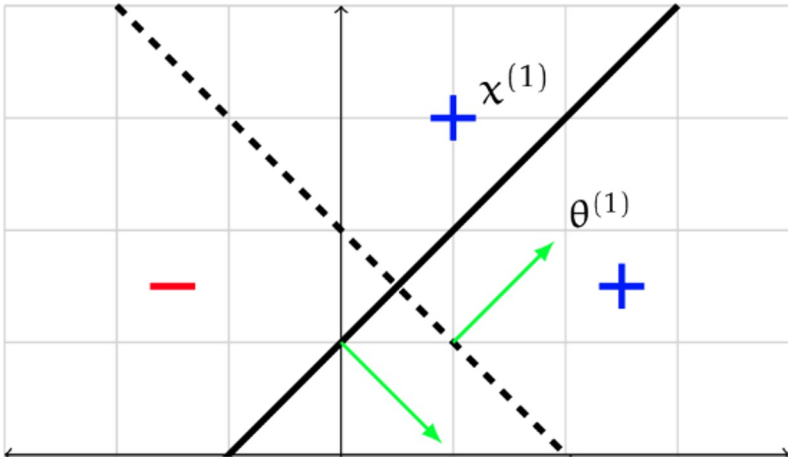
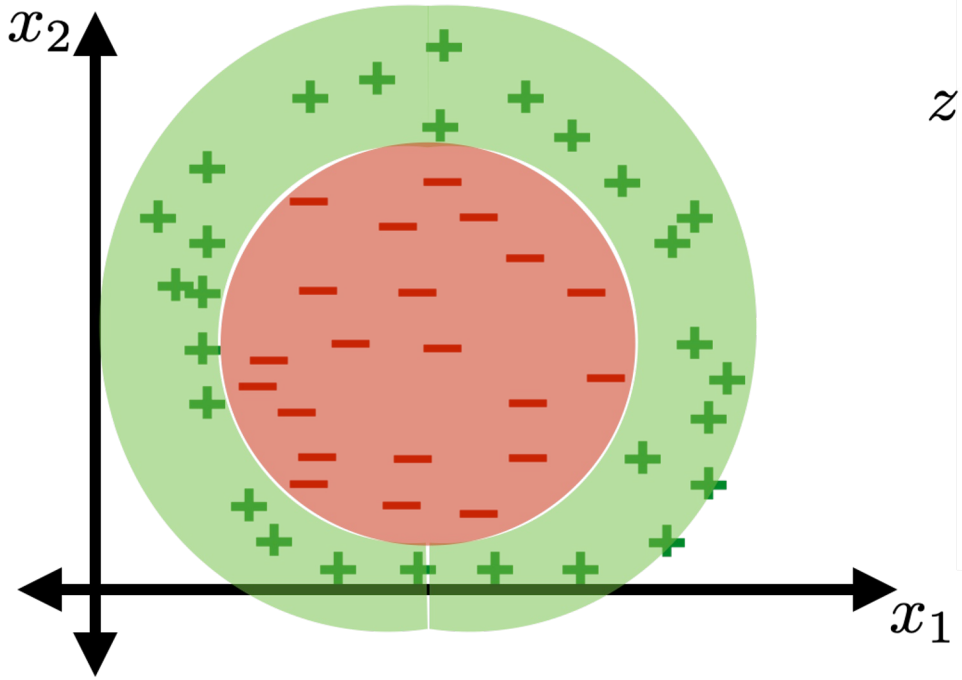
Spring 2023!

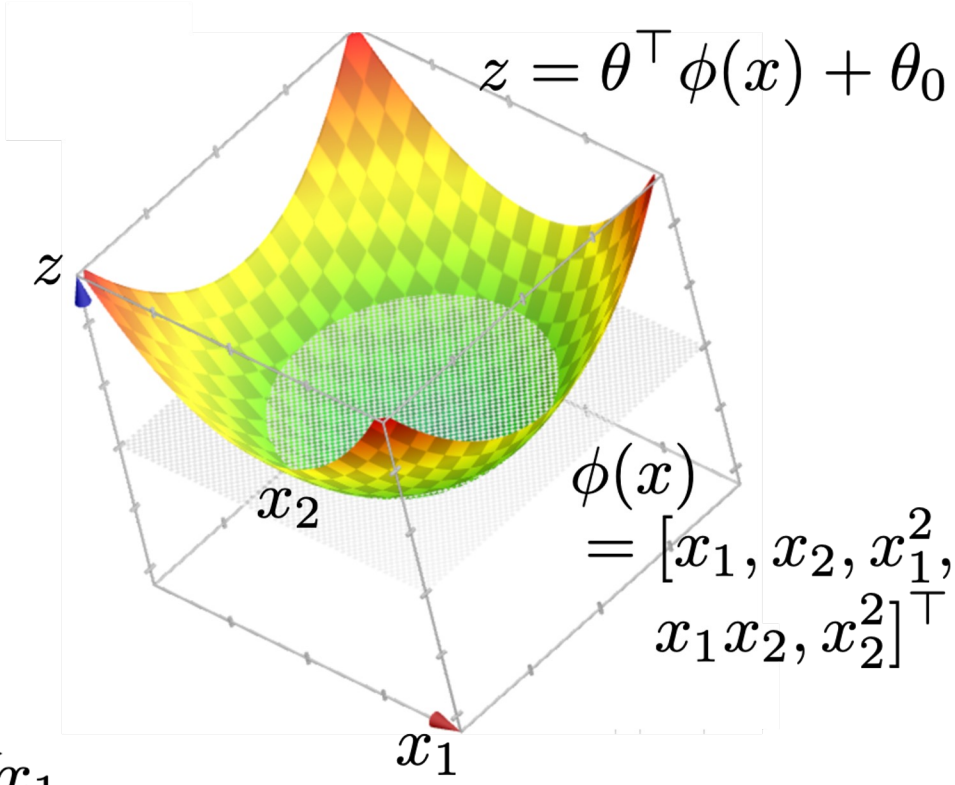
Introduction to Machine Learning



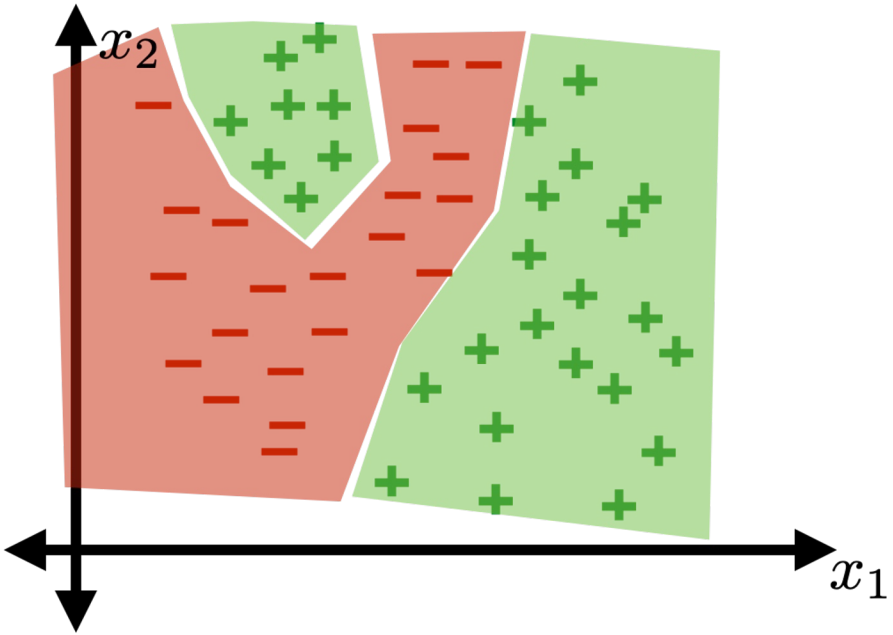
Neural Networks I

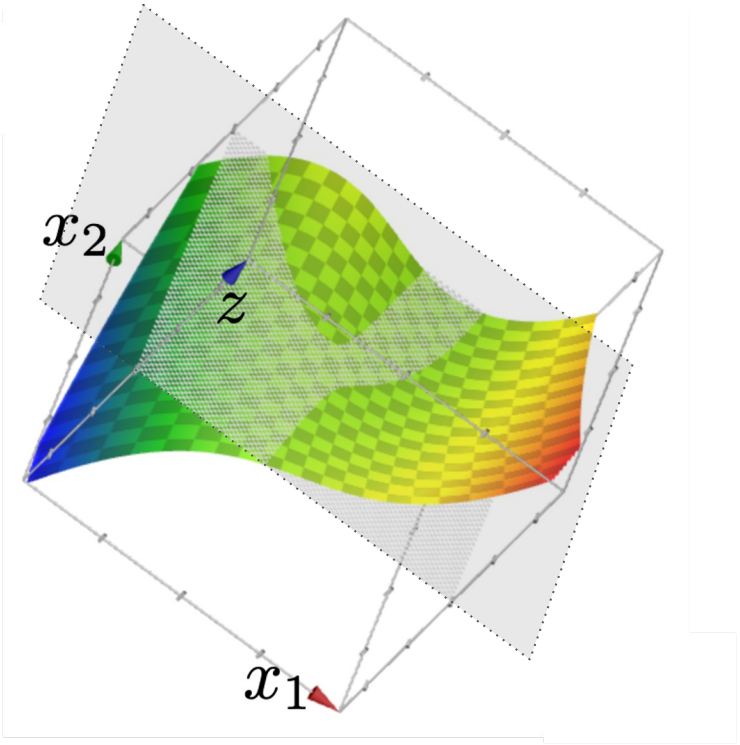
Review: Non-linear classifiers





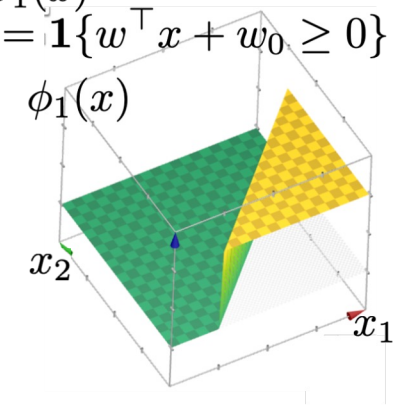
Review: Non-linear classifiers





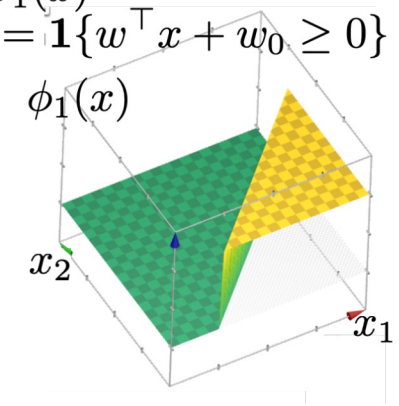
New features: step functions

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

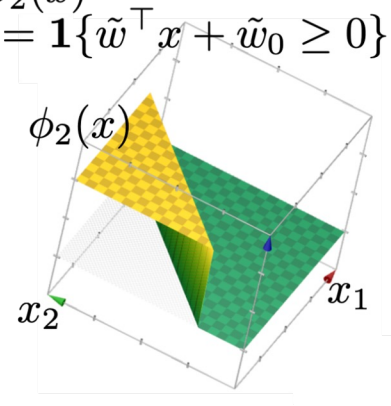


New features: step functions

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

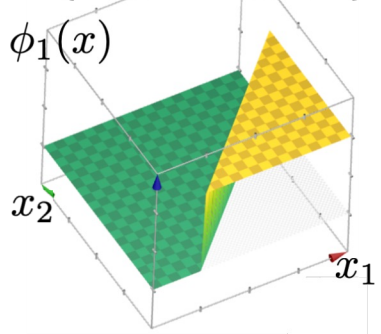


$$\phi_2(x) = \mathbf{1}\{\tilde{w}^\top x + \tilde{w}_0 \geq 0\}$$

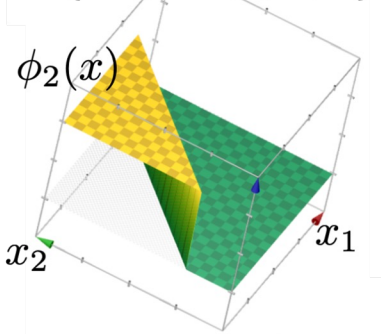


New features: step functions

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



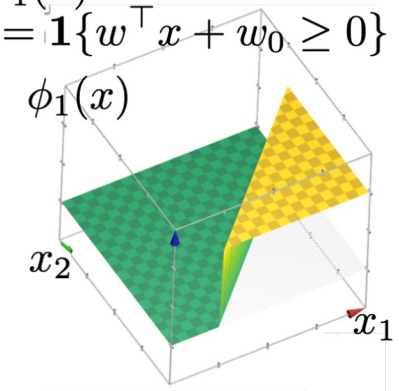
$\phi_2(x)$
$= \mathbf{1}\{\tilde{w}^\top x + \tilde{w}_0 \geq 0\}$



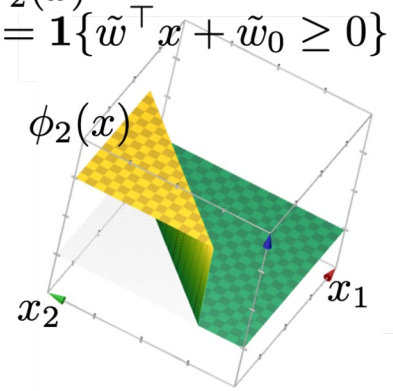
$$z = \theta^\top \phi(x) + \theta_0$$
$$= \theta_1 \phi_1(x) + \theta_2 \phi_2(x) + \theta_0$$
$$= 1 \cdot \phi_1(x) + 1 \cdot \phi_2(x) + (-0.5)$$

New features: step functions

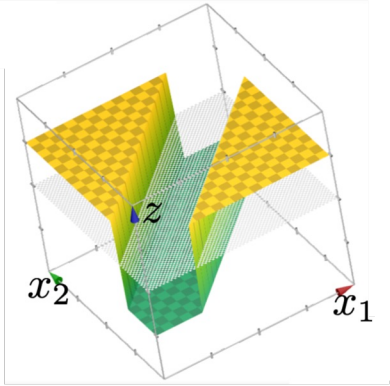
$$\phi_1(x) = \mathbf{1}\{w^\top x + w_0 \geq 0\}$$



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

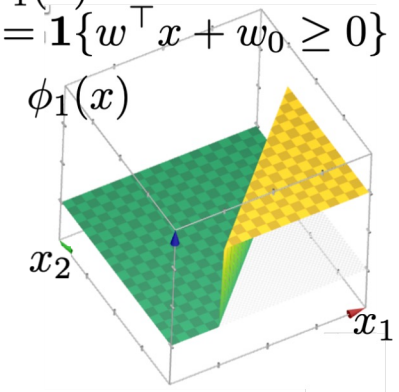


$$z = \theta^\top \phi(x) + \theta_0$$
$$= \theta_1 \phi_1(x) + \theta_2 \phi_2(x) + \theta_0$$
$$= 1 \cdot \phi_1(x) + 1 \cdot \phi_2(x) + (-0.5)$$

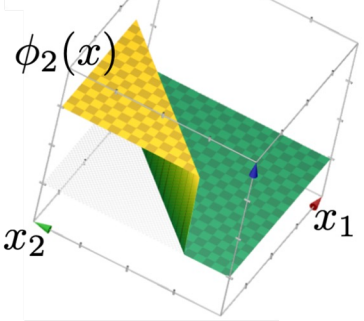


New features: step functions

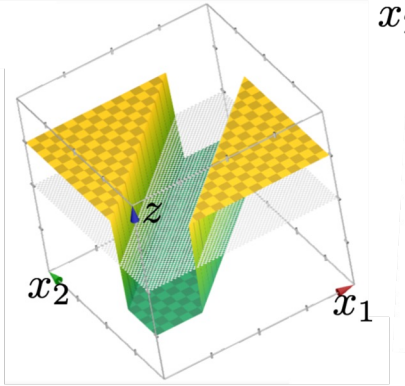
$$\phi_1(x) = \mathbf{1}\{w^\top x + w_0 \geq 0\}$$

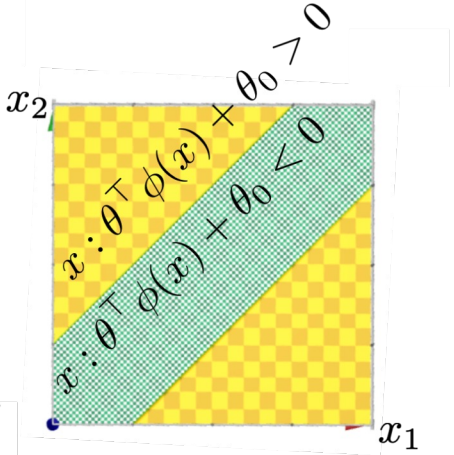


$$\phi_2(x) = \mathbf{1}\{\tilde{w}^\top x + \tilde{w}_0 \geq 0\}$$



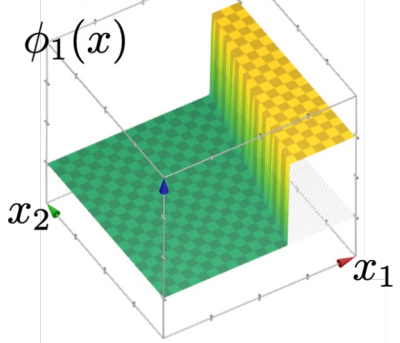
$$z = \theta^\top \phi(x) + \theta_0$$
$$= \theta_1 \phi_1(x) + \theta_2 \phi_2(x) + \theta_0$$
$$= 1 \cdot \phi_1(x) + 1 \cdot \phi_2(x) + (-0.5)$$



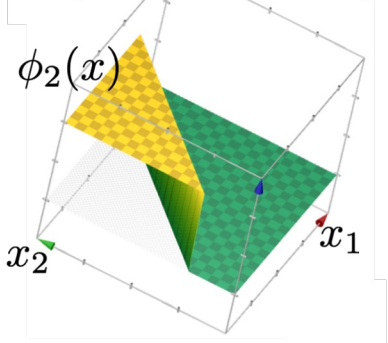


New features: step functions

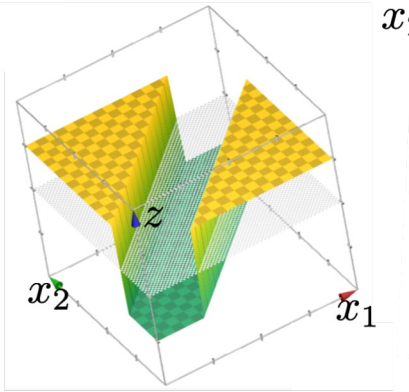
$\phi_1(x) = \mathbf{1}\{w^\top x + w_0 \geq 0\}$

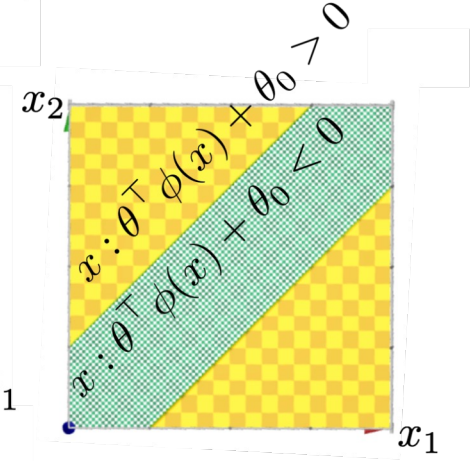


$\phi_2(x) = \mathbf{1}\{\tilde{w}^\top x + \tilde{w}_0 \geq 0\}$



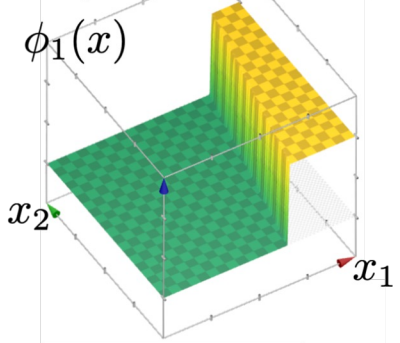
$$z = \theta^\top \phi(x) + \theta_0$$
$$= \theta_1 \phi_1(x) + \theta_2 \phi_2(x) + \theta_0$$
$$= 1 \cdot \phi_1(x) + 1 \cdot \phi_2(x) + (-0.5)$$



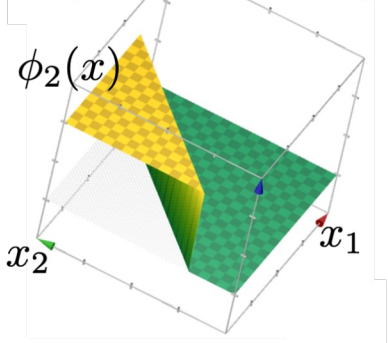


New features: step functions

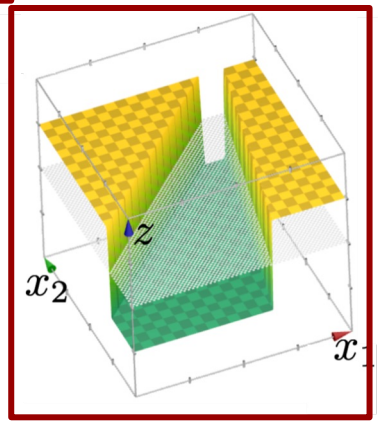
$$\phi_1(x) = \mathbf{1}\{w^\top x + w_0 \geq 0\}$$

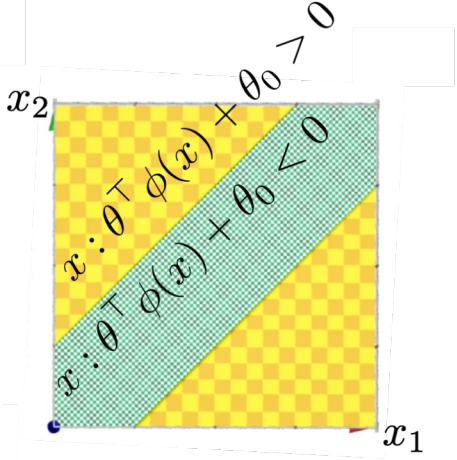


$$\phi_2(x) = \mathbf{1}\{\tilde{w}^\top x + \tilde{w}_0 \geq 0\}$$



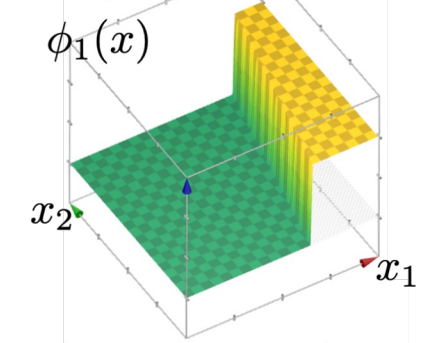
$$z = \theta^\top \phi(x) + \theta_0$$
$$= \theta_1 \phi_1(x) + \theta_2 \phi_2(x) + \theta_0$$
$$= 1 \cdot \phi_1(x) + 1 \cdot \phi_2(x) + (-0.5)$$



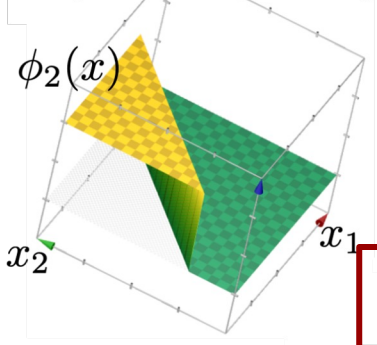


New features: step functions

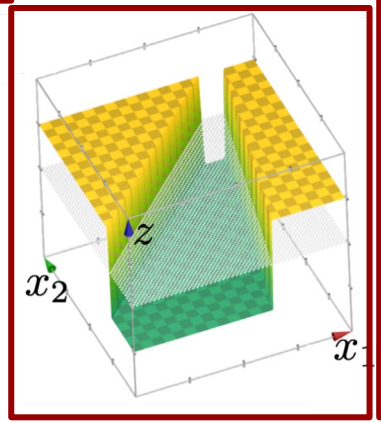
$$\phi_1(x) = \mathbf{1}\{w^\top x + w_0 \geq 0\}$$

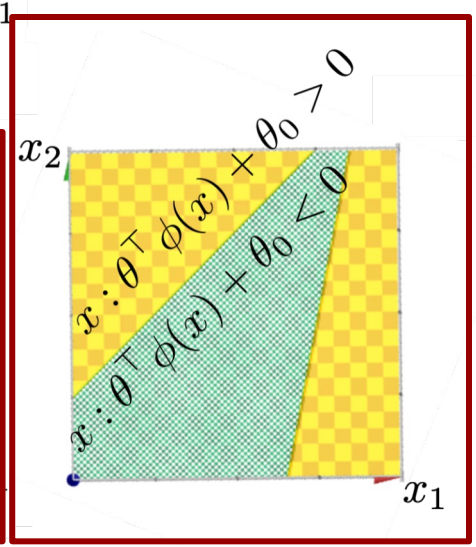


$$\phi_2(x) = \mathbf{1}\{\tilde{w}^\top x + \tilde{w}_0 \geq 0\}$$



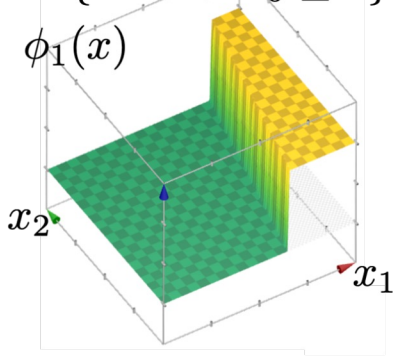
$$z = \theta^\top \phi(x) + \theta_0$$
$$= \theta_1 \phi_1(x) + \theta_2 \phi_2(x) + \theta_0$$
$$= 1 \cdot \phi_1(x) + 1 \cdot \phi_2(x) + (-0.5)$$



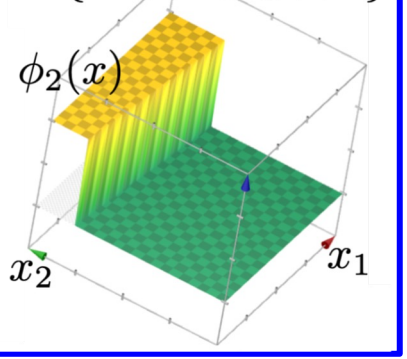


New features: step functions

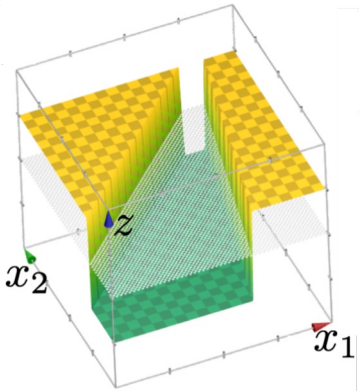
$\phi_1(x)$
$= \mathbf{1}\{w^\top x + w_0 \geq 0\}$

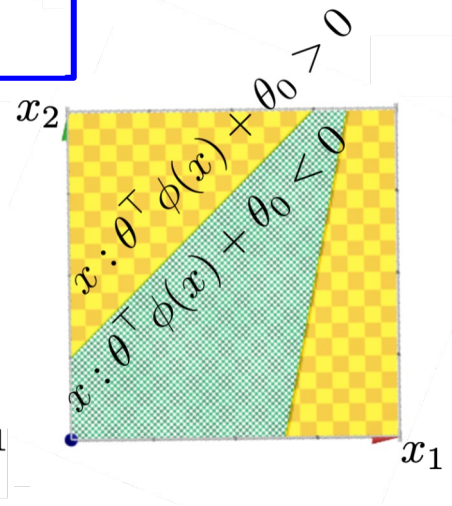


$\phi_2(x)$
$= \mathbf{1}\{\tilde{w}^\top x + \tilde{w}_0 \geq 0\}$



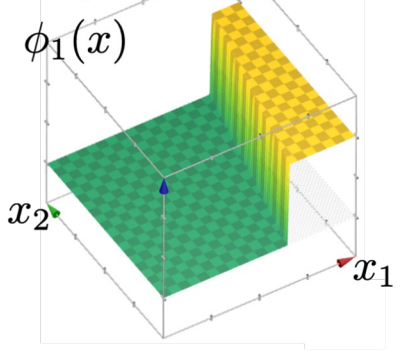
$$z = \theta^\top \phi(x) + \theta_0$$
$$= \theta_1 \phi_1(x) + \theta_2 \phi_2(x) + \theta_0$$
$$= 1 \cdot \phi_1(x) + 1 \cdot \phi_2(x) + (-0.5)$$



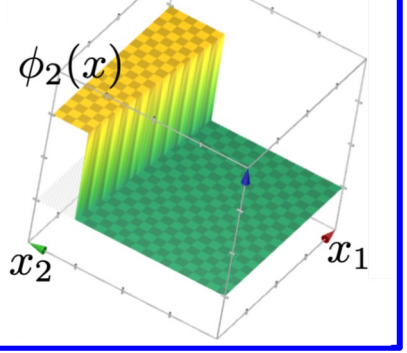


New features: step functions

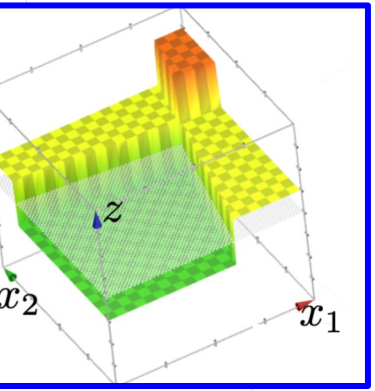
$$\phi_1(x) = \mathbf{1}\{w^\top x + w_0 \geq 0\}$$

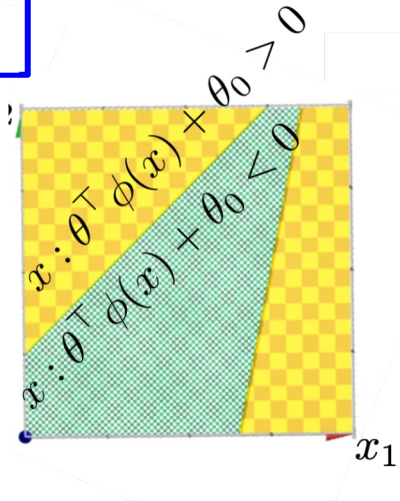


$$\phi_2(x) = \mathbf{1}\{\tilde{w}^\top x + \tilde{w}_0 \geq 0\}$$



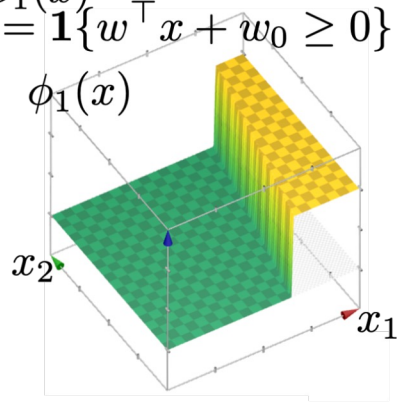
$$z = \theta^\top \phi(x) + \theta_0$$
$$= \theta_1 \phi_1(x) + \theta_2 \phi_2(x) + \theta_0$$
$$= 1 \cdot \phi_1(x) + 1 \cdot \phi_2(x) + (-0.5)$$



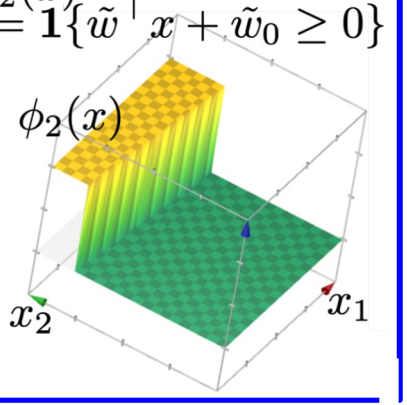


New features: step functions

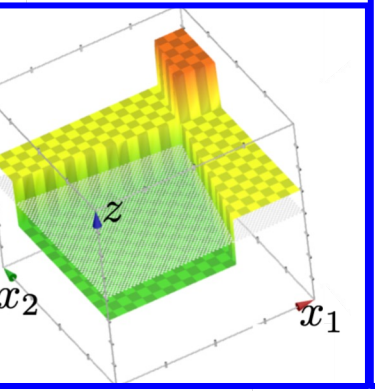
$$\phi_1(x) = \mathbf{1}\{w^\top x + w_0 \geq 0\}$$

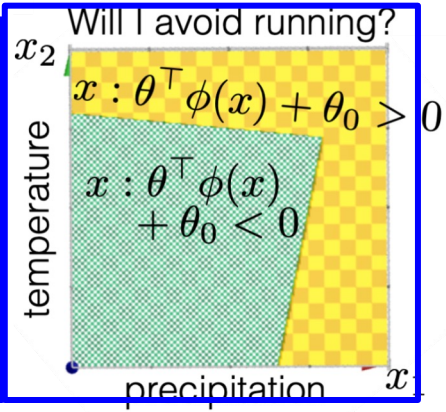


$$\phi_2(x) = \mathbf{1}\{\tilde{w}^\top x + \tilde{w}_0 \geq 0\}$$



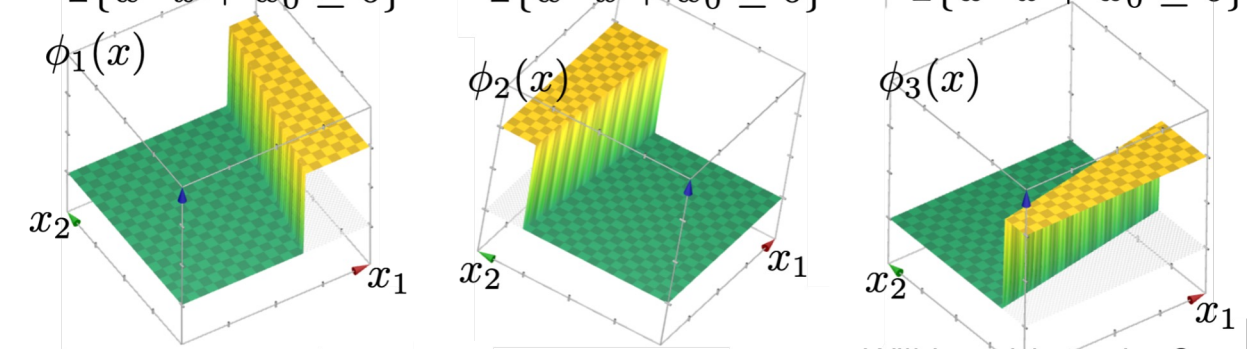
$$z = \theta^\top \phi(x) + \theta_0$$
$$= \theta_1 \phi_1(x) + \theta_2 \phi_2(x) + \theta_0$$
$$= 1 \cdot \phi_1(x) + 1 \cdot \phi_2(x) + (-0.5)$$





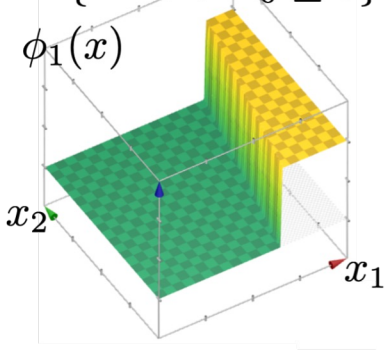
New features: step functions

$\phi_1(x) = \mathbf{1}\{w^\top x + w_0 \geq 0\}$ $\phi_2(x) = \mathbf{1}\{\tilde{w}^\top x + \tilde{w}_0 \geq 0\}$ $\phi_3(x) = \mathbf{1}\{\tilde{\tilde{w}}^\top x + \tilde{\tilde{w}}_0 \geq 0\}$

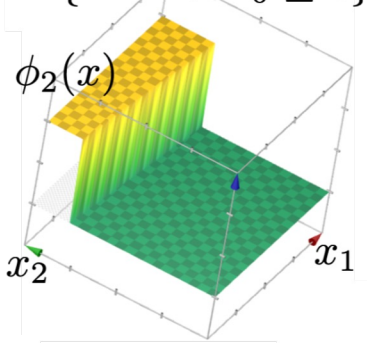


New features: step functions

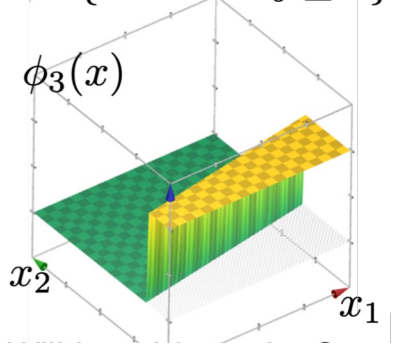
$\phi_1(x) = \mathbf{1}\{w^\top x + w_0 \geq 0\}$



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



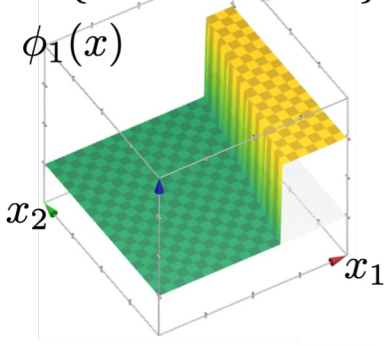
$\phi_3(x) = \mathbf{1}\{\tilde{\tilde{w}}^\top x + \tilde{\tilde{w}}_0 \geq 0\}$



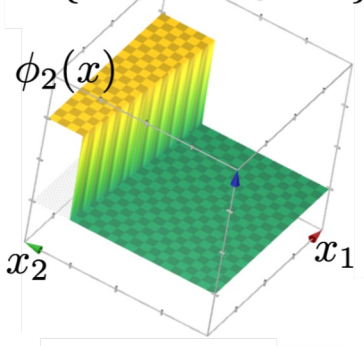
$$z = \theta^\top \phi(x) + \theta_0$$
$$= \theta_1\phi_1(x) + \theta_2\phi_2(x) + \theta_3\phi_3(x) + \theta_0$$
$$= 1 \cdot \phi_1(x) + 1 \cdot \phi_2(x) + 1 \cdot \phi_3(x) + (-0.5)$$

New features: step functions

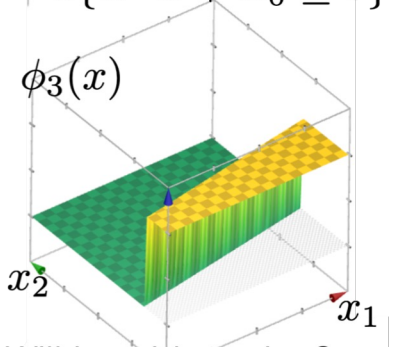
$$\phi_1(x) = \mathbf{1}\{w^\top x + w_0 \geq 0\}$$



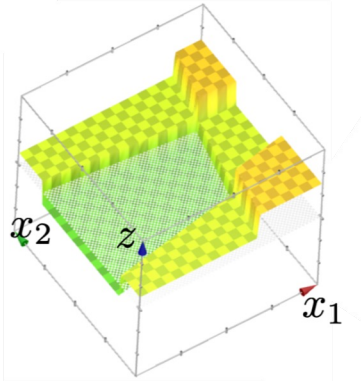
$$\phi_2(x) = \mathbf{1}\{\tilde{w}^\top x + \tilde{w}_0 \geq 0\}$$



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

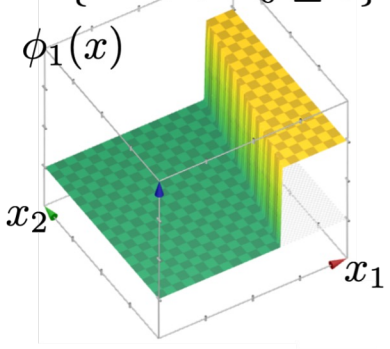


$$z = \theta^\top \phi(x) + \theta_0$$
$$= \theta_1\phi_1(x) + \theta_2\phi_2(x) + \theta_3\phi_3(x) + \theta_0$$
$$= 1 \cdot \phi_1(x) + 1 \cdot \phi_2(x) + 1 \cdot \phi_3(x) + (-0.5)$$

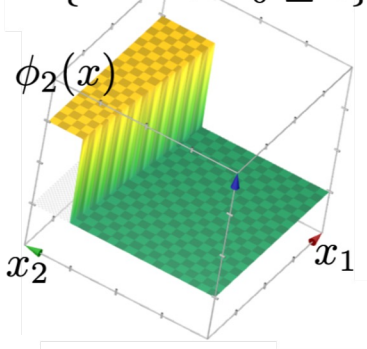


New features: step functions

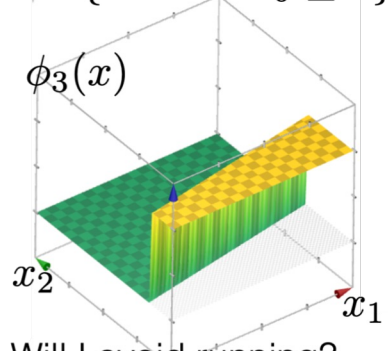
$\phi_1(x) = \mathbf{1}\{w^\top x + w_0 \geq 0\}$



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

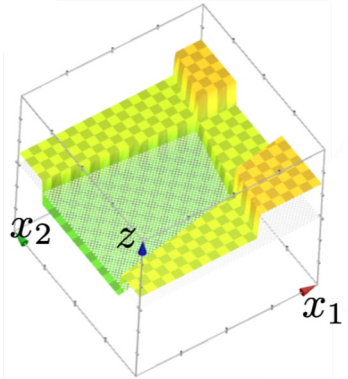


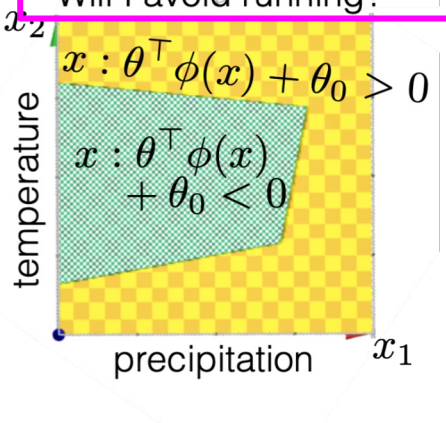
$\phi_3(x) = \mathbf{1}\{\tilde{\tilde{w}}^\top x + \tilde{\tilde{w}}_0 \geq 0\}$



Will I avoid running?

$$z = \theta^\top \phi(x) + \theta_0$$
$$= \theta_1 \phi_1(x) + \theta_2 \phi_2(x) + \theta_3 \phi_3(x) + \theta_0$$
$$= 1 \cdot \phi_1(x) + 1 \cdot \phi_2(x) + 1 \cdot \phi_3(x) + (-0.5)$$



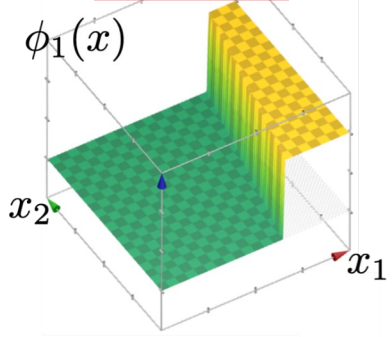


New features: step functions

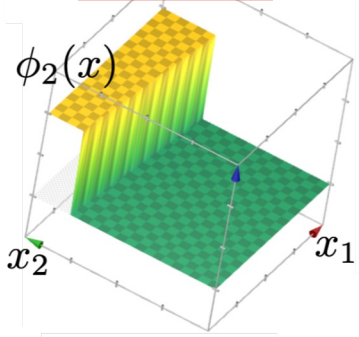
These weights decide the how the input is transformed

These weights decide how the new features are combined

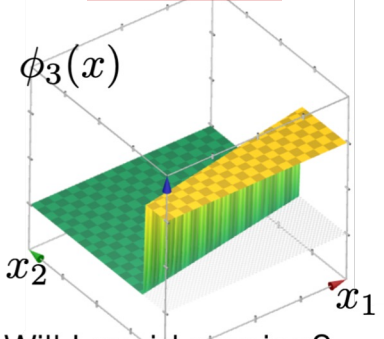
$$\phi_1(x) = \mathbf{1}\{w^\top x + w_0 \geq 0\}$$



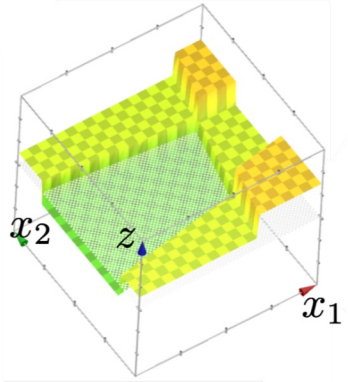
$$\phi_2(x) = \mathbf{1}\{\tilde{w}^\top x + \tilde{w}_0 \geq 0\}$$



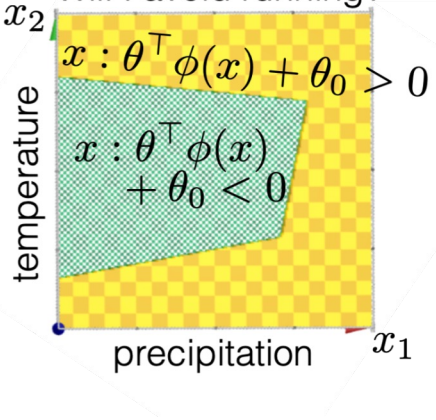
$$\phi_3(x) = \mathbf{1}\{\tilde{\tilde{w}}^\top x + \tilde{\tilde{w}}_0 \geq 0\}$$



$$z = \theta^\top \phi(x) + \theta_0$$
$$= \theta_1\phi_1(x) + \theta_2\phi_2(x) + \theta_3\phi_3(x) + \theta_0$$
$$= 1 \cdot \phi_1(x) + 1 \cdot \phi_2(x) + 1 \cdot \phi_3(x) + (-0.5)$$

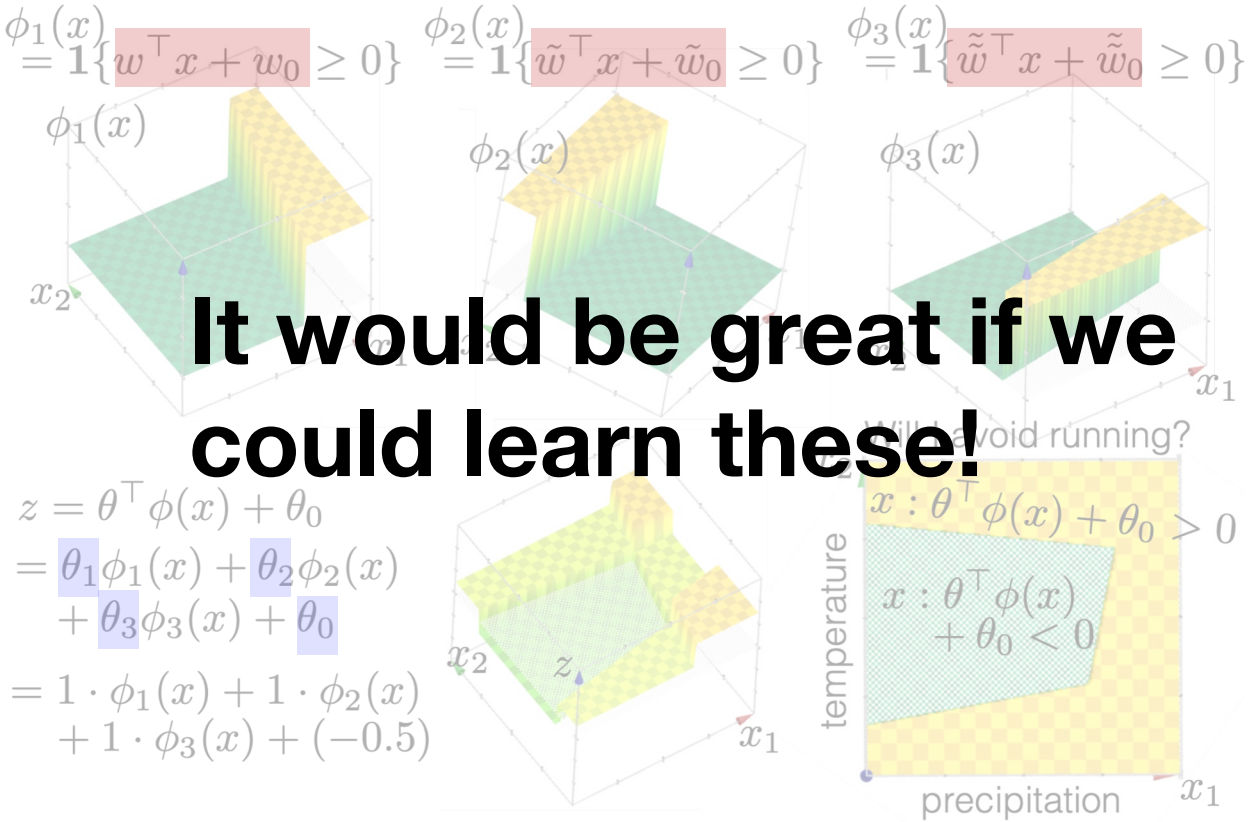


Will I avoid running?



New features: step functions

These weights decide the how the input is transformed



It would be great if we could learn these!

These weights decide how the new features are combined

Neural Networks

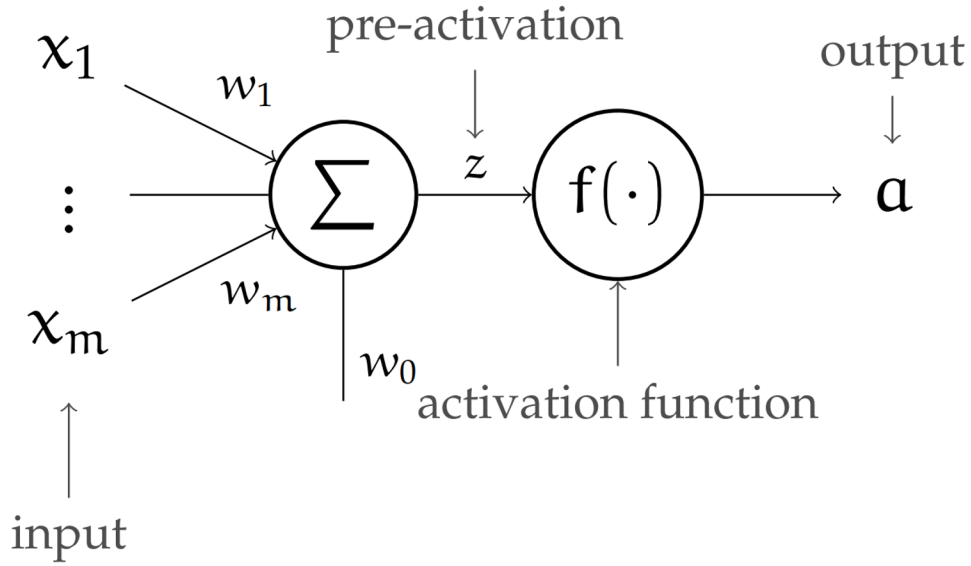
- A rich hypothesis class of parameterized functions with multiple layers of “hidden” units that can learn to represent complex features of the input.

- Trained with gradient descent (“backpropagation”).

- Originally inspired by the brain, but connection is tentative.

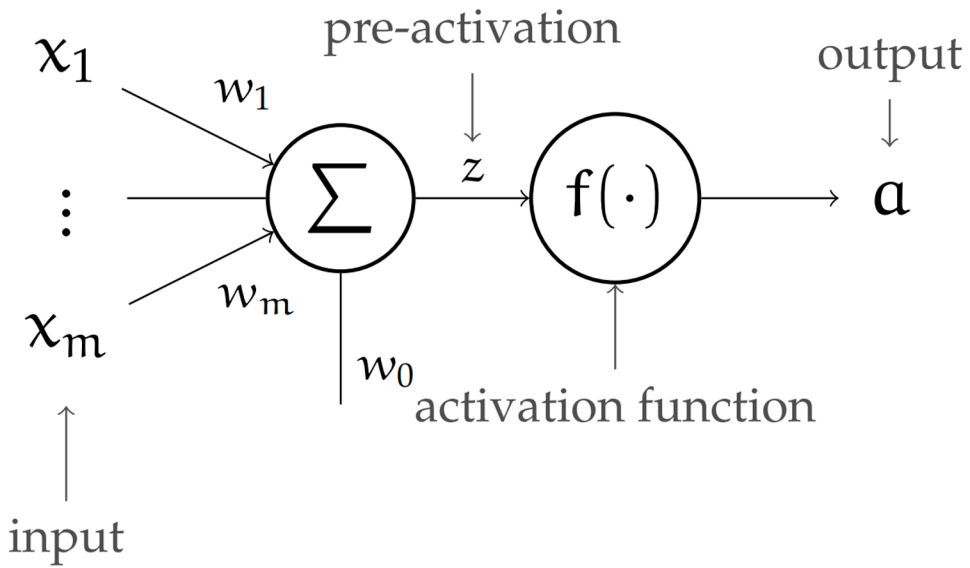
Neural Networks

$$a = f(z) = f\left(\left(\sum_{j=1}^{m} x_j w_j\right) + w_0\right) = f(w^T x + w_0)$$



Neural Networks

$$a = f(z) = f\left(\left(\sum_{j=1}^{m} x_j w_j\right) + w_0\right) = f(w^T x + w_0)$$



$$f(x) = \sigma(x)$$

$$J(w, w_0) = \sum_i L\left(NN(x^{(i)}; w, w_0), y^{(i)}\right)$$

$$L(g^{(i)}, y^{(i)}) = y^{(i)} \log g^{(i)} + (1 - y^{(i)}) \log(1 - g^{(i)})$$

Neural Networks

$$a = f(z) = f\left(\left(\sum_{j=1}^{m} x_j w_j\right) + w_0\right) = f(w^T x + w_0)$$

This is just logistic regression!!

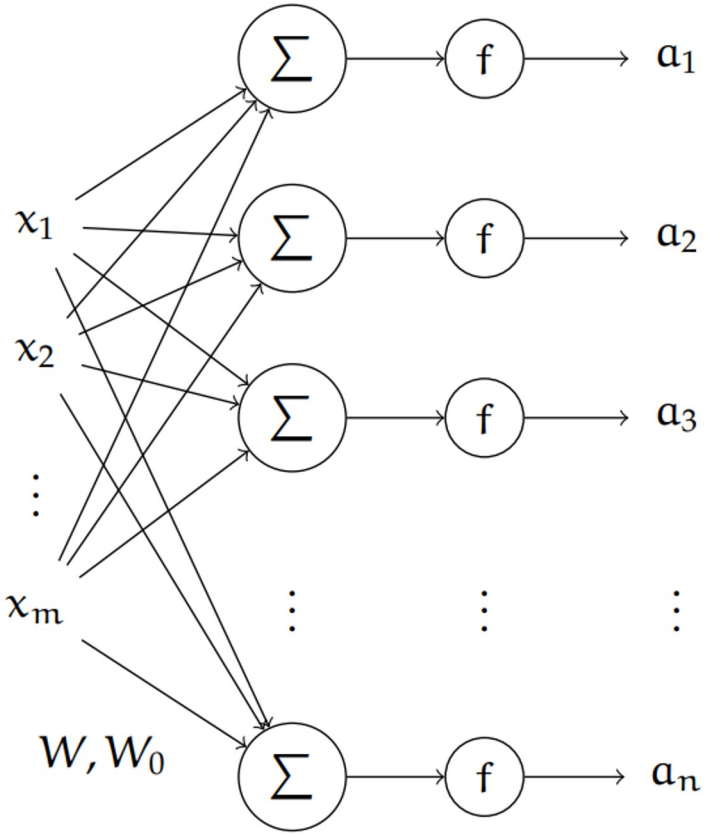
Neural Network with 0 hidden layers

$$J(w, w_0)$$

$$= \sum_i L\left(NN(x^{(i)}; w, w_0), y^{(i)}\right)$$

$$f(x) = \sigma(x)$$

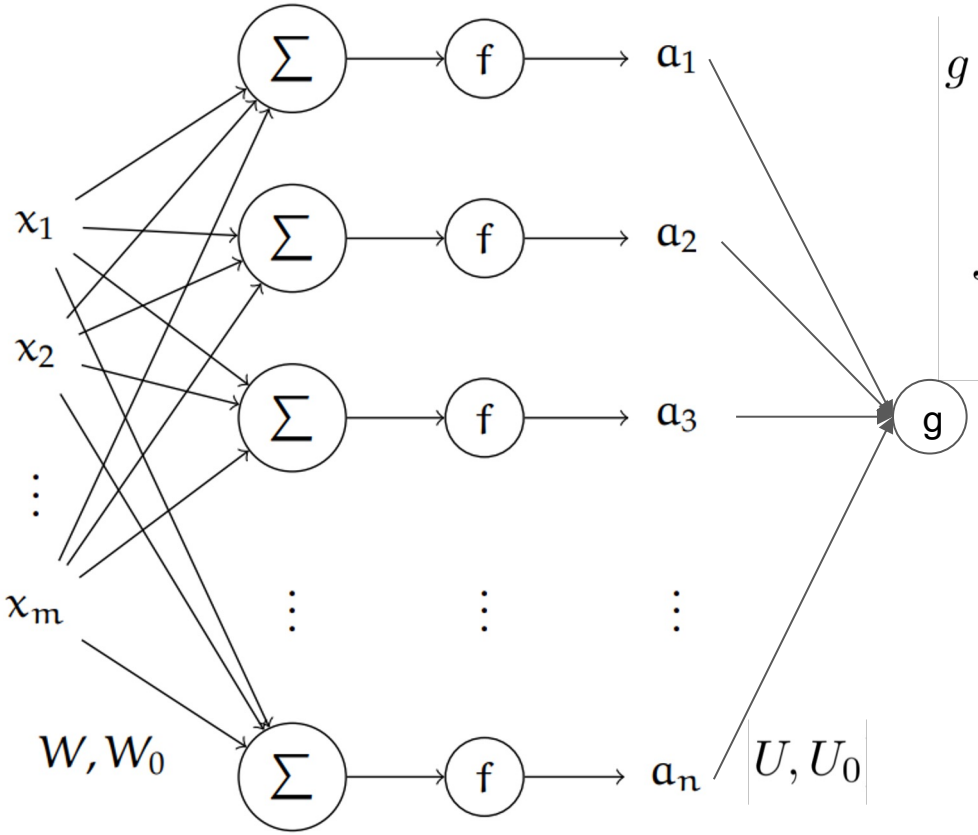
Neural Networks



$$A = f(Z) = f(W^T X + W_0)$$

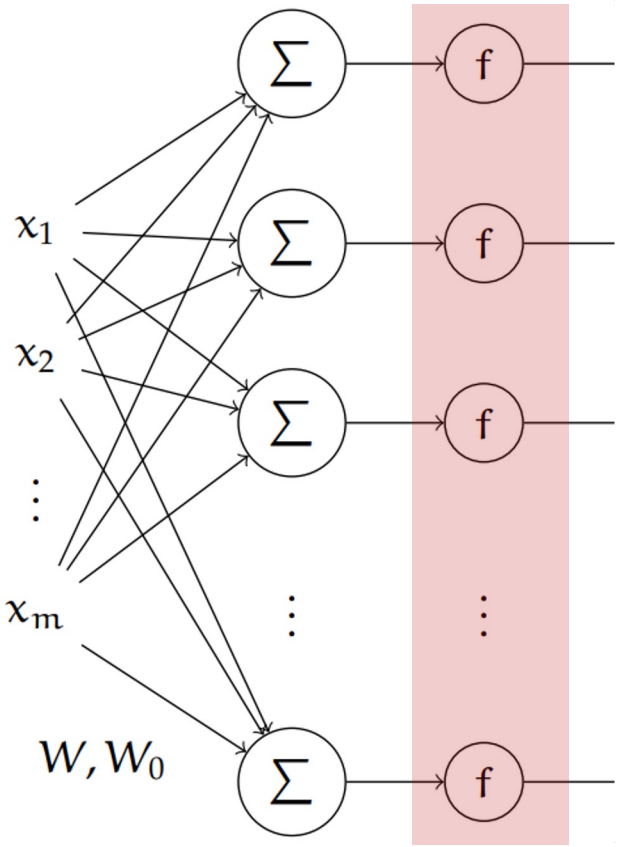
- W is an $m \times n$ matrix,
- W_0 is an $n \times 1$ column vector,
- X, the input, is an $m \times 1$ column vector,
- $Z = W^T X + W_0$, the *pre-activation*, is an $n \times 1$ column vector,
- A, the *activation*, is an $n \times 1$ column vector,

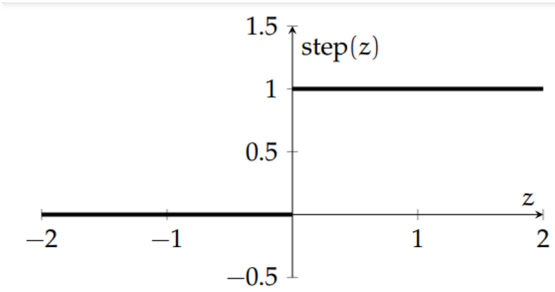
Two Layer Neural Network for Regression

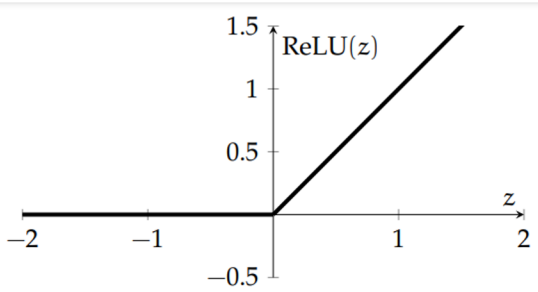


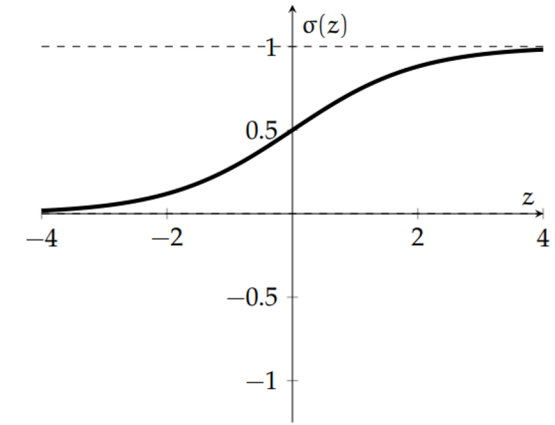
$$g = U^\top A + U_0$$
$$= U^\top f(W^\top x + W_0) + U_0$$
$$J(W, W_0, U, U_0) = \frac{1}{n}\sum_{i=1}^{n}(y^{(i)} - g^{(i)})^2$$

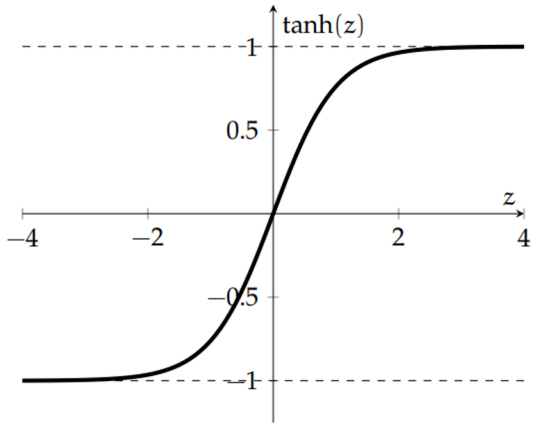
Neural Network Activation Function



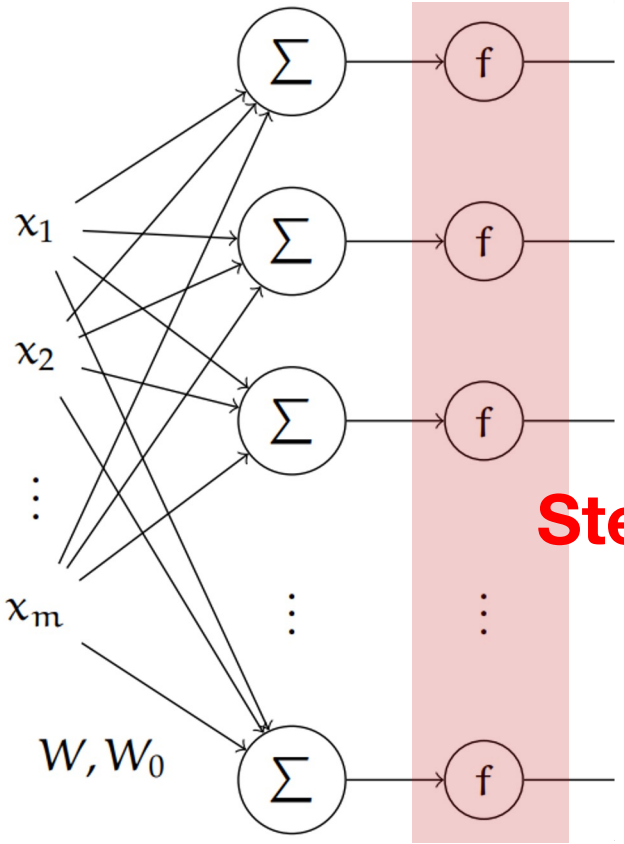


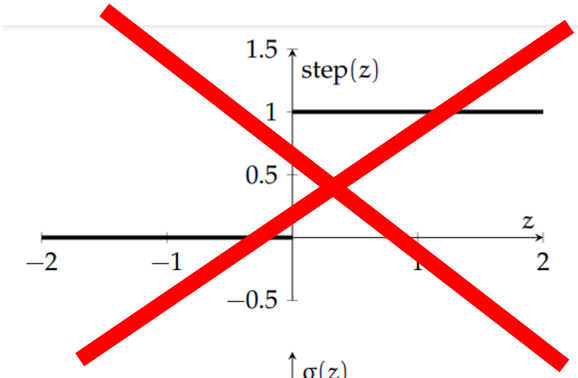


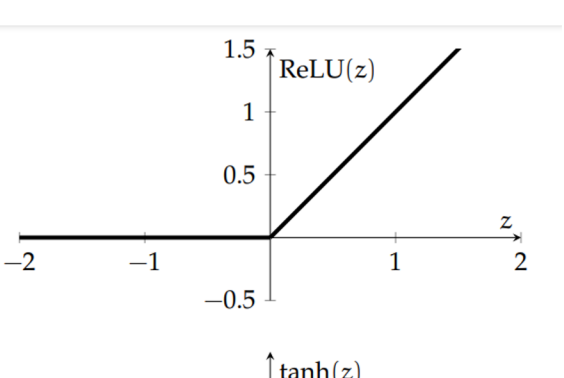


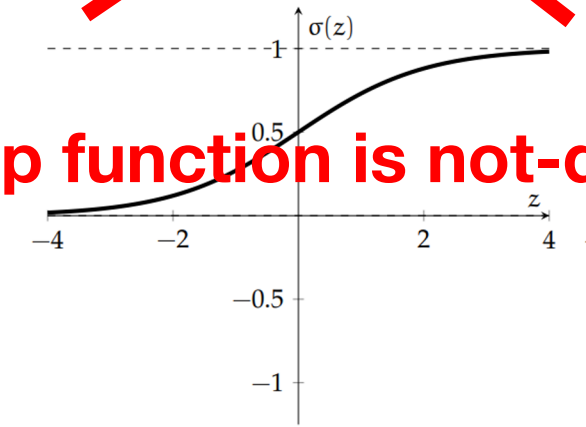


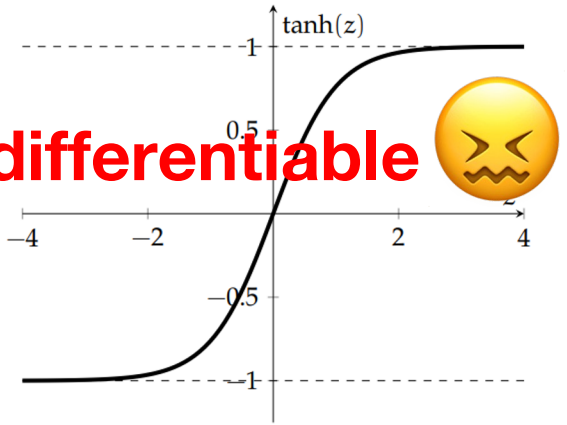
Neural Network Activation Function





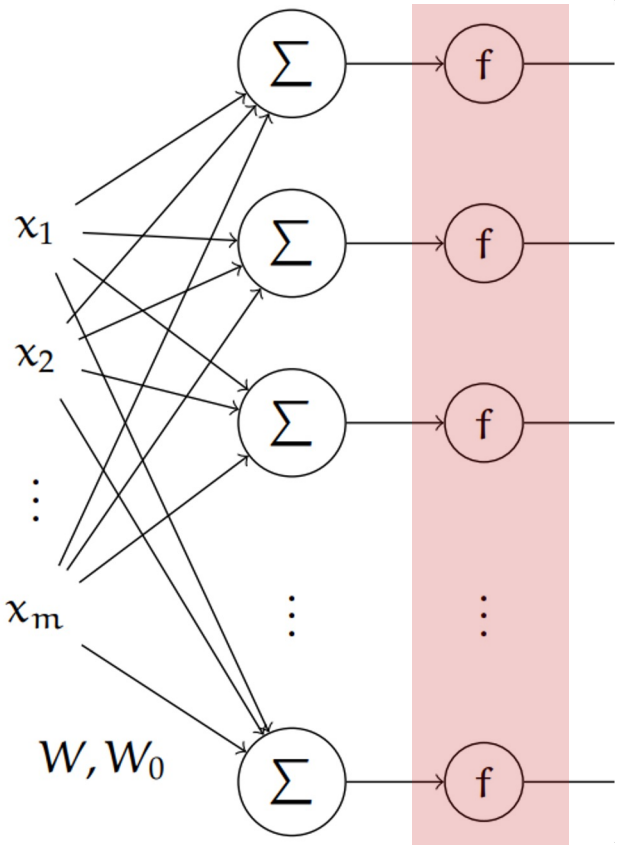


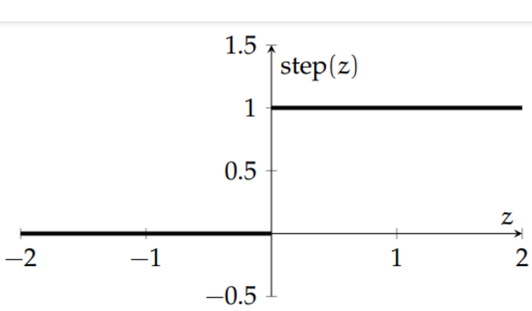


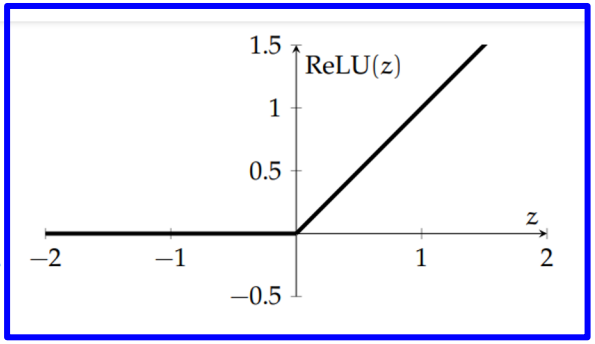


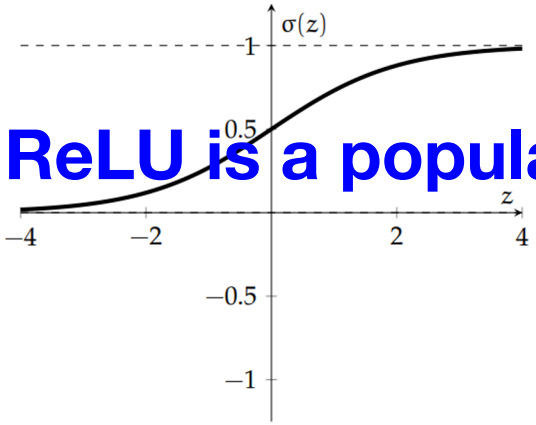
Step function is not-differentiable 😖

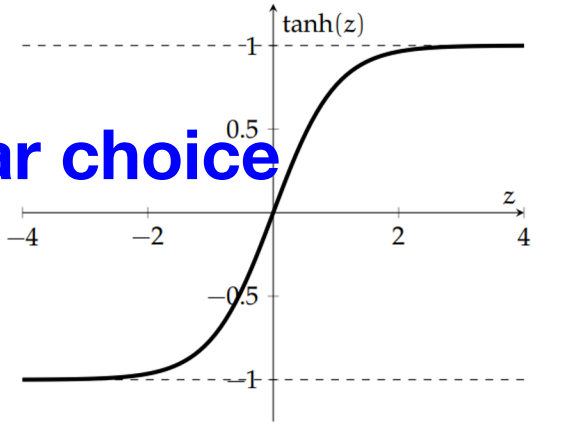
Neural Network Activation Function





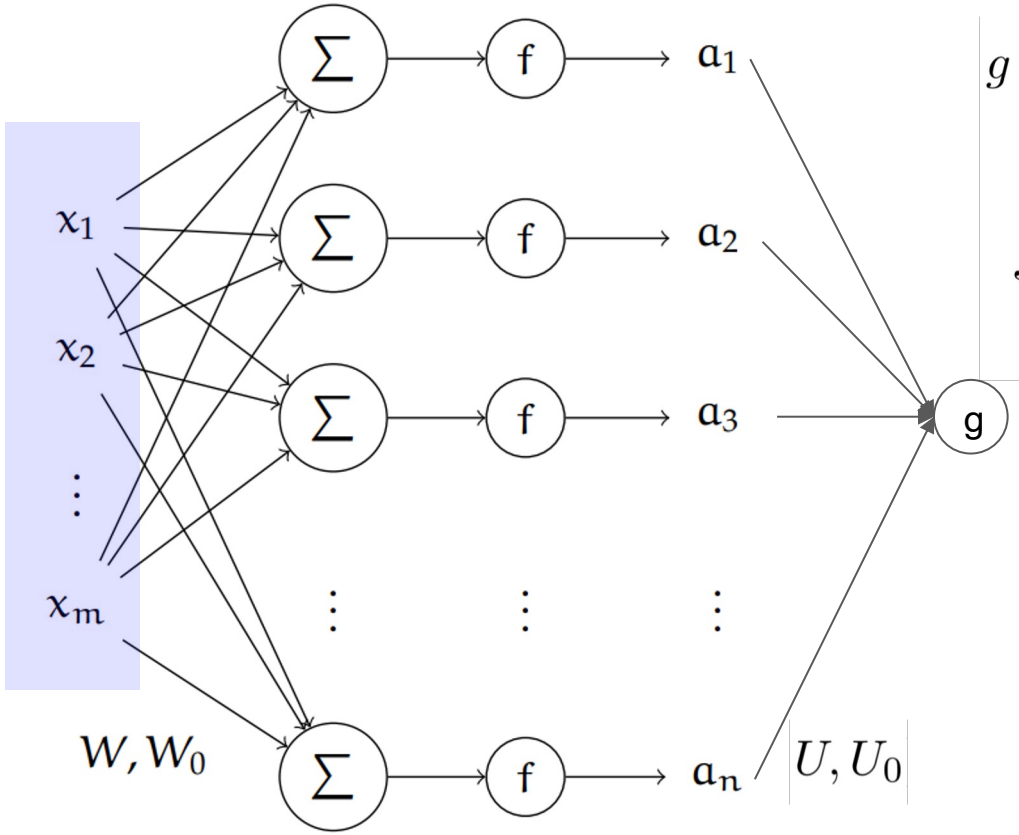






ReLU is a popular choice

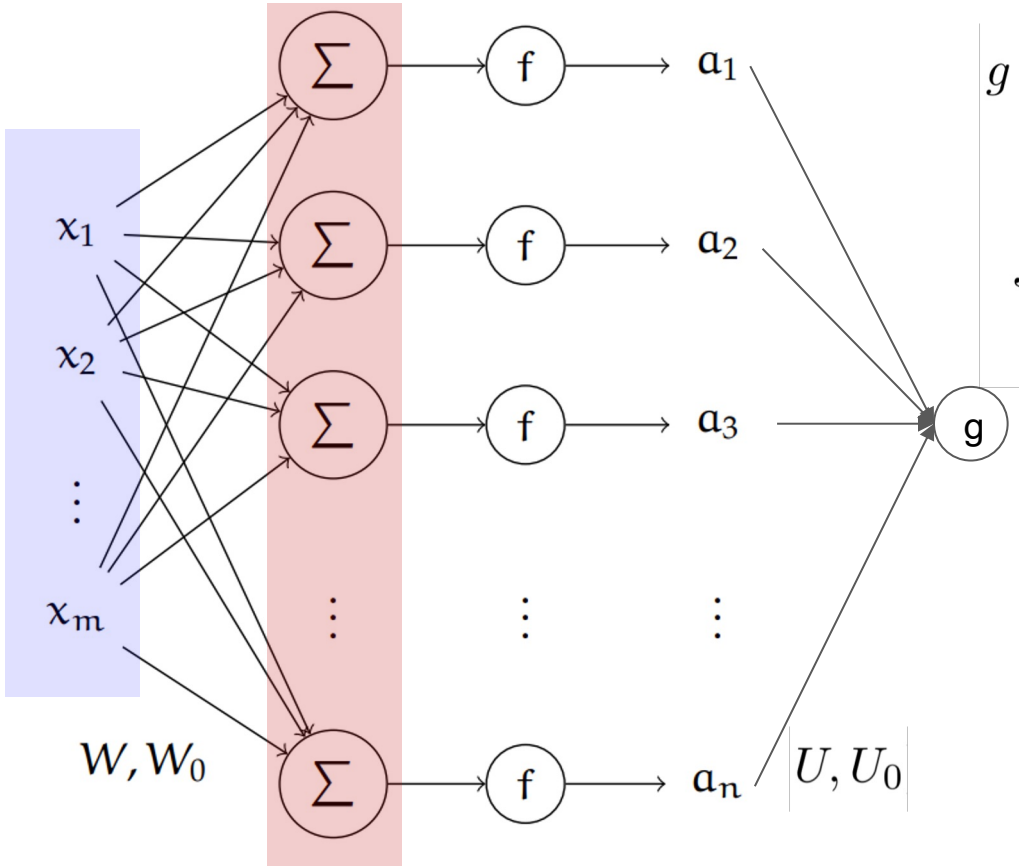
Two Layer Neural Network for Regression



$$g = U^\top A + U_0$$
$$= U^\top f(W^\top x + W_0) + U_0$$
$$J(W, W_0, U, U_0) = \frac{1}{n}\sum_{i=1}^{n}(y^{(i)} - g^{(i)})^2$$

Input

Two Layer Neural Network for Regression

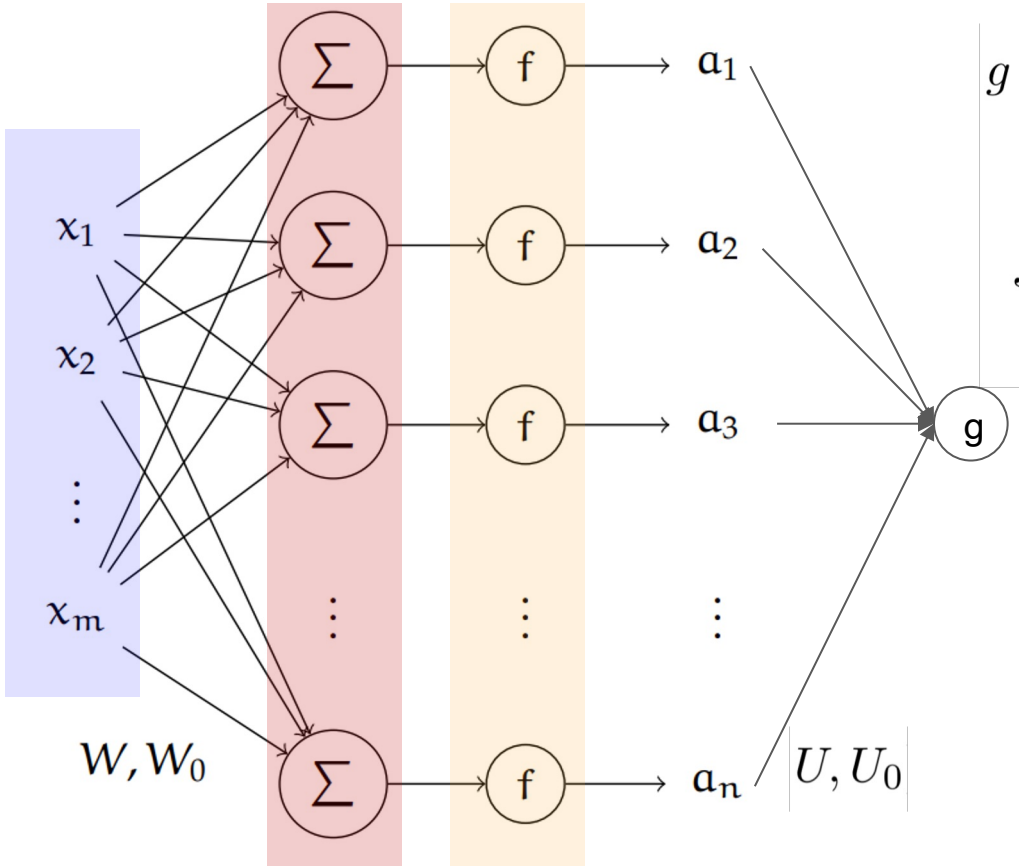


$$g = U^\top A + U_0$$
$$= U^\top f(W^\top x + W_0) + U_0$$
$$J(W, W_0, U, U_0) = \frac{1}{n}\sum_{i=1}^{n}(y^{(i)} - g^{(i)})^2$$

Input

Preactivation

Two Layer Neural Network for Regression



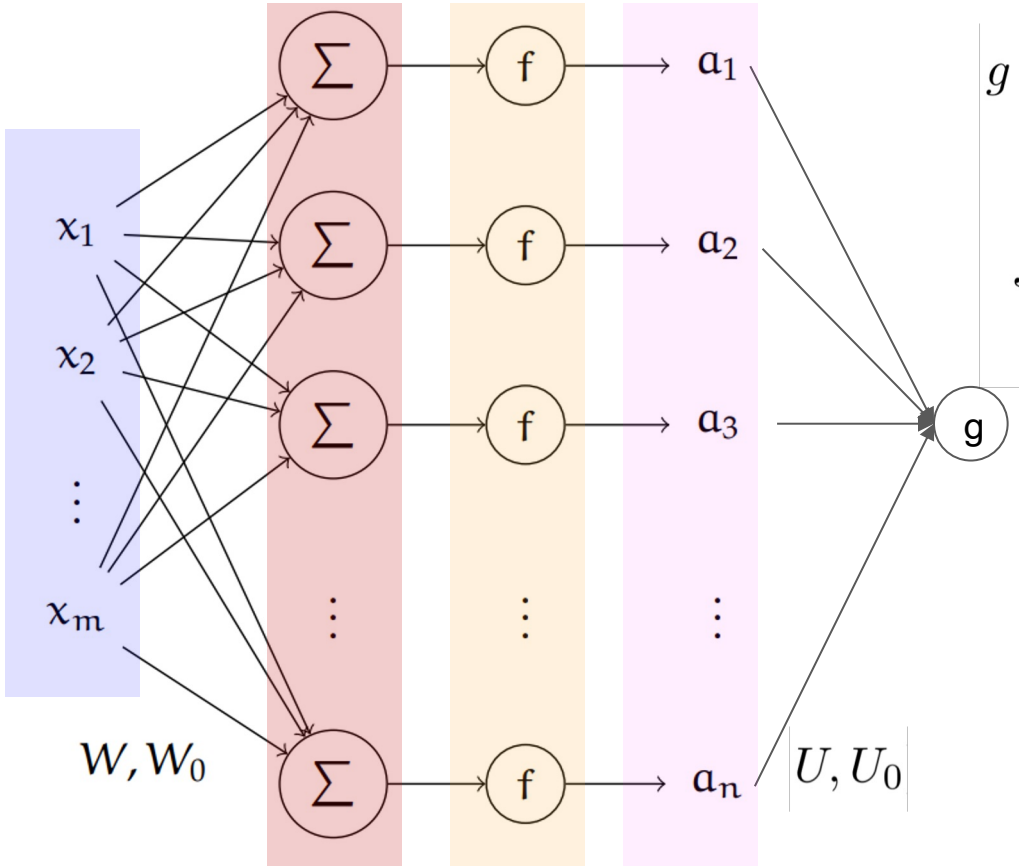
$$g = U^\top A + U_0$$
$$= U^\top f(W^\top x + W_0) + U_0$$
$$J(W, W_0, U, U_0) = \frac{1}{n}\sum_{i=1}^{n}(y^{(i)} - g^{(i)})^2$$

Input

Preactivation

Nonlinearity

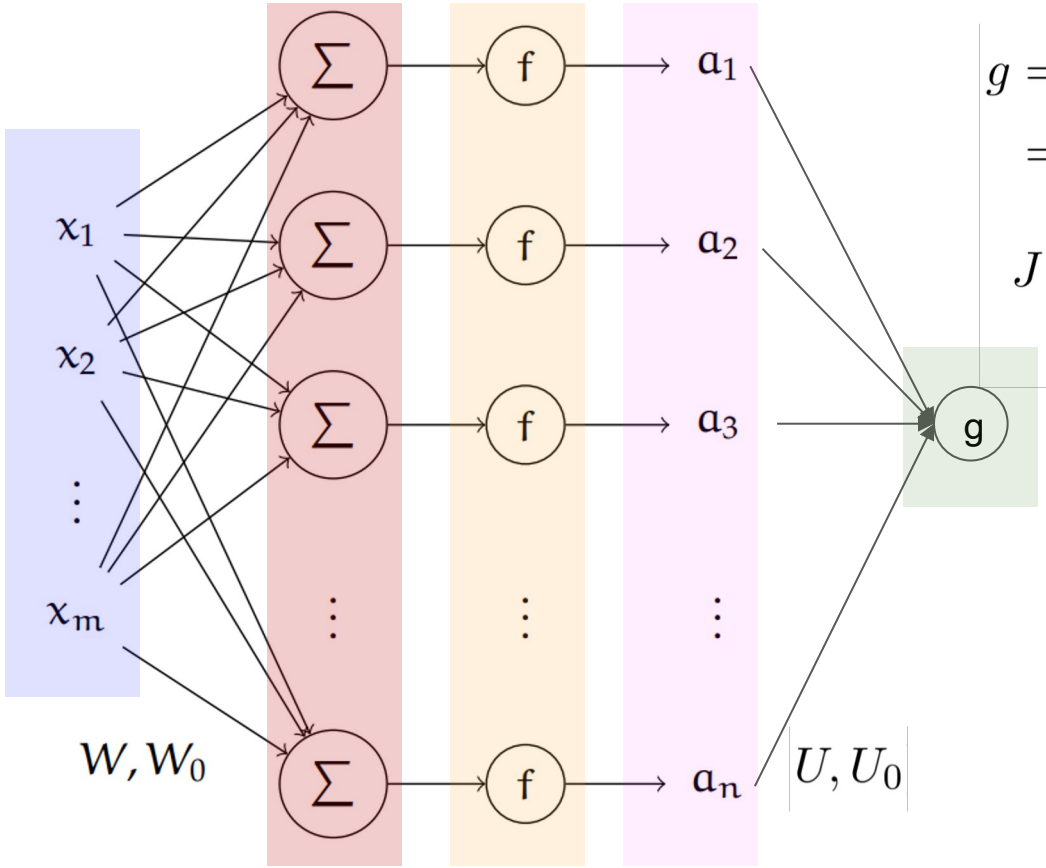
Two Layer Neural Network for Regression



$$g = U^\top A + U_0$$
$$= U^\top f(W^\top x + W_0) + U_0$$
$$J(W, W_0, U, U_0) = \frac{1}{n}\sum_{i=1}^{n}(y^{(i)} - g^{(i)})^2$$

Input
Preactivation
Nonlinearity
Hidden Layer

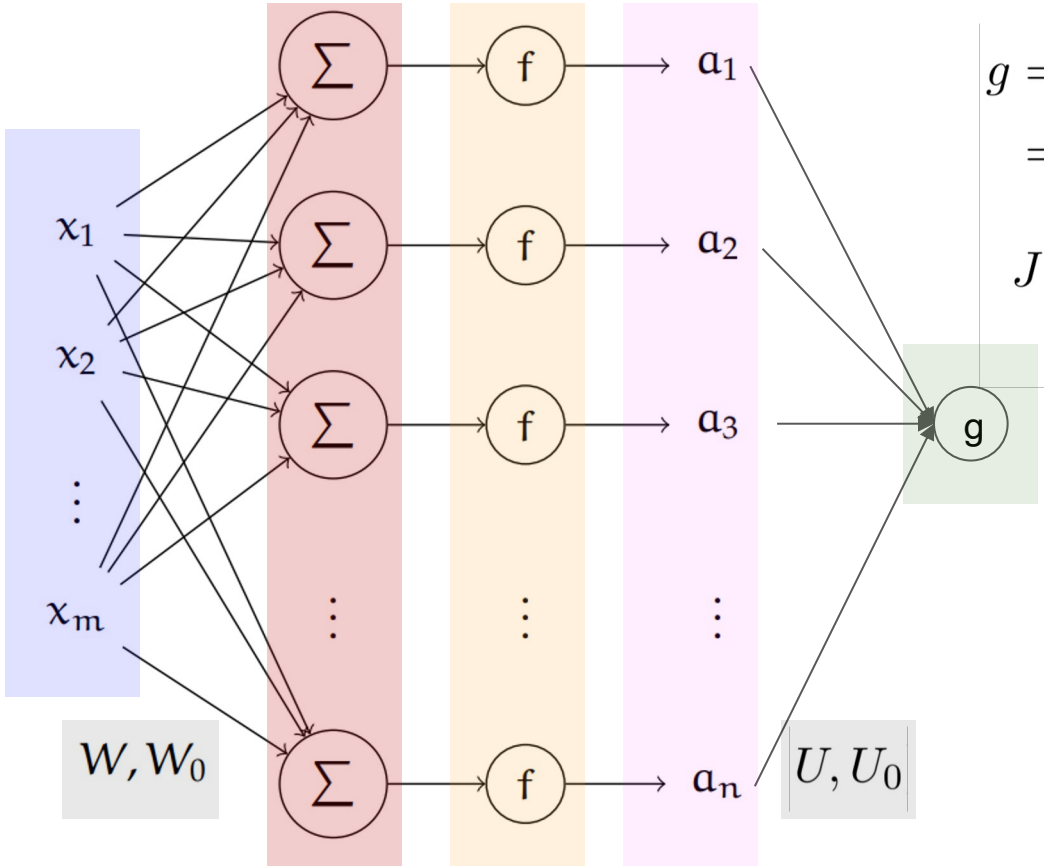
Two Layer Neural Network for Regression



$$g = U^\top A + U_0$$
$$= U^\top f(W^\top x + W_0) + U_0$$
$$J(W, W_0, U, U_0) = \frac{1}{n}\sum_{i=1}^{n}(y^{(i)} - g^{(i)})^2$$

Input
Preactivation
Nonlinearity
Hidden Layer
Output

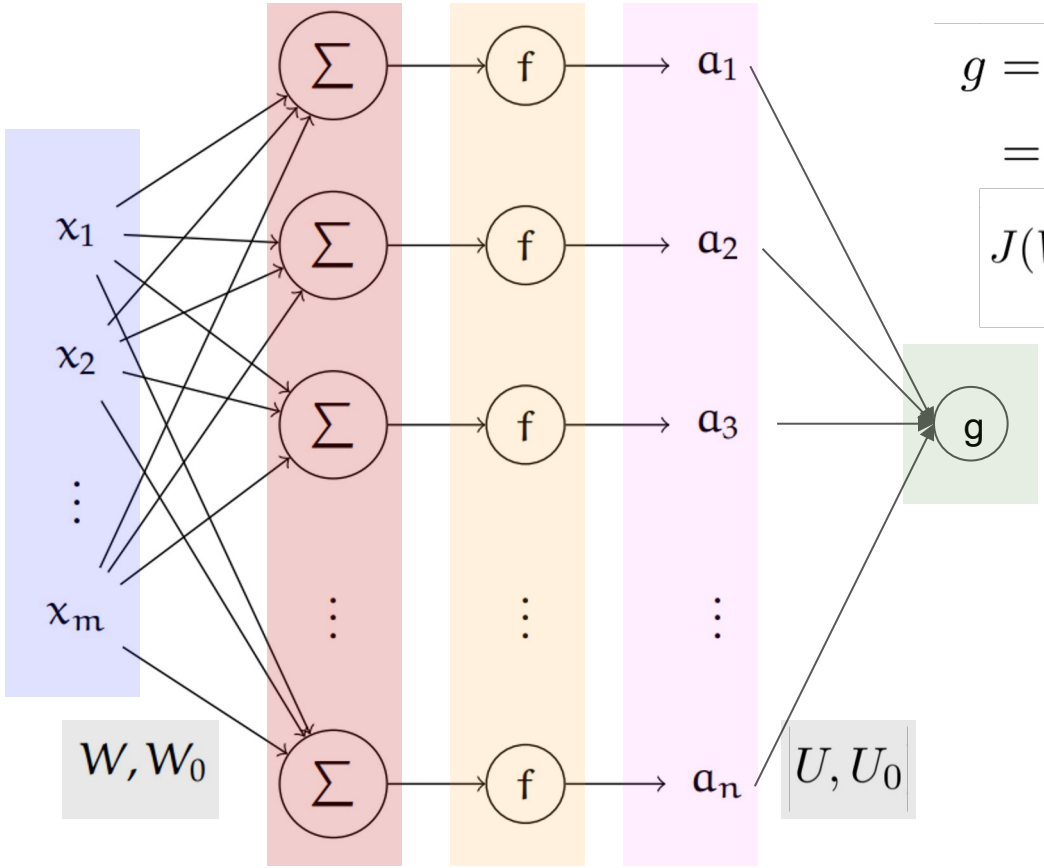
Two Layer Neural Network for Regression



$$g = U^\top A + U_0$$
$$= U^\top f(W^\top x + W_0) + U_0$$
$$J(W, W_0, U, U_0) = \frac{1}{n}\sum_{i=1}^{n}(y^{(i)} - g^{(i)})^2$$

Input
Preactivation
Nonlinearity
Hidden Layer
Output
Learnable Parameters

Two Layer Neural Network for Classification

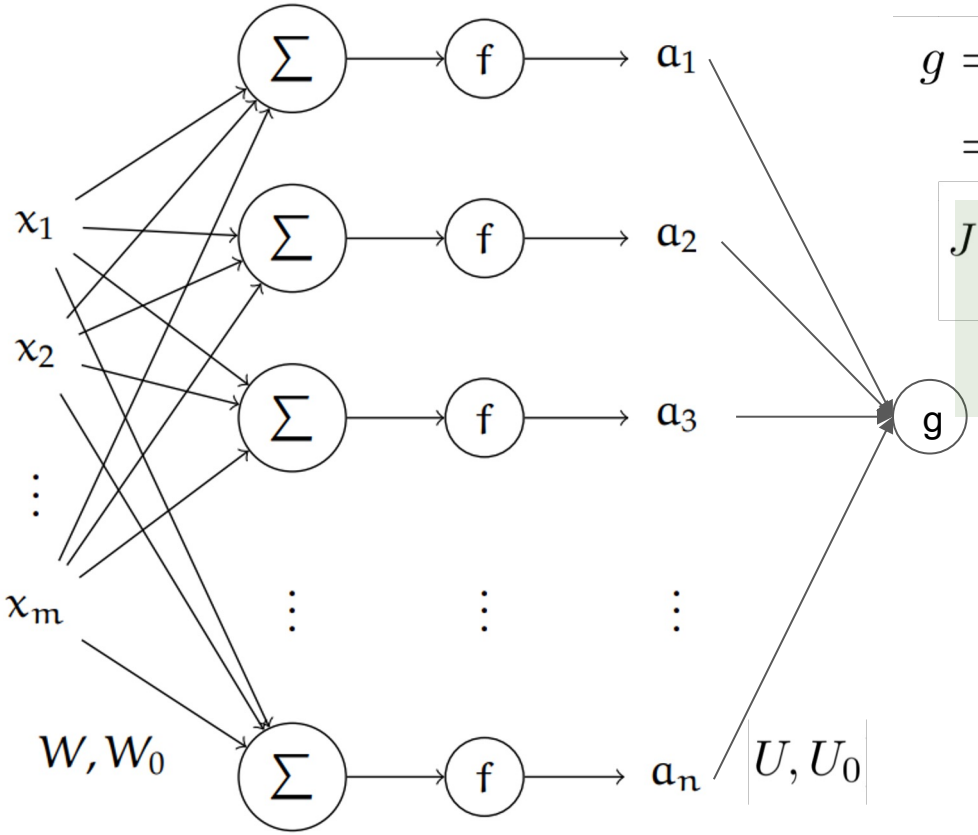


$$g = \sigma(U^\top A + U_0)$$
$$= \sigma(U^\top f(W^\top x + W_0) + U_0)$$

$$J(W, W_0, U, U_0) =$$
$$\frac{1}{n}\sum_{i=1}^{n} y^{(i)} \log g^{(i)} + (1 - y^{(i)}) \log(1 - g^{(i)})$$

Input
Preactivation
Nonlinearity
Hidden Layer
Output
Learnable Parameters

Two Layer Neural Network for Classification

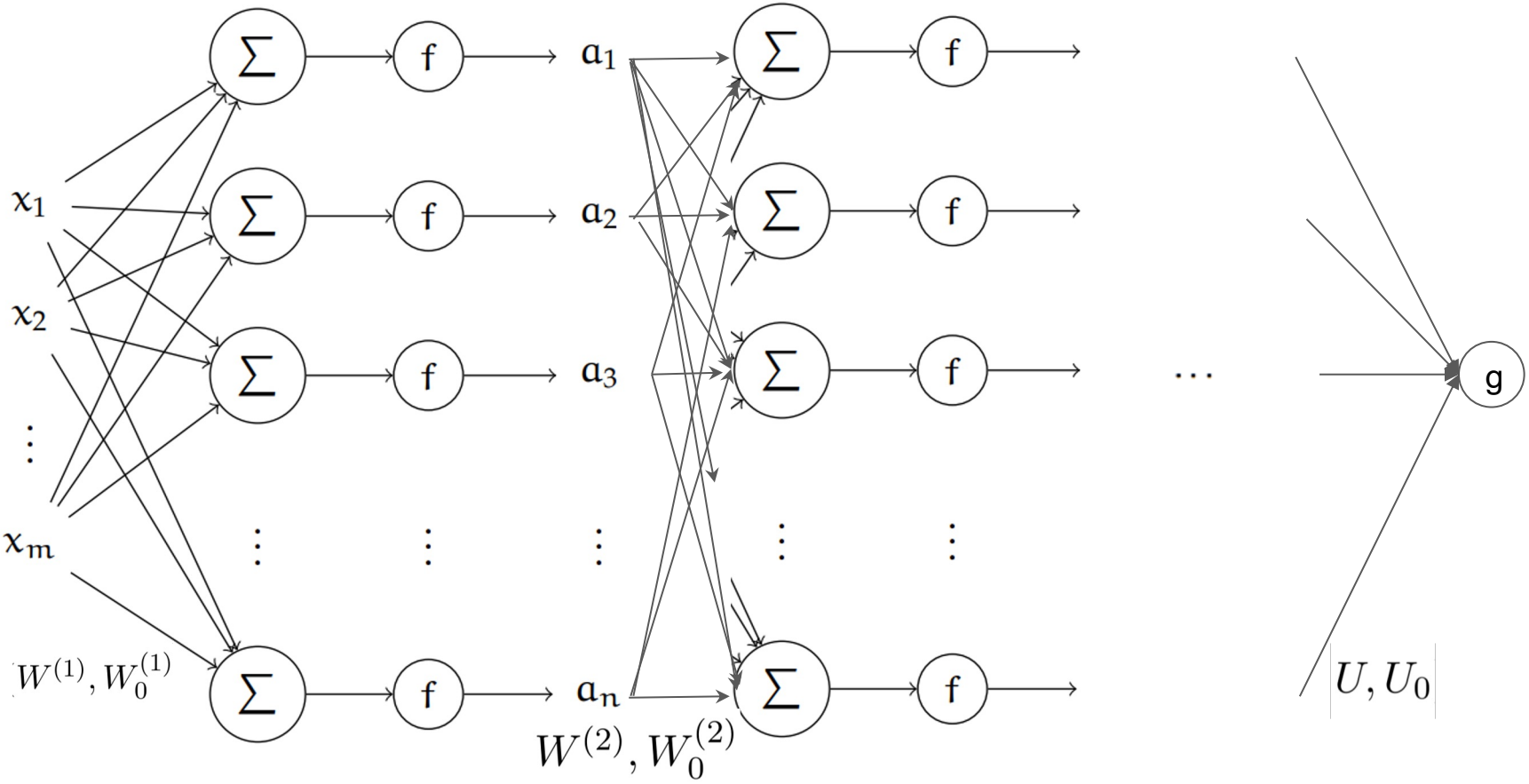


$$g = \sigma(U^\top A + U_0)$$
$$= \sigma(U^\top f(W^\top x + W_0) + U_0)$$

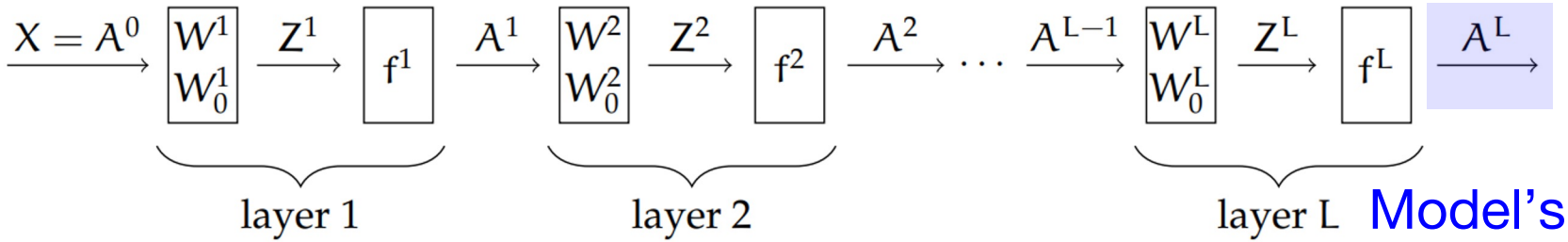
$$J(W, W_0, U, U_0) = \frac{1}{n}\sum_{i=1}^{n} y^{(i)} \log g^{(i)} + (1 - y^{(i)}) \log(1 - g^{(i)})$$

This objective is a differentiable function of model parameters and hence we can perform gradient-based optimization as in Logistic regression

Multi-layer Neural Networks



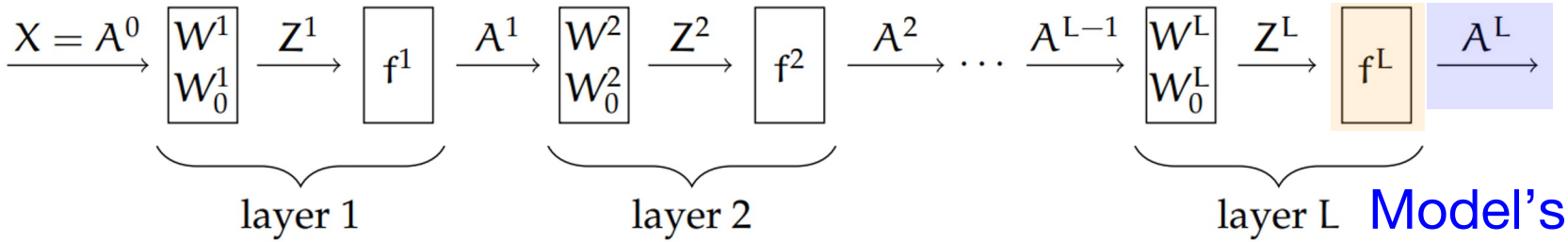
Multi-layer Neural Networks



$$Z^l = W^{l\,T} A^{l-1} + W_0^l$$

$$A^l = f^l(Z^l)$$

Multi-layer Neural Networks



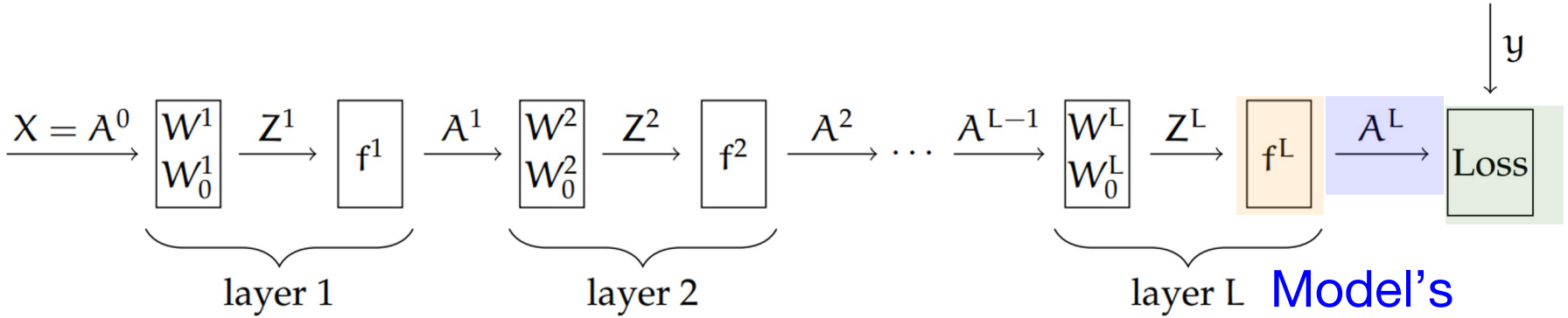
$$Z^l = W^{l\,T} A^{l-1} + W_0^l$$

$$A^l = f^l(Z^l)$$

f^L	task
linear	regression
sigmoid	classification
softmax	multi-class classification

Last activation function depends on problem at hand

Multi-layer Neural Networks



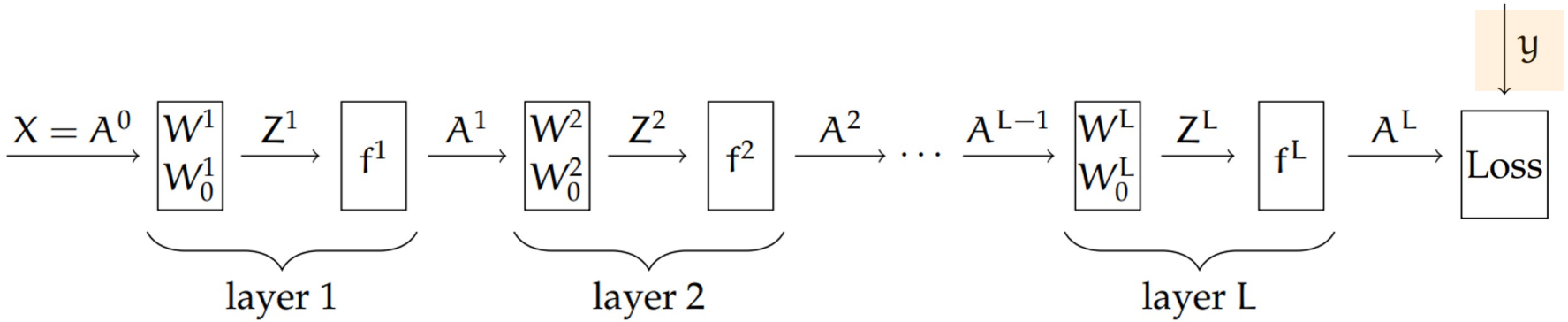
$$Z^l = W^{l\,T} A^{l-1} + W_0^l$$

$$A^l = f^l(Z^l)$$

Loss	f^L	task
squared	linear	regression
NLL	sigmoid	classification
NLLM	softmax	multi-class classification

Last activation function and loss function depends on problem at hand

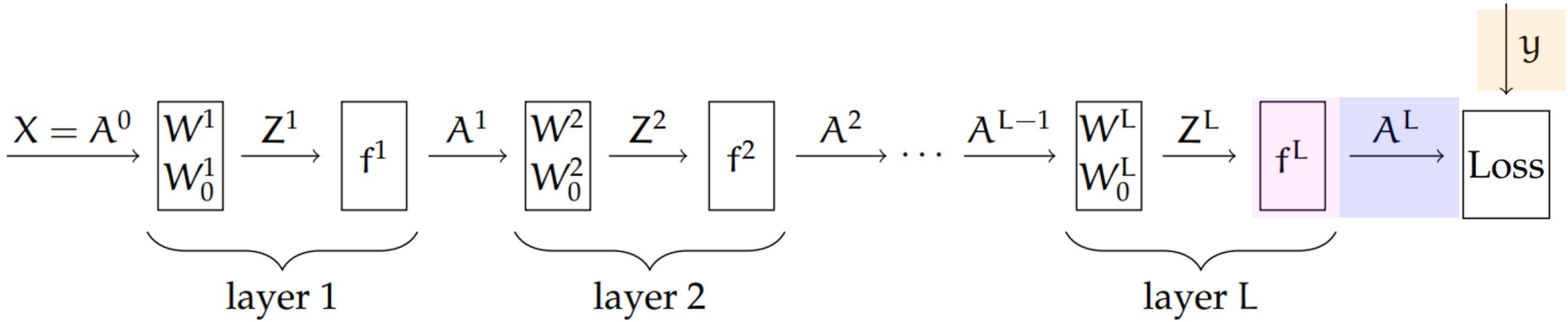
Multi-class Classification



$y \in \{1, \ldots, K\}$

$y \in \{[1, 0, \ldots 0], [0, 1, \ldots, 0], \ldots [0, \ldots, 0, 1]\}$

Multi-class Classification



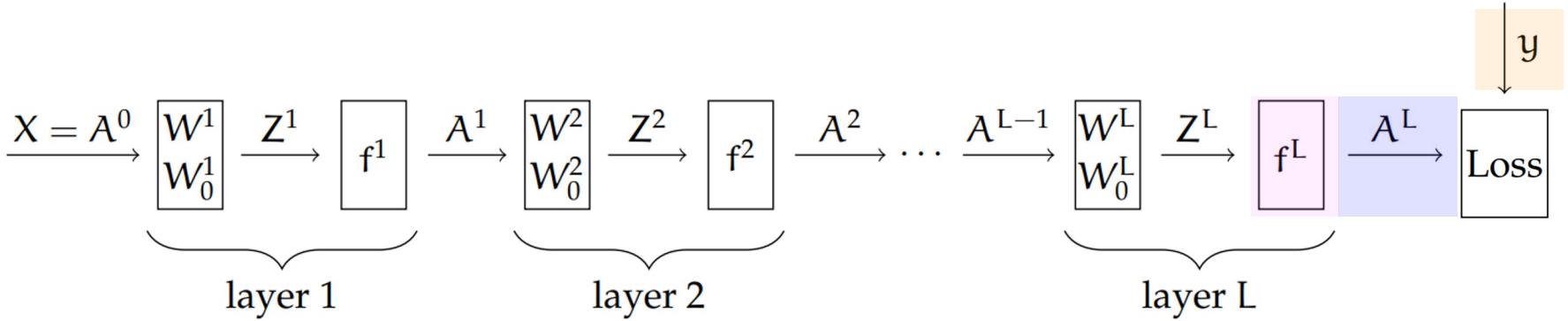
$y \in \{1, \ldots, K\}$

$y \in \{[1, 0, \ldots 0], [0, 1, \ldots, 0], \ldots [0, \ldots, 0, 1]\}$

$$g = A^L = f^L(Z^L) = \text{softmax}(Z^L)$$

$$= \begin{bmatrix} \exp(z_1)/\sum_i \exp(z_i) \\ \vdots \\ \exp(z_k)/\sum_i \exp(z_i) \end{bmatrix}$$

Multi-class Classification

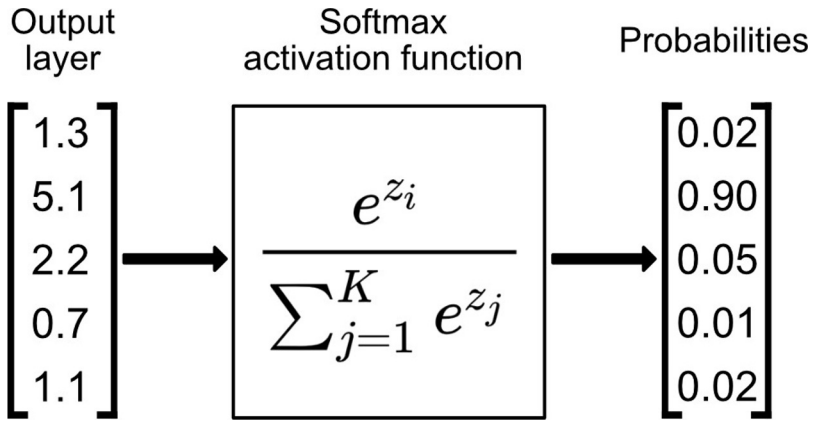


$$y \in \{1, \ldots, K\}$$

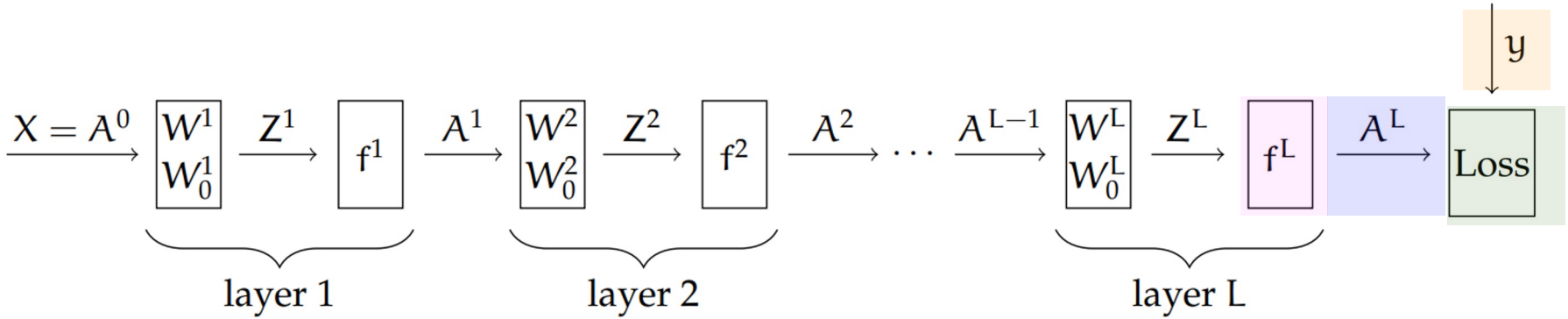
$$y \in \{[1, 0, \ldots 0], [0, 1, \ldots, 0], \ldots [0, \ldots, 0, 1]\}$$

$$g = A^L = f^L(Z^L) = \text{softmax}(Z^L)$$

$$= \begin{bmatrix} \exp(z_1)/\sum_i \exp(z_i) \\ \vdots \\ \exp(z_k)/\sum_i \exp(z_i) \end{bmatrix}$$



Multi-class Classification



$y \in \{1, \ldots, K\}$

$y \in \{[1, 0, \ldots 0], [0, 1, \ldots, 0], \ldots [0, \ldots, 0, 1]\}$

$$g = A^L = f^L(Z^L) = \text{softmax}(Z^L)$$

$$= \begin{bmatrix} \exp(z_1)/\sum_i \exp(z_i) \\ \vdots \\ \exp(z_k)/\sum_i \exp(z_i) \end{bmatrix}$$

$$\mathcal{L}_{\text{nllm}}(g, y) = -\sum_{k=1}^{K} y_k \cdot \log(g_k)$$

Multi-class negative log likelihood. (Still differentiable)

Gradient Descent

- All model parameters are differentiable functions of the loss ⇒ can train with (stochastic) GD

- Backpropagation: an approach to efficiently obtain gradients in multilayer neural networks.

- An instance of **dynamic programming**. Avoid redundant computation by breaking down the gradient expression into shared subproblems.

Review: Matrix Chain Rule

One layer neural network with linear activation:
Input dimension = 4, output dimension = 3

$$\underset{3\text{x}1}{A^{(1)}} = \underbrace{W^{(1)\top}}_{4\text{x}3} \underset{4\text{x}1}{x} + \underset{3\text{x}1}{W_0^{(1)}}$$

$$\underset{1\text{x}1}{\text{loss}} = L(A^{(1)}, y) = \sum_{k=1}^{3} (A_k^{(1)} - y_k)^2$$

Review: Matrix Chain Rule

One layer neural network with linear activation:
Input dimension = 4, output dimension = 3

$$\underset{3\text{x}1}{A^{(1)}} = \underbrace{W^{(1)\top}}_{4\text{x}3} \underset{4\text{x}1}{x} + \underset{3\text{x}1}{W_0^{(1)}}$$

$$\underset{1\text{x}1}{\text{loss}} = L(A^{(1)}, y) = \sum_{k=1}^{3} (A_k^{(1)} - y_k)^2$$

$$\underset{1\text{x}1}{\frac{\partial \text{loss}}{\partial W_{0,j}^{(1)}}} = \underset{1\text{x}3}{\frac{\partial A^{(1)}}{\partial W_{0,j}^{(1)}}} \underset{3\text{x}1}{\frac{\partial \text{loss}}{\partial A^{(1)}}}$$

Matrix chain rule

Review: Matrix Chain Rule

One layer neural network with linear activation:
Input dimension = 4, output dimension = 3

$$\underset{3\text{x}1}{A^{(1)}} = \underbrace{W^{(1)\top}}_{4\text{x}3} \underset{4\text{x}1}{x} + \underset{3\text{x}1}{W_0^{(1)}} \qquad \underset{1\text{x}1}{\text{loss}} = L(A^{(1)}, y) = \sum_{k=1}^{3} (A_k^{(1)} - y_k)^2$$

$$\underset{1\text{x}1}{\frac{\partial \text{loss}}{\partial W_{0,j}^{(1)}}} = \underset{1\text{x}3}{\frac{\partial A^{(1)}}{\partial W_{0,j}^{(1)}}} \underset{3\text{x}1}{\frac{\partial \text{loss}}{\partial A^{(1)}}} = \sum_{k=1}^{3} \underset{1\text{x}1}{\frac{\partial \text{loss}}{\partial A_k^{(1)}}} \underset{1\text{x}1}{\frac{\partial A_k^{(1)}}{\partial W_{0,j}^{(1)}}}$$

Matrix chain rule Scalar chain rule

Review: Matrix Chain Rule

One layer neural network with linear activation:
Input dimension = 4, output dimension = 3

$$\underset{3\text{x}1}{A^{(1)}} = \underbrace{W^{(1)\top}}_{4\text{x}3} \underset{4\text{x}1}{x} + \underset{3\text{x}1}{W_0^{(1)}}$$

$$\underset{1\text{x}1}{\text{loss}} = L(A^{(1)}, y) = \sum_{k=1}^{3} (A_k^{(1)} - y_k)^2$$

$$\underset{1\text{x}1}{\frac{\partial \text{loss}}{\partial W_{0,j}^{(1)}}} = \underset{1\text{x}3}{\frac{\partial A^{(1)}}{\partial W_{0,j}^{(1)}}} \underset{3\text{x}1}{\frac{\partial \text{loss}}{\partial A^{(1)}}}$$

Matrix chain rule

$$\underset{3\text{x}1}{\frac{\partial \text{loss}}{\partial W_0^{(1)}}} = \underset{3\text{x}3}{\frac{\partial A^{(1)}}{\partial W_0^{(1)}}} \underset{3\text{x}1}{\frac{\partial \text{loss}}{\partial A^{(1)}}}$$

Can calculate for all W0 via matrix multiplications!

Review: Matrix Chain Rule

One layer neural network with linear activation:
Input dimension = 4, output dimension = 3

$$A^{(1)} = W^{(1)\top} x + W_0^{(1)} \qquad \text{loss} = L(A^{(1)}, y) = \sum_{k=1}^{3} (A_k^{(1)} - y_k)^2$$

3x1 4x3 4x1 3x1 1x1

1x1
$$\frac{\partial \text{loss}}{\partial W_{j,k}^{(1)}} = \sum_{p=1}^{3} \frac{\partial A_p^{(1)}}{\partial W_{j,k}^{(1)}} \frac{\partial \text{loss}}{\partial A_p^{(1)}} = \frac{\partial A_k^{(1)}}{\partial W_{j,k}^{(1)}} \frac{\partial \text{loss}}{\partial A_k^{(1)}} = x_j \frac{\partial \text{loss}}{\partial A_k^{(1)}}$$

Review: Matrix Chain Rule

One layer neural network with linear activation:
Input dimension = 4, output dimension = 3

$$\underset{3\text{x}1}{A^{(1)}} = \underbrace{W^{(1)\top}}_{4\text{x}3} \underset{4\text{x}1}{x} + \underset{3\text{x}1}{W_0^{(1)}} \qquad \underset{1\text{x}1}{\text{loss}} = L(A^{(1)}, y) = \sum_{k=1}^{3} (A_k^{(1)} - y_k)^2$$

$$\underset{1\text{x}1}{\frac{\partial \text{loss}}{\partial W_{j,k}^{(1)}}} = \sum_{p=1}^{3} \frac{\partial A_p^{(1)}}{\partial W_{j,k}^{(1)}} \frac{\partial \text{loss}}{\partial A_p^{(1)}} = \frac{\partial A_k^{(1)}}{\partial W_{j,k}^{(1)}} \frac{\partial \text{loss}}{\partial A_k^{(1)}} = x_j \frac{\partial \text{loss}}{\partial A_k^{(1)}}$$

$$\underset{4\text{x}3}{\frac{\partial \text{loss}}{\partial W^{(1)}}} = \underset{4\text{x}1}{x} \underbrace{\left(\frac{\partial \text{loss}}{\partial A^{(1)}} \right)^{\top}}_{3\text{x}1}$$

Again, can calculate everything at once via matrix multiplications (outer products)

Review: Matrix Chain Rule

One layer neural network with linear activation:
Input dimension = 4, output dimension = 3

$$\underset{3\text{x}1}{A^{(1)}} = \underset{4\text{x}3}{W^{(1)\top}} \underset{4\text{x}1}{x} + \underset{3\text{x}1}{W_0^{(1)}} \qquad \underset{1\text{x}1}{\text{loss}} = L(A^{(1)}, y) = \sum_{k=1}^{3} (A_k^{(1)} - y_k)^2$$

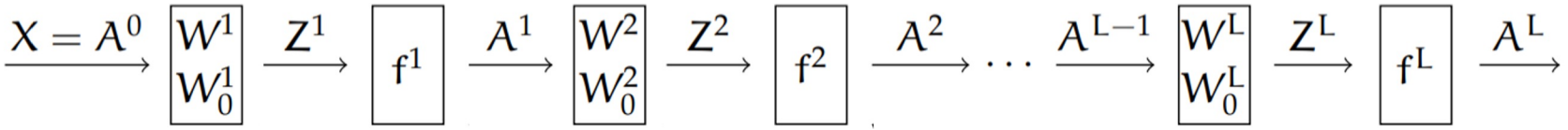
$$\underset{1\text{x}1}{\frac{\partial \text{loss}}{\partial W_{j,k}^{(1)}}} = \sum_{p=1}^{3} \frac{\partial A_p^{(1)}}{\partial W_{j,k}^{(1)}} \frac{\partial \text{loss}}{\partial A_p^{(1)}} = \frac{\partial A_k^{(1)}}{\partial W_{j,k}^{(1)}} \frac{\partial \text{loss}}{\partial A_k^{(1)}} = x_j \frac{\partial \text{loss}}{\partial A_k^{(1)}}$$

$$\underset{4\text{x}3}{\frac{\partial \text{loss}}{\partial W^{(1)}}} = \underset{4\text{x}1}{x} \underset{3\text{x}1}{\left(\frac{\partial \text{loss}}{\partial A^{(1)}} \right)^{\top}}$$

Exercise: convince yourselves that this is correct!!

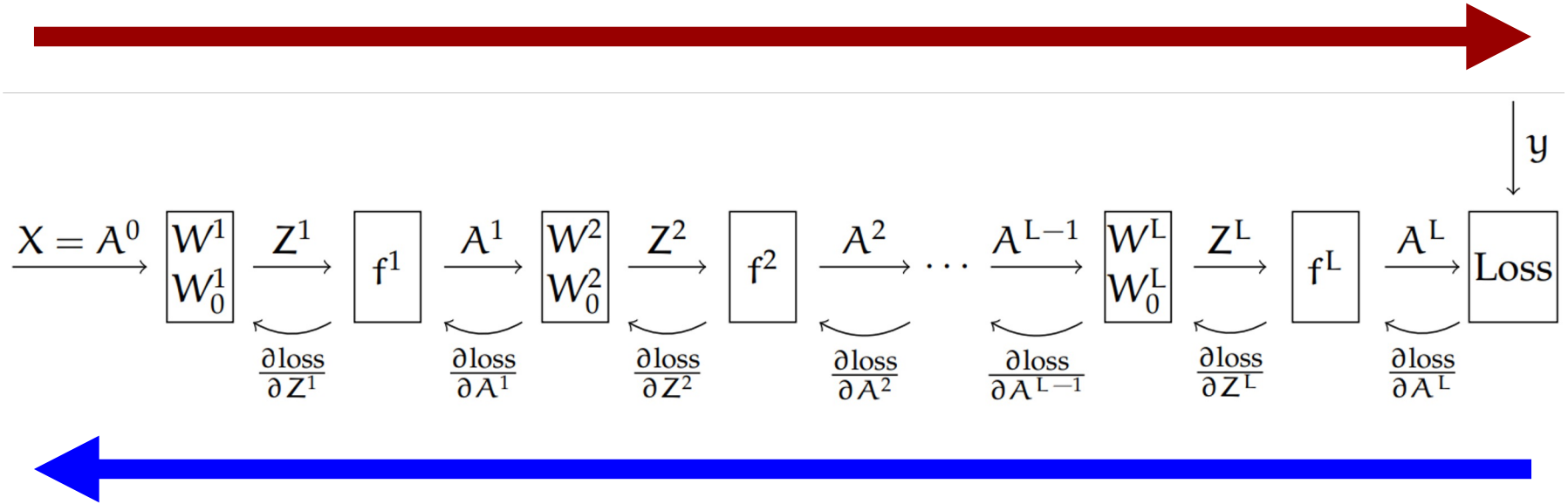
Backpropagation Intuition

Forward propagation to obtain the output (model's guess)



Backpropagation Intuition

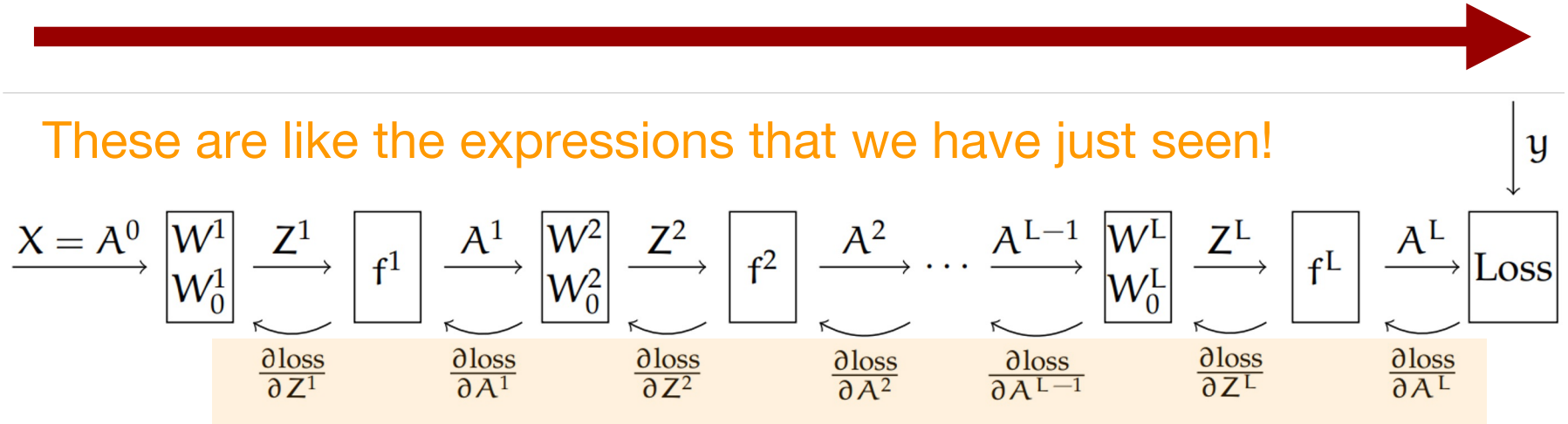
Forward propagation to obtain the output (model's guess)



Backpropagation to obtain gradients with respect to the loss

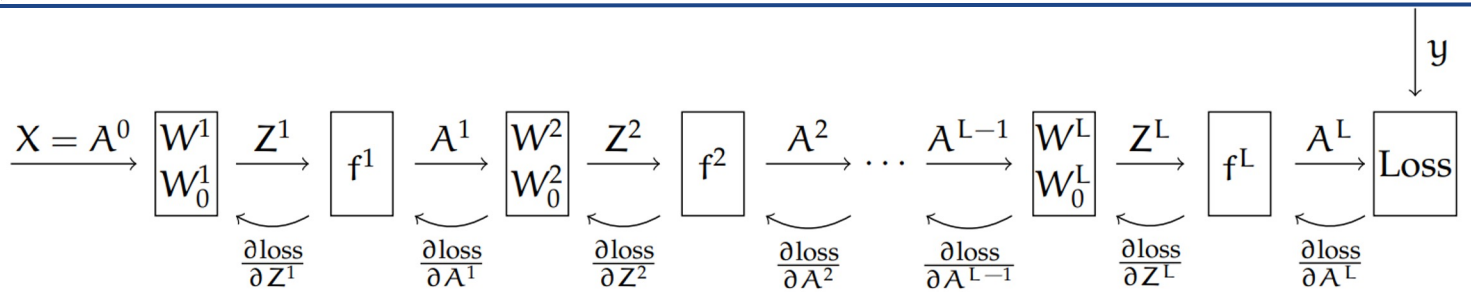
Backpropagation Intuition

Forward propagation to obtain the output (model's guess)



Backpropagation to obtain gradients with respect to the loss

Backpropagation Math

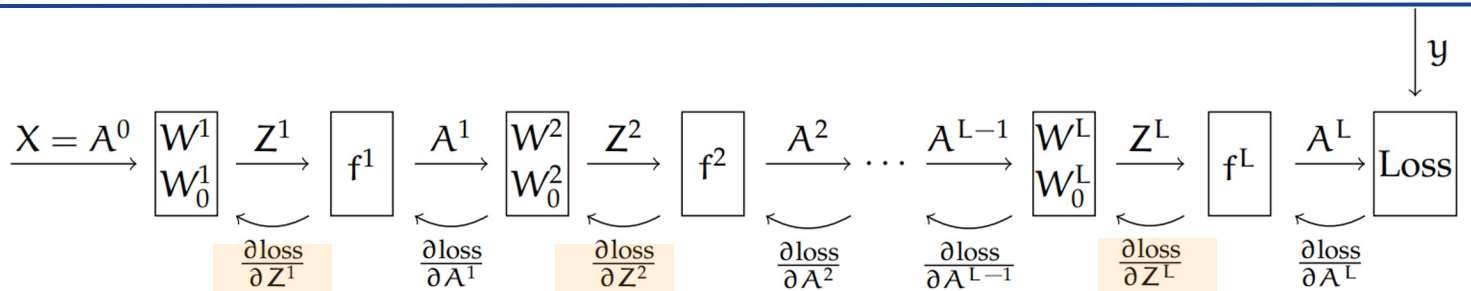


$$A^{(\ell)} = f^{\ell}(Z^{(\ell)}), Z^{(\ell)} = W^{(\ell)\top} A^{(\ell-1)} + W_0^{(\ell)}$$

$$\frac{\partial \text{loss}}{\partial W_{j,k}^{(\ell)}} = \frac{\partial Z_k^{(\ell)}}{\partial W_{j,k}^{(\ell)}} \frac{\partial \text{loss}}{\partial Z_k^{(\ell)}} = A_j^{(\ell-1)} \frac{\partial \text{loss}}{\partial Z_k^{(\ell)}} \Rightarrow \frac{\partial \text{loss}}{\partial W^{(\ell)}} = A^{(\ell-1)} \left(\frac{\partial \text{loss}}{\partial Z^{(\ell)}} \right)^{\top}$$

1x1 1x1 1x1 1x1 1x1 m^ℓxn^ℓ m^ℓx1 1xn^ℓ

Backpropagation Math

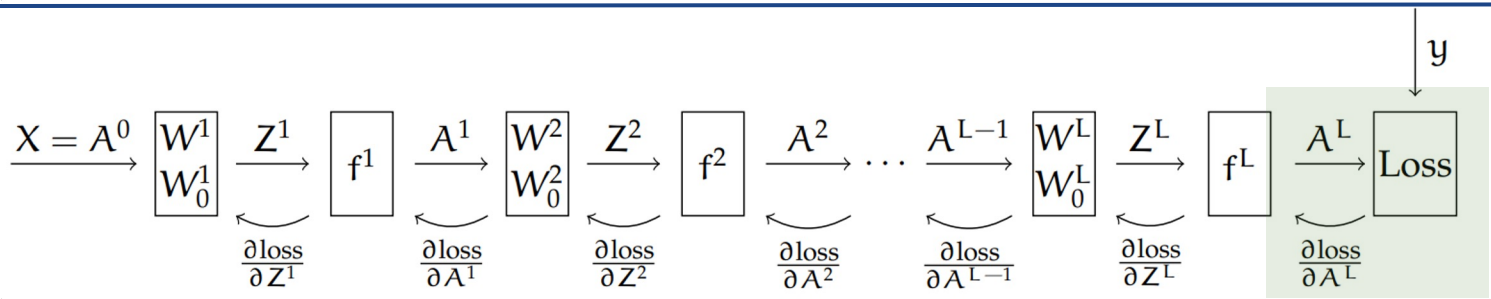


$$A^{(\ell)} = f^{\ell}(Z^{(\ell)}), Z^{(\ell)} = W^{(\ell)\top} A^{(\ell-1)} + W_0^{(\ell)}$$

$$\frac{\partial \text{loss}}{\partial W_{j,k}^{(\ell)}} = \frac{\partial Z_k^{(\ell)}}{\partial W_{j,k}^{(\ell)}} \frac{\partial \text{loss}}{\partial Z_k^{(\ell)}} = A_j^{(\ell-1)} \frac{\partial \text{loss}}{\partial Z_k^{(\ell)}} \Rightarrow \frac{\partial \text{loss}}{\partial W^{(\ell)}} = A^{(\ell-1)} \left(\frac{\partial \text{loss}}{\partial Z^{(\ell)}} \right)^{\top}$$

1x1 1x1 1x1 1x1 1x1 $m^\ell \text{x} n^\ell$ $m^\ell \text{x} 1$ $1 \text{x} n^\ell$

Backpropagation Math



$$A^{(\ell)} = f^{\ell}(Z^{(\ell)}), Z^{(\ell)} = W^{(\ell)\top} A^{(\ell-1)} + W_0^{(\ell)}$$

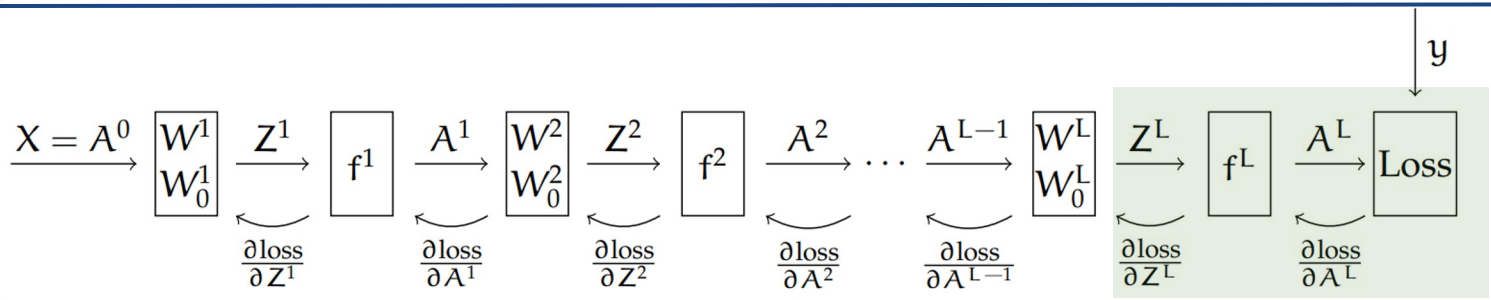
$$\frac{\partial \text{loss}}{\partial W_{j,k}^{(\ell)}} = \frac{\partial Z_k^{(\ell)}}{\partial W_{j,k}^{(\ell)}} \frac{\partial \text{loss}}{\partial Z_k^{(\ell)}} = A_j^{(\ell-1)} \frac{\partial \text{loss}}{\partial Z_k^{(\ell)}} \Rightarrow \frac{\partial \text{loss}}{\partial W^{(\ell)}} = A^{(\ell-1)} \left(\frac{\partial \text{loss}}{\partial Z^{(\ell)}} \right)^{\top}$$

1x1 1x1 1x1 1x1 1x1 m^{ℓ}xn^{ℓ} m^{ℓ}x1 1xn^{ℓ}

$$\frac{\partial \text{loss}}{\partial Z^{(\ell)}} = \qquad \frac{\partial \text{loss}}{\partial A^{(L)}}$$

n^{ℓ}x1 n^{L}x1

Backpropagation Math



$$A^{(\ell)} = f^{\ell}(Z^{(\ell)}), Z^{(\ell)} = W^{(\ell)\top} A^{(\ell-1)} + W_0^{(\ell)}$$

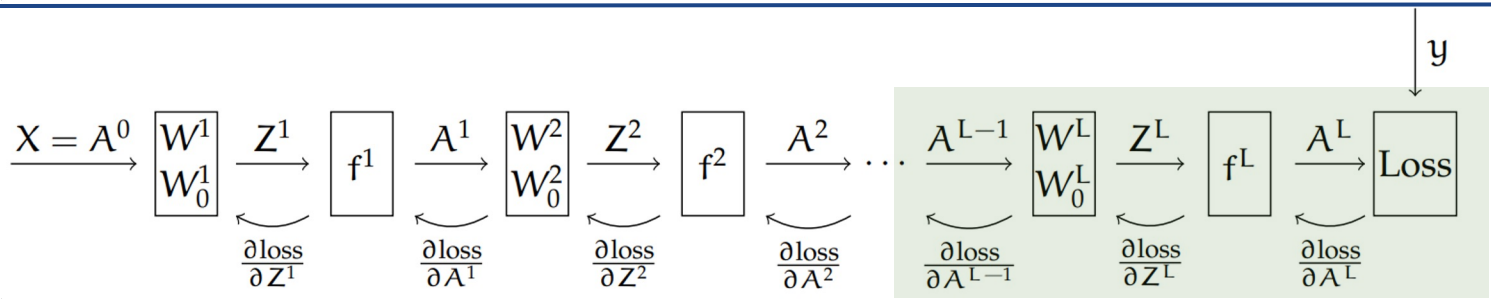
$$\frac{\partial \text{loss}}{\partial W_{j,k}^{(\ell)}} = \frac{\partial Z_k^{(\ell)}}{\partial W_{j,k}^{(\ell)}} \frac{\partial \text{loss}}{\partial Z_k^{(\ell)}} = A_j^{(\ell-1)} \frac{\partial \text{loss}}{\partial Z_k^{(\ell)}} \Rightarrow \frac{\partial \text{loss}}{\partial W^{(\ell)}} = A^{(\ell-1)} \left(\frac{\partial \text{loss}}{\partial Z^{(\ell)}} \right)^{\top}$$

1x1 1x1 1x1 1x1 1x1 m^{ℓ}xn^{ℓ} m^{ℓ}x1 1xn^{ℓ}

$$\frac{\partial \text{loss}}{\partial Z^{(\ell)}} = \cdot \frac{\partial A^{(L)}}{\partial Z^{(L)}} \frac{\partial \text{loss}}{\partial A^{(L)}}$$

n^{ℓ}x1 n^{L}xn^{L} n^{L}x1

Backpropagation Math



$$A^{(\ell)} = f^\ell(Z^{(\ell)}), Z^{(\ell)} = W^{(\ell)\top} A^{(\ell-1)} + W_0^{(\ell)}$$

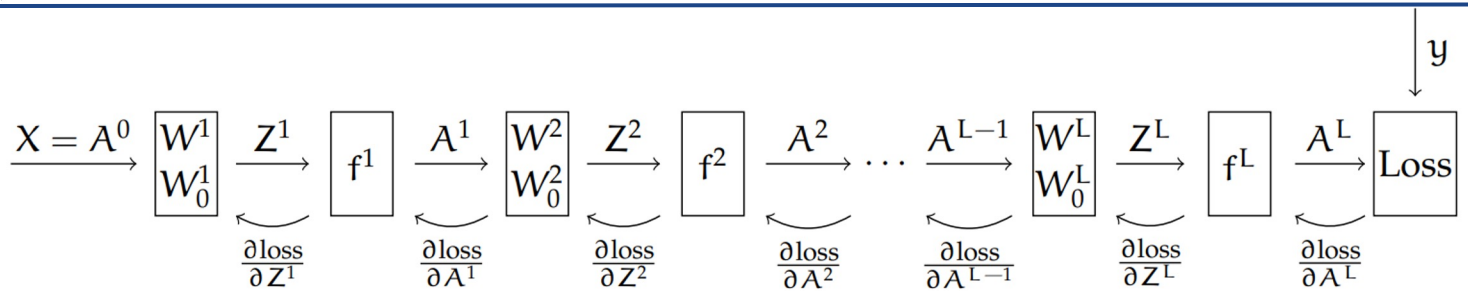
$$\frac{\partial \text{loss}}{\partial W_{j,k}^{(\ell)}} = \frac{\partial Z_k^{(\ell)}}{\partial W_{j,k}^{(\ell)}} \frac{\partial \text{loss}}{\partial Z_k^{(\ell)}} = A_j^{(\ell-1)} \frac{\partial \text{loss}}{\partial Z_k^{(\ell)}} \Rightarrow \frac{\partial \text{loss}}{\partial W^{(\ell)}} = A^{(\ell-1)} \left(\frac{\partial \text{loss}}{\partial Z^{(\ell)}} \right)^\top$$

1x1 1x1 1x1 1x1 1x1 $m^\ell \text{x} n^\ell$ $m^\ell \text{x} 1$ $1 \text{x} n^\ell$

$$\frac{\partial \text{loss}}{\partial Z^{(\ell)}} = \frac{\partial Z^{(L)}}{\partial A^{(L-1)}} \frac{\partial A^{(L)}}{\partial Z^{(L)}} \frac{\partial \text{loss}}{\partial A^{(L)}}$$

$n^\ell \text{x} 1$ $n^{L-1} \text{x} n^L$ $n^L \text{x} n^L$ $n^L \text{x} 1$

Backpropagation Math



$$A^{(\ell)} = f^{\ell}(Z^{(\ell)}), Z^{(\ell)} = W^{(\ell)\top} A^{(\ell-1)} + W_0^{(\ell)}$$

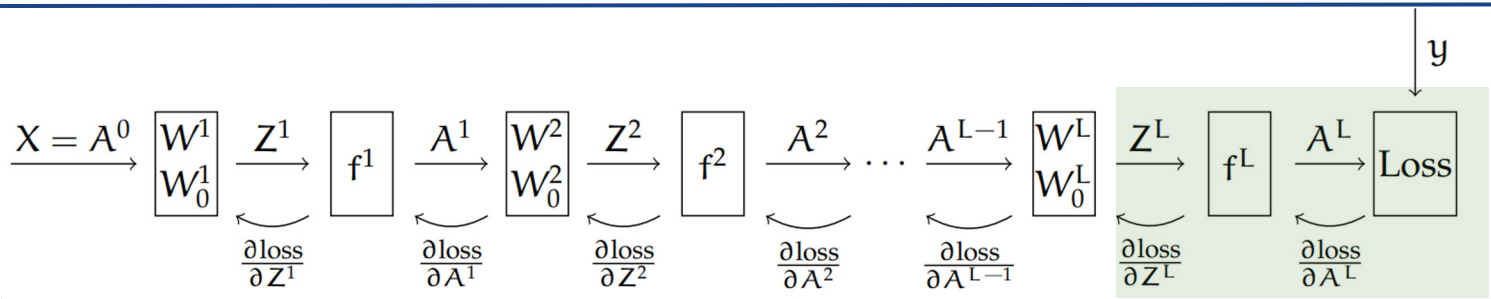
$$\frac{\partial \text{loss}}{\partial W_{j,k}^{(\ell)}} = \frac{\partial Z_k^{(\ell)}}{\partial W_{j,k}^{(\ell)}} \frac{\partial \text{loss}}{\partial Z_k^{(\ell)}} = A_j^{(\ell-1)} \frac{\partial \text{loss}}{\partial Z_k^{(\ell)}} \Rightarrow \frac{\partial \text{loss}}{\partial W^{(\ell)}} = A^{(\ell-1)} \left(\frac{\partial \text{loss}}{\partial Z^{(\ell)}} \right)^{\top}$$

1x1 1x1 1x1 1x1 1x1 $m^{\ell} \times n^{\ell}$ $m^{\ell} \times 1$ $1 \times n^{\ell}$

$$\frac{\partial \text{loss}}{\partial Z^{(\ell)}} = \frac{\partial A^{(\ell)}}{\partial Z^{(\ell)}} \frac{\partial Z^{(\ell+1)}}{\partial A^{(\ell)}} \frac{\partial A^{(\ell+1)}}{\partial Z^{(\ell+1)}} \cdots \frac{\partial A^{(L-1)}}{\partial Z^{(L-1)}} \frac{\partial Z^{(L)}}{\partial A^{(L-1)}} \frac{\partial A^{(L)}}{\partial Z^{(L)}} \frac{\partial \text{loss}}{\partial A^{(L)}}$$

$n^{\ell} \times 1$ $n^{\ell} \times n^{\ell}$ $n^{\ell} \times n^{\ell+1}$ $n^{\ell+1} \times n^{\ell+1}$ $n^{L-1} \times n^{L-1}$ $n^{L-1} \times n^{L}$ $n^{L} \times n^{L}$ $n^{L} \times 1$

Backpropagation Math



$$A^{(\ell)} = f^{\ell}(Z^{(\ell)}), Z^{(\ell)} = W^{(\ell)\top} A^{(\ell-1)} + W_0^{(\ell)}$$

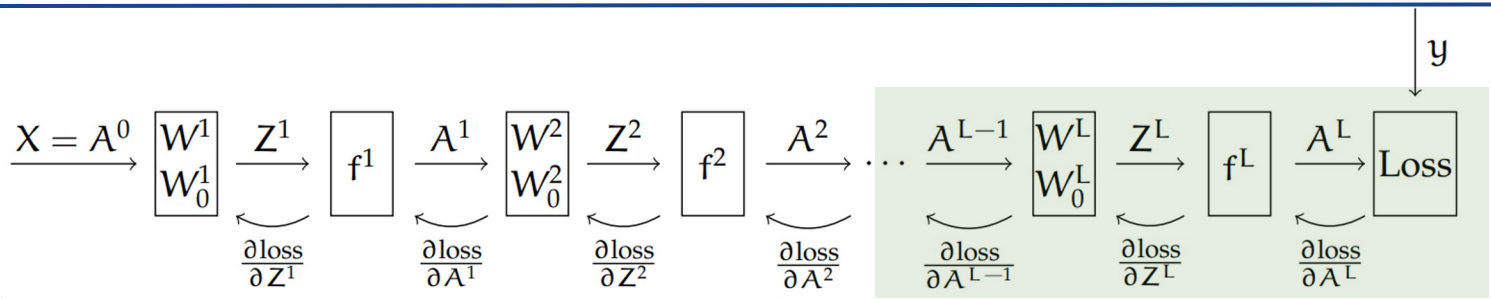
$$\frac{\partial \text{loss}}{\partial W_{j,k}^{(\ell)}} = \frac{\partial Z_k^{(\ell)}}{\partial W_{j,k}^{(\ell)}} \frac{\partial \text{loss}}{\partial Z_k^{(\ell)}} = A_j^{(\ell-1)} \frac{\partial \text{loss}}{\partial Z_k^{(\ell)}} \rightarrow \frac{\partial \text{loss}}{\partial W^{(\ell)}} = A^{(\ell-1)} \frac{\partial \text{loss}}{\partial Z^{(L)}}$$

1x1 1x1 1x1 1x1 1x1

$$\frac{\partial \text{loss}}{\partial Z^{(\ell)}} = \frac{\partial A^{(\ell)}}{\partial Z^{(\ell)}} \frac{\partial Z^{(\ell+1)}}{\partial A^{(\ell)}} \frac{\partial A^{(\ell+1)}}{\partial Z^{(\ell+1)}} \cdots \frac{\partial A^{(L-1)}}{\partial Z^{(L-1)}} \frac{\partial Z^{(L)}}{\partial A^{(L-1)}} \frac{\partial A^{(L)}}{\partial Z^{(L)}} \frac{\partial \text{loss}}{\partial A^{(L)}}$$

$n^{\ell} \times 1$ $n^{\ell} \times n^{\ell}$ $n^{\ell} \times n^{\ell+1}$ $n^{\ell+1} \times n^{\ell+1}$ $n^{L-1} \times n^{L-1}$ $n^{L-1} \times n^{L}$ $n^{L} \times n^{L}$ $n^{L} \times 1$

Backpropagation Math



$$A^{(\ell)} = f^{\ell}(Z^{(\ell)}), Z^{(\ell)} = W^{(\ell)\top} A^{(\ell-1)} + W_0^{(\ell)}$$

$$\frac{\partial \text{loss}}{\partial W_{j,k}^{(\ell)}} = \frac{\partial Z_k^{(\ell)}}{\partial W_{j,k}^{(\ell)}} \frac{\partial \text{loss}}{\partial Z_k^{(\ell)}} = A_j^{(\ell-1)} \frac{\partial \text{loss}}{\partial Z_k^{(\ell)}}$$

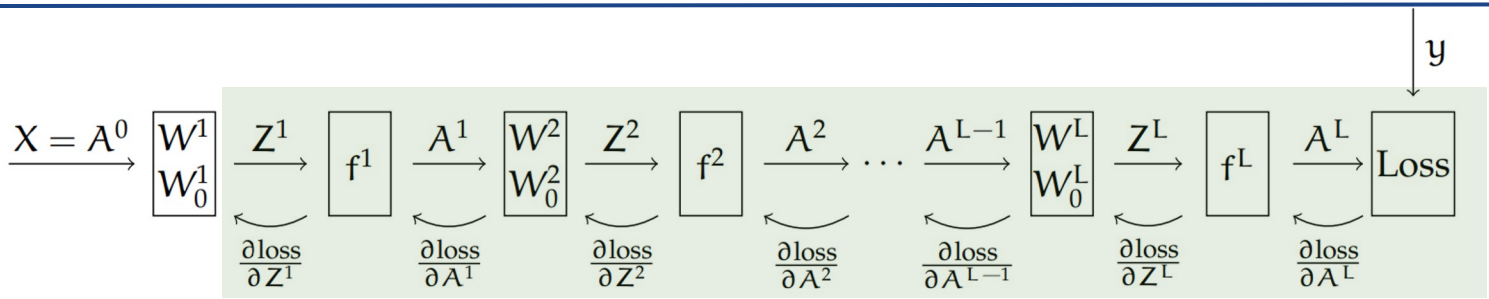
1x1 1x1 1x1 1x1 1x1

$$\frac{\partial \text{loss}}{\partial Z^{(L-1)}} \quad \frac{\partial \text{loss}}{\partial Z^{(L)}}$$

$$\frac{\partial \text{loss}}{\partial Z^{(\ell)}} = \frac{\partial A^{(\ell)}}{\partial Z^{(\ell)}} \frac{\partial Z^{(\ell+1)}}{\partial A^{(\ell)}} \frac{\partial A^{(\ell+1)}}{\partial Z^{(\ell+1)}} \cdots \frac{\partial A^{(L-1)}}{\partial Z^{(L-1)}} \frac{\partial Z^{(L)}}{\partial A^{(L-1)}} \frac{\partial A^{(L)}}{\partial Z^{(L)}} \frac{\partial \text{loss}}{\partial A^{(L)}}$$

$n^{\ell} \times 1$ $n^{\ell} \times n^{\ell}$ $n^{\ell} \times n^{\ell+1}$ $n^{\ell+1} \times n^{\ell+1}$ $n^{L-1} \times n^{L-1}$ $n^{L-1} \times n^{L}$ $n^{L} \times n^{L}$ $n^{L} \times 1$

Backpropagation Math



$$A^{(\ell)} = f^{\ell}(Z^{(\ell)}), Z^{(\ell)} = W^{(\ell)\top} A^{(\ell-1)} + W_0^{(\ell)}$$

$$\frac{\partial \text{loss}}{\partial W_{j,k}^{(\ell)}} = \frac{\partial \text{loss}}{\partial Z^{(1)}} \frac{\partial \text{ss}}{\partial \ell} = A_j^{(\ell-1} \cdots \frac{\partial \text{loss}}{\partial Z^{(L-1)}} \frac{\partial \text{loss}}{\partial Z^{(L)}}$$

1x1 1x1 1x1 1x1

$$\frac{\partial \text{loss}}{\partial Z^{(\ell)}} = \frac{\partial A^{(\ell)}}{\partial Z^{(\ell)}} \frac{\partial Z^{(\ell+1)}}{\partial A^{(\ell)}} \frac{\partial A^{(\ell+1)}}{\partial Z^{(\ell+1)}} \cdots \frac{\partial A^{(L-1)}}{\partial Z^{(L-1)}} \frac{\partial Z^{(L)}}{\partial A^{(L-1)}} \frac{\partial A^{(L)}}{\partial Z^{(L)}} \frac{\partial \text{loss}}{\partial A^{(L)}}$$

$n^{\ell} \times 1$ $n^{\ell} \times n^{\ell}$ $n^{\ell} \times n^{\ell+1}$ $n^{\ell+1} \times n^{\ell+1}$ $n^{L-1} \times n^{L-1}$ $n^{L-1} \times n^{L}$ $n^{L} \times n^{L}$ $n^{L} \times 1$

Backpropagation Math

Can calculate $\frac{\partial \text{loss}}{\partial Z^{(l)}}$ for all layers in one “backward” pass due to shared computations

$\frac{\partial \text{loss}}{\partial Z^{(l)}}$ needed to calculate $\frac{\partial \text{loss}}{\partial W^{(\ell)}}$ and $\frac{\partial \text{loss}}{\partial W_0^{(l)}}$

$$\frac{\partial \text{loss}}{\partial W^{(\ell)}} = A^{(\ell-1)} \left(\frac{\partial \text{loss}}{\partial Z^{(\ell)}} \right)^\top$$

That’s it!