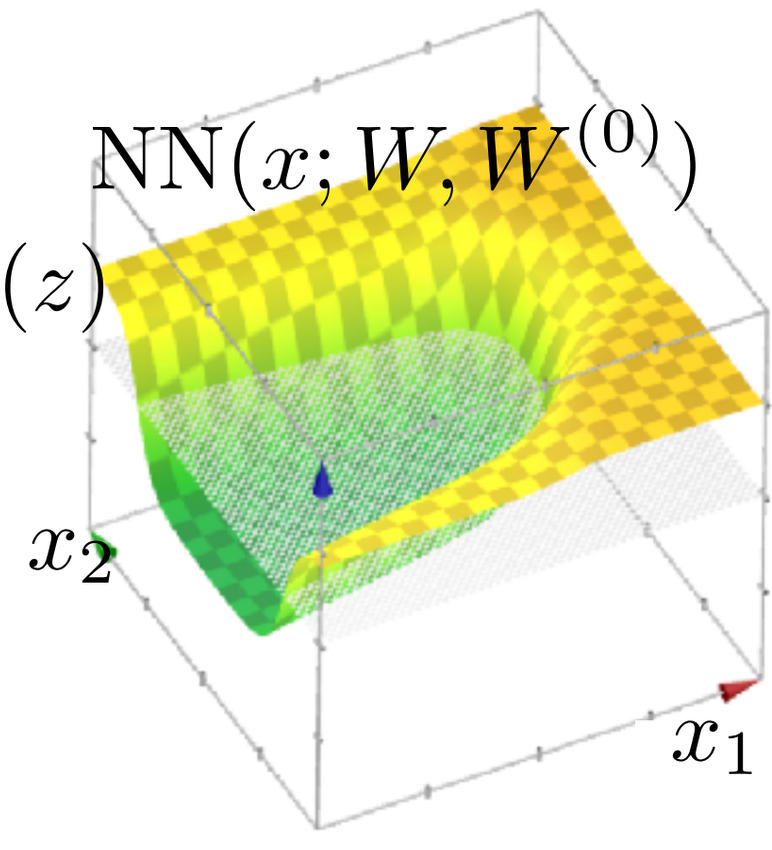
- 1st layer: $A_i^{(1)} = f^{(1)}(w_i^{(1)\top} x + w_0^{(1)})$
  - Choose $f^{(1)}(z) = \sigma(z)$



- 2nd layer: $A_i^{(2)} = f^{(2)}(w_i^{(2)\top} A^{(1)} + w_0^{(2)})$
  - Choose $f^{(2)}(z) = z$



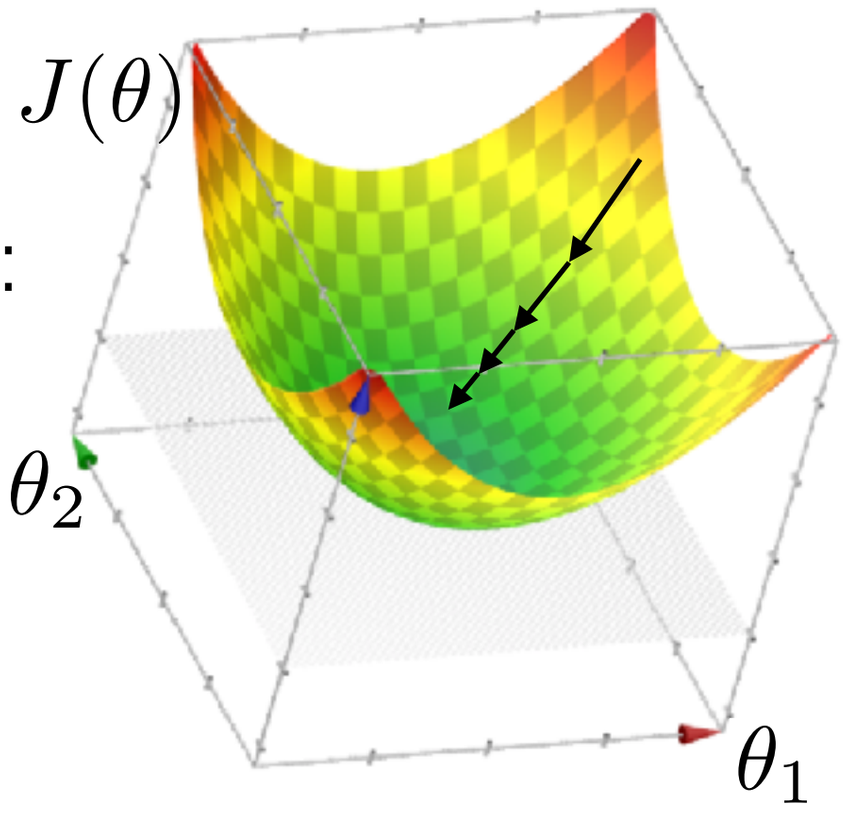
  - Choose $f^{(2)}(z) = \sigma(z)$



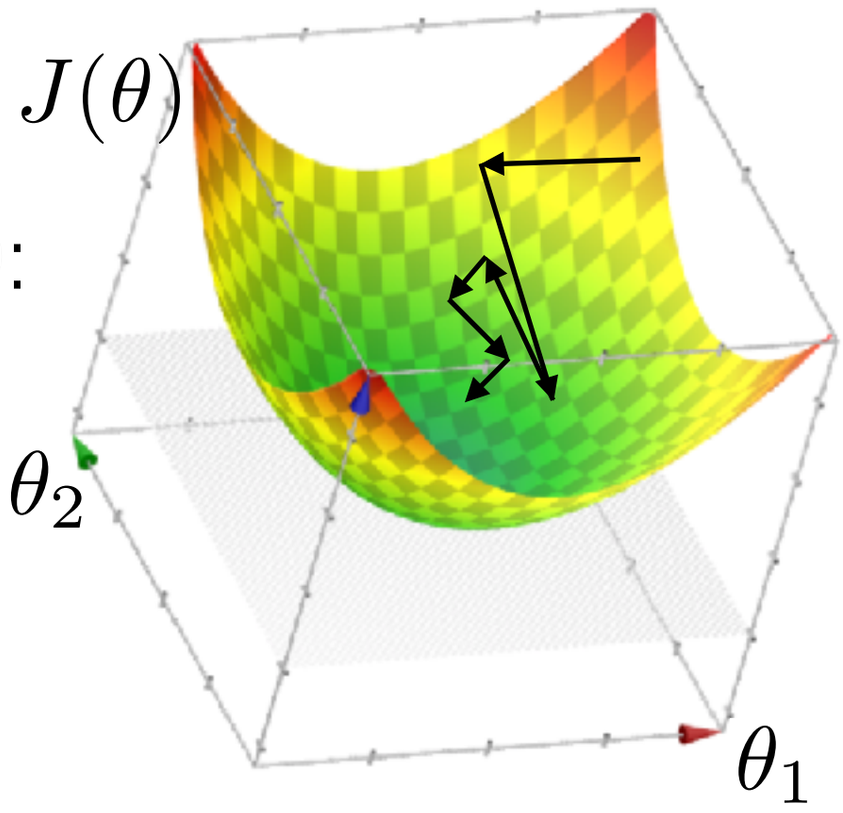
Learning the parameters

- Objective: $J(W, W_0) = \frac{1}{n_{\text{data}}} \sum_{i=1}^{n_{\text{data}}} L(h(x^{(i)}; W, W_0), y^{(i)})$

- GD:



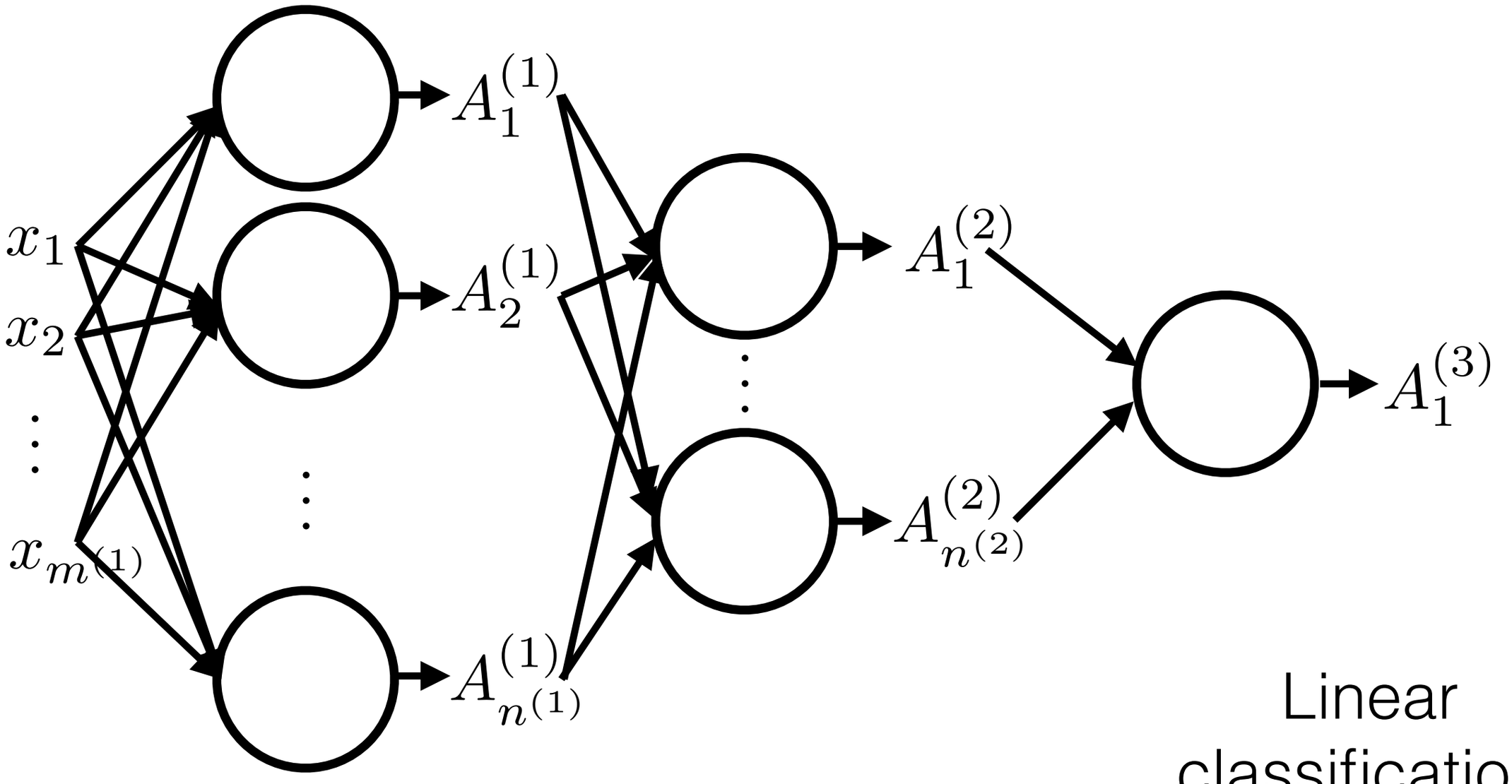
- SGD:



- **Theorem**: (Roughly) if the objective is nice and convex, GD and SGD perform well
- **Big challenge**: the NN objective is a (very) non-convex function of the parameters (except in e.g. 1 layer)
- Huge bag of tricks to optimize / regularize

More layers!

- Why stop at 2 layers?



- Just one layer: linear classification/regression with default features