Data Science and Deep Learning (2024)

Lecture 4

Multilayer Perceptron

Stan Z. Li

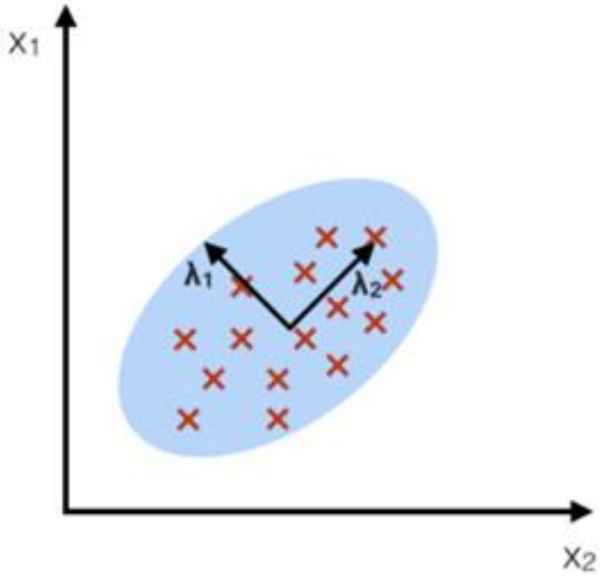
西湖大學

WESTLAKE UNIVERSITY

Linear Projections $\mathbf{y} = \mathbf{P}\mathbf{x}^{\mathrm{T}} \triangleq f(\mathbf{x})$

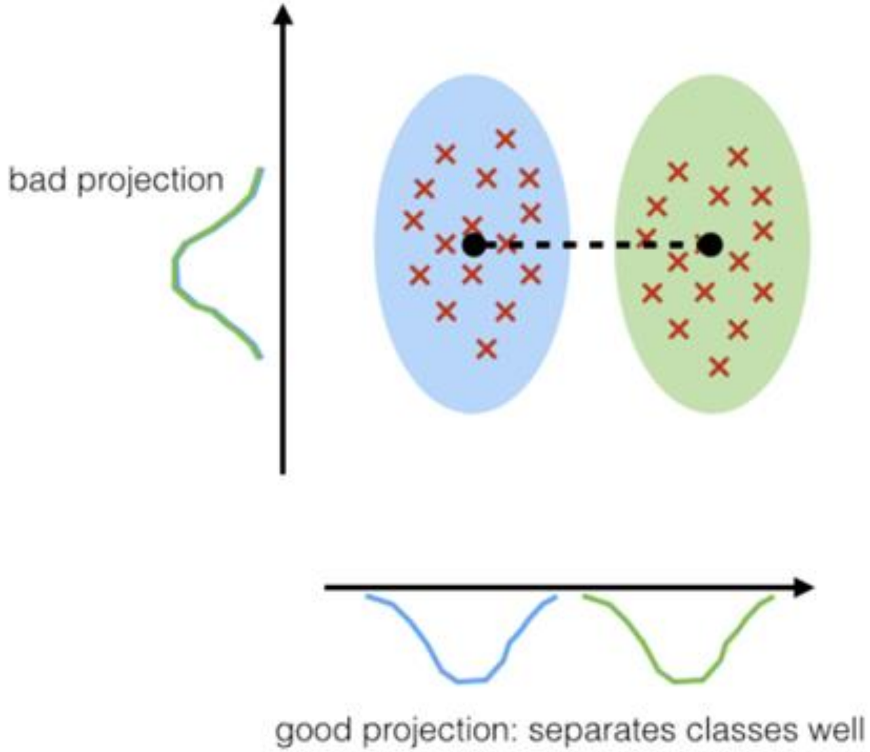
PCA

component axes that maximize the variance

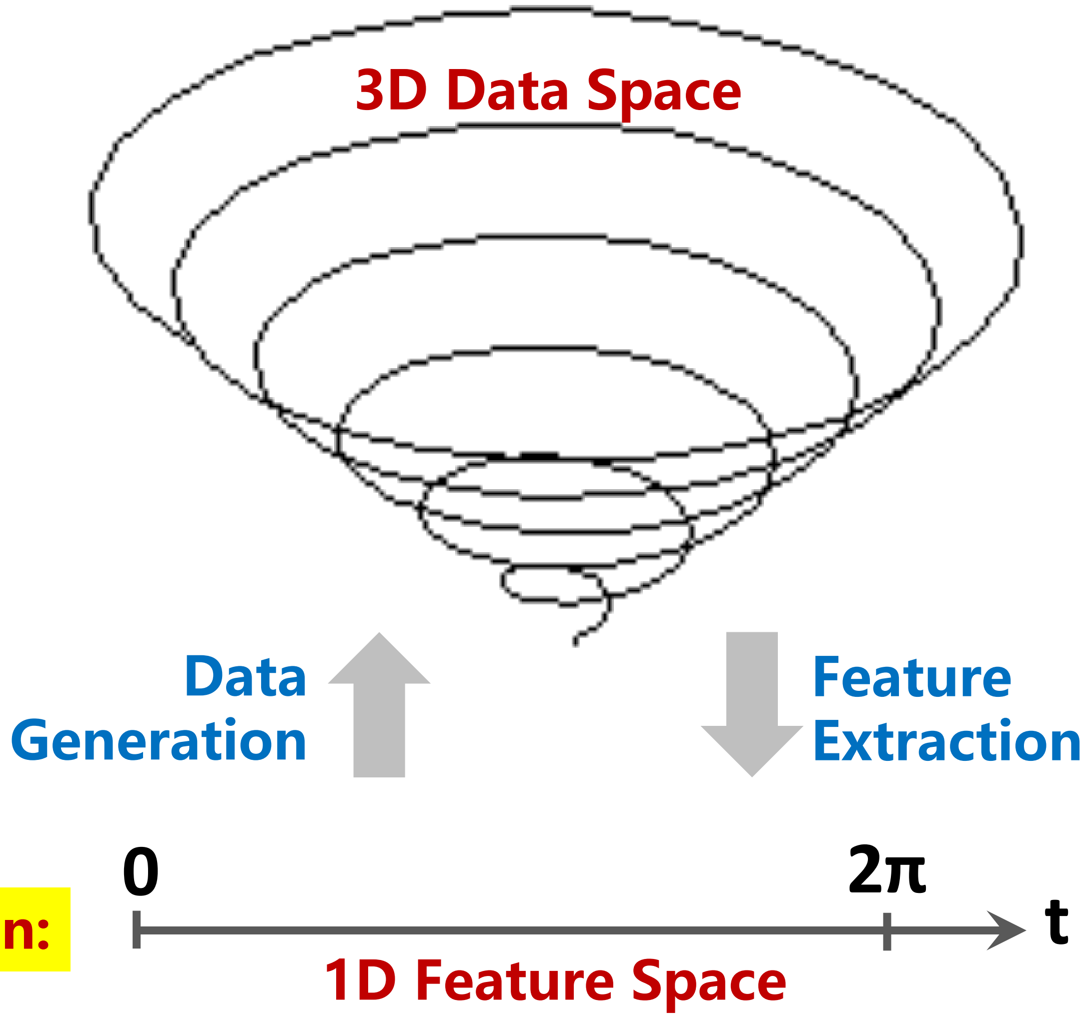


LDA

maximizing the component axes for class-separation

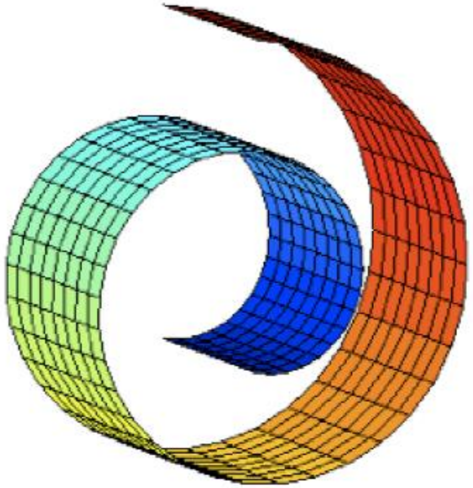


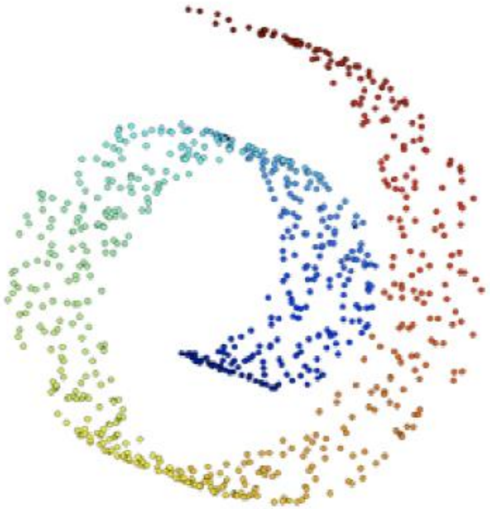
Nonlinear: 1D Manifold in 3D Space



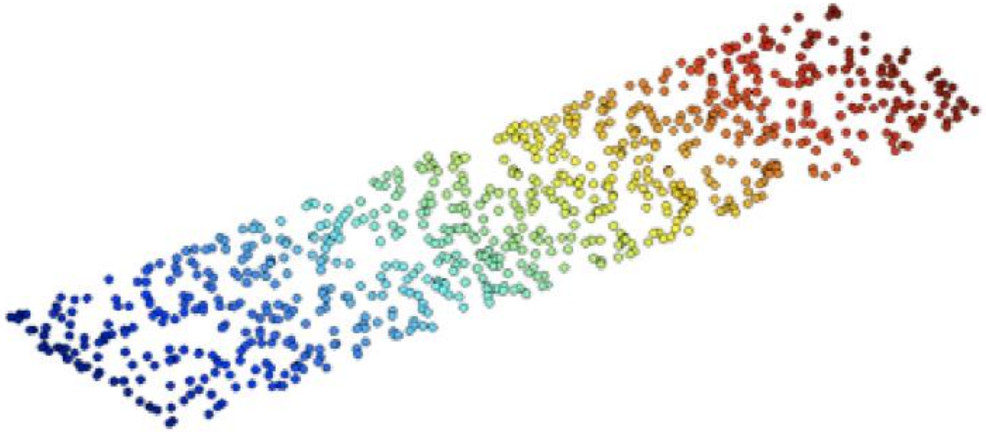
The Best Representation:

Nonlinear: 2D Manifold in 3D Space





The Best Representation:



Swiss Roll:

x=φcos(φ), y=φsin(φ), z=ψ

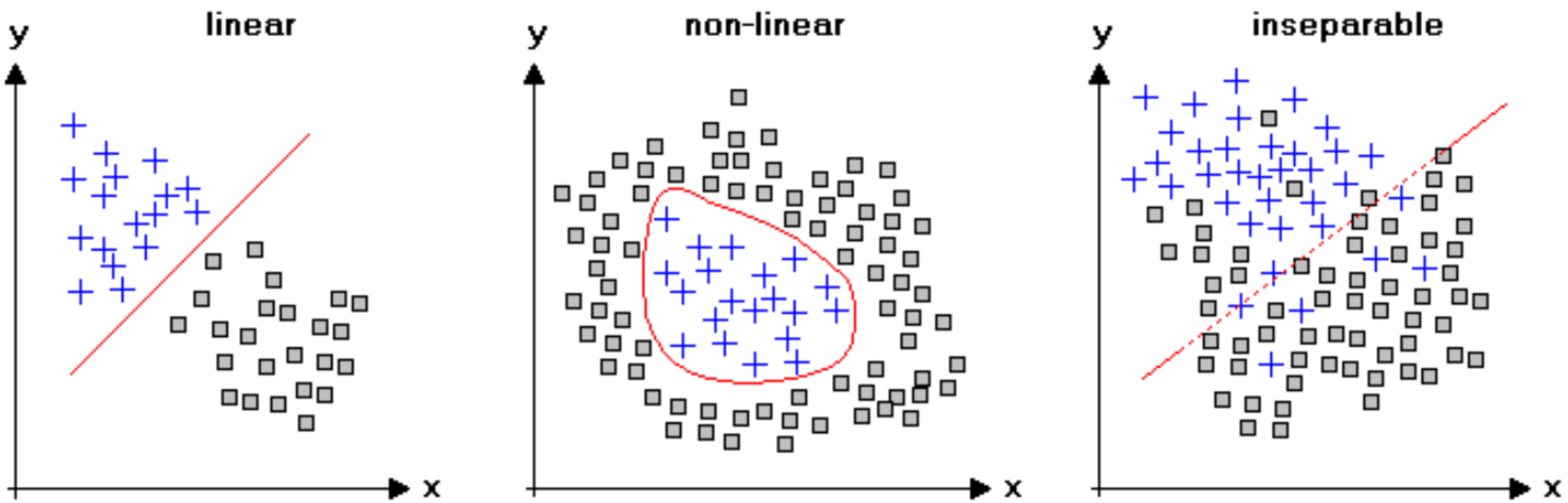
1.5π ≤ φ ≤ 4.5π, 0 ≤ ψ ≤ 10

Manifold: 2D rectangle

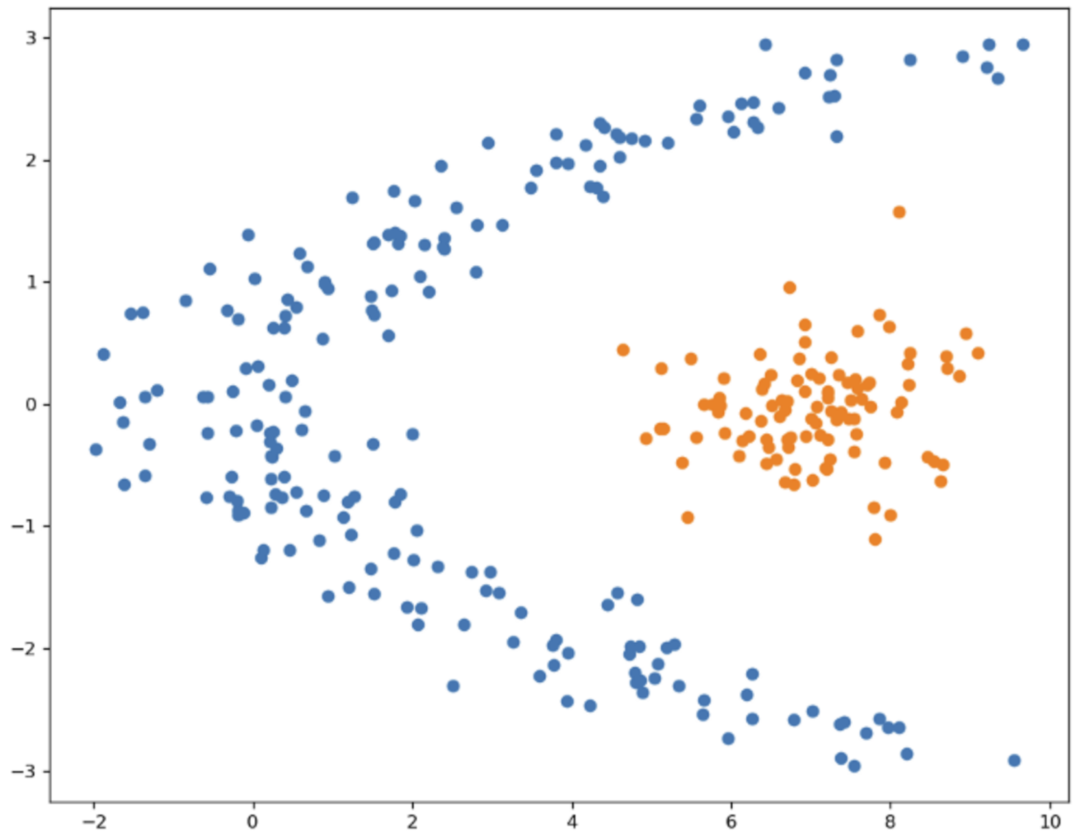
generated by two latent variables φ, ψ

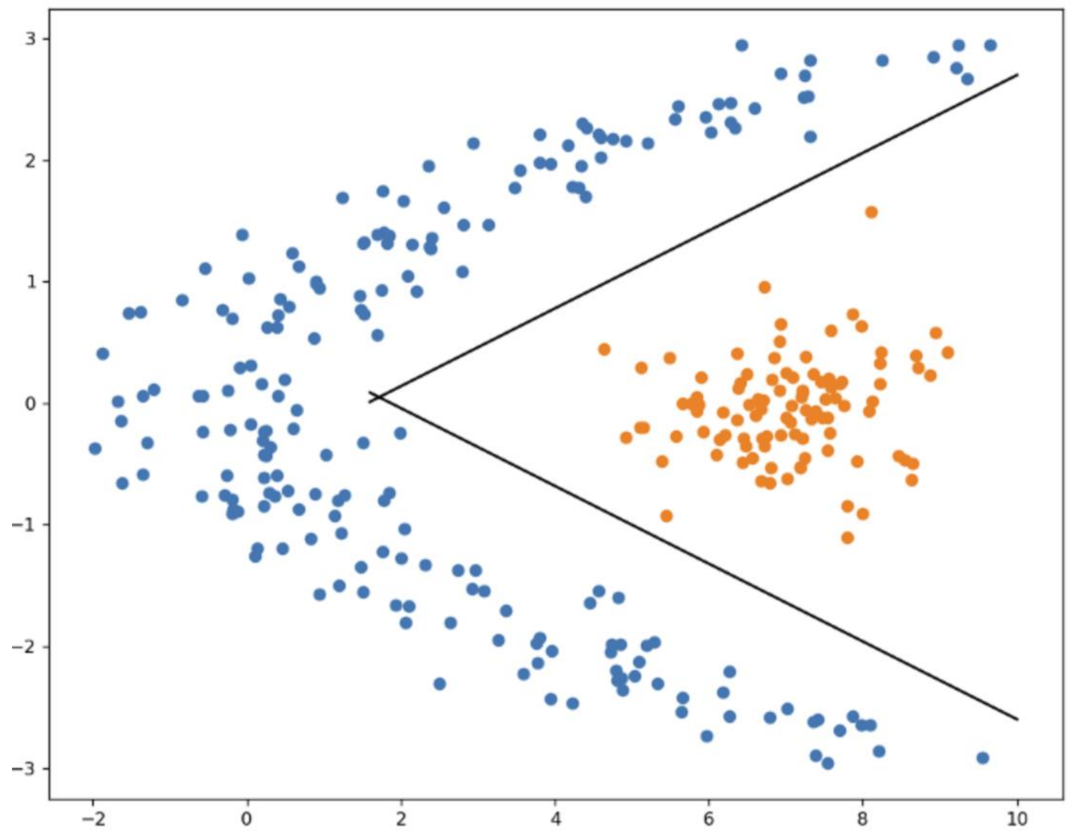
Separability in Classification Problems

- However, data samples are not always linearly separable, but may be **nonlinearly separable**



Nonlinearly Separable





{ (x1,x2) }

Transformation Function $y=f(x)$

- $f: X \to Y$ is a mapping from $x \in X$ to $y \in Y$
- x can be a scalar number, a vector $(x_1,\ldots,x_n)$, or a matrix $x_{i,j}$
- y can take value:
  - a real number (confidence, predicted stock value, etc),
  - a token value (decision, animal name, etc),
  - a vector (of confidences, 3D coordinates, etc)

What is a Function $f(x \mid w)$ Determined by?

Parameterized Function: $f(x \mid w)$ parameterized by w

Example: $y = f(x \mid \omega) = \sin(\omega x)$ with the form of sine and parameter ω

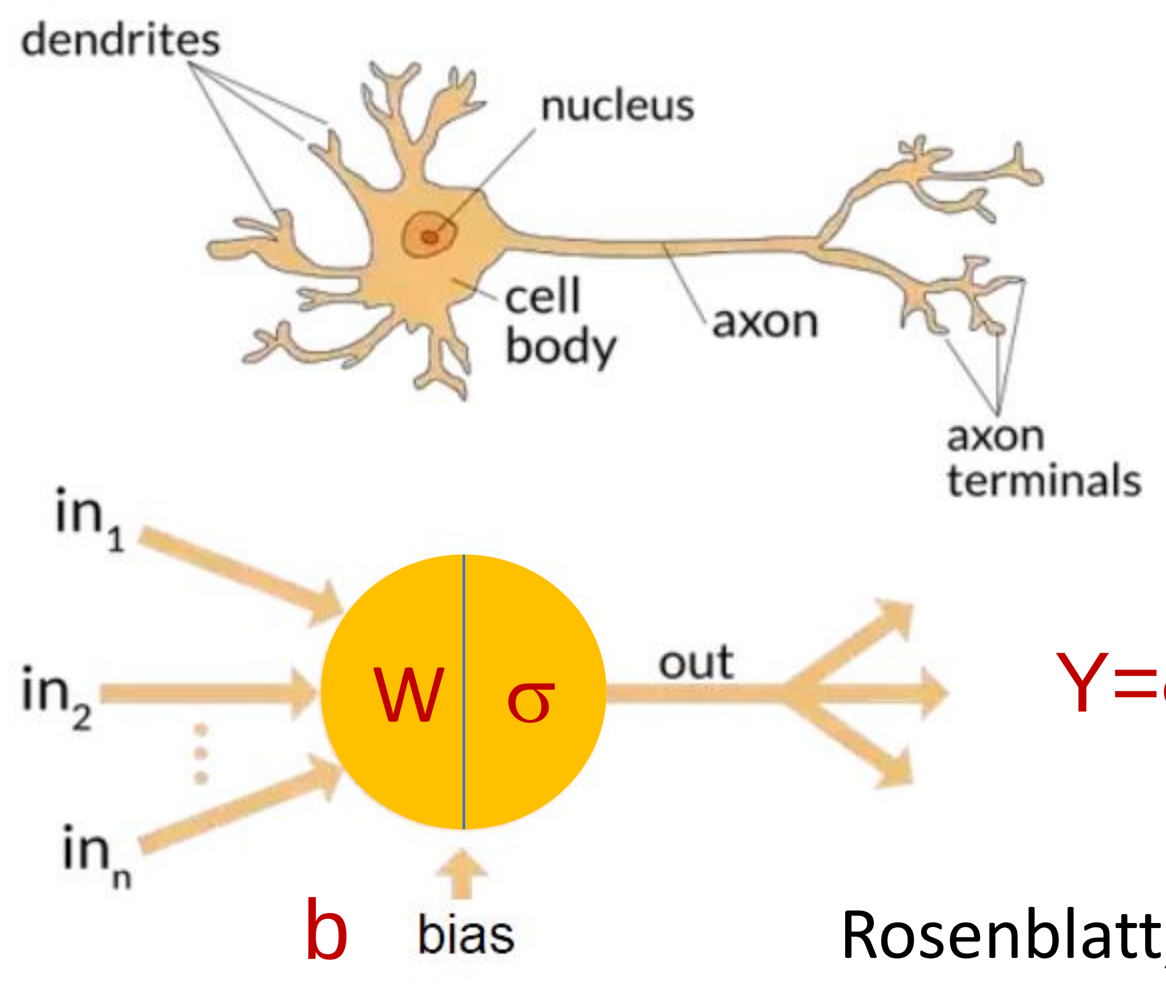
A function is determined by

1. **Functional form *f***

   → neural network structure, nonlinear activation, etc

2. **Parameters w in *f***

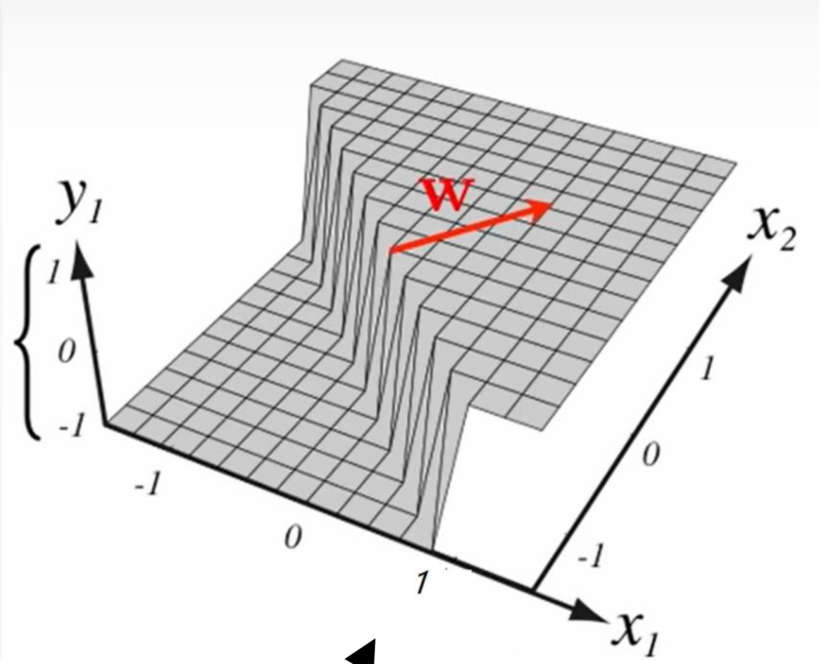
Nonlinearity in Neurons: Biological vs Artificial



X

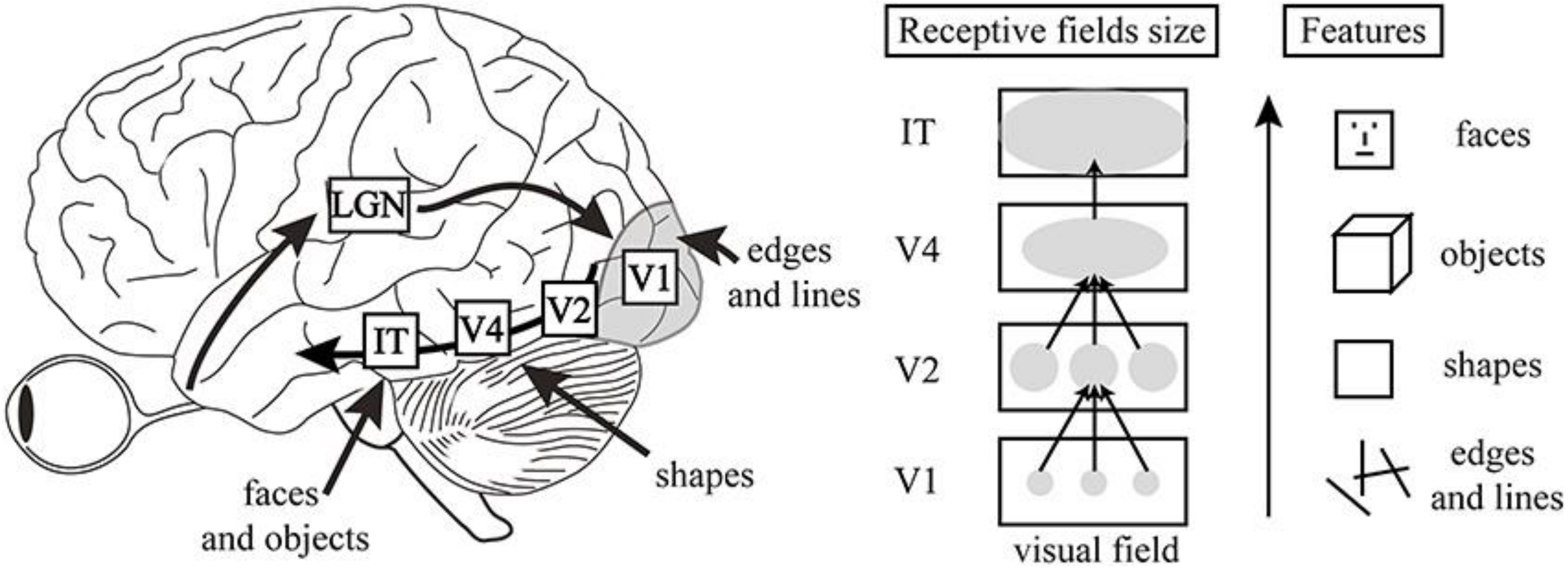
b

$Y=\sigma(XW+b)$



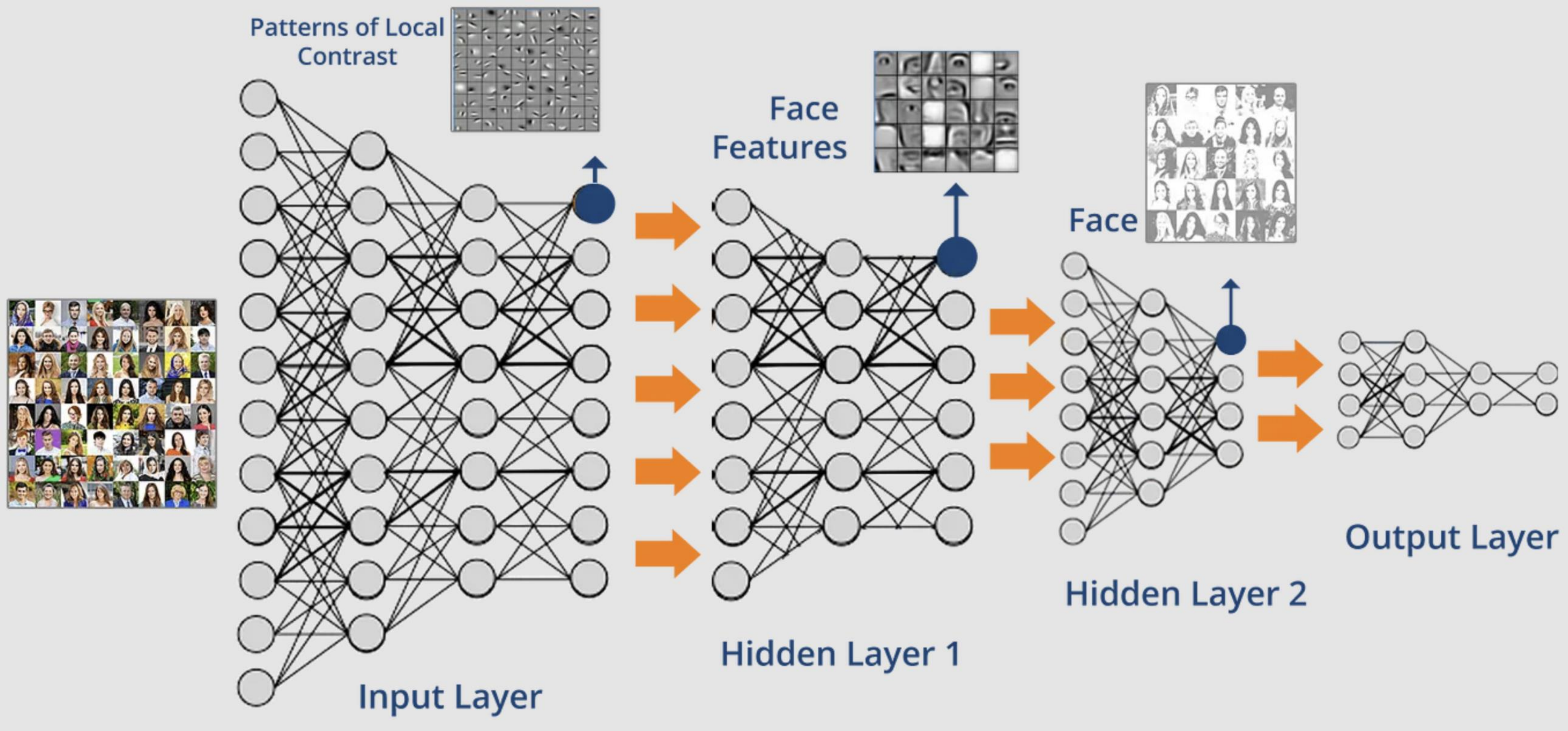
Rosenblatt, F. The Perceptron — Report 85–460–1, Cornell Aeronautical Laboratory. 1957.

Visual Neural Networks: Biological vs. Artificial

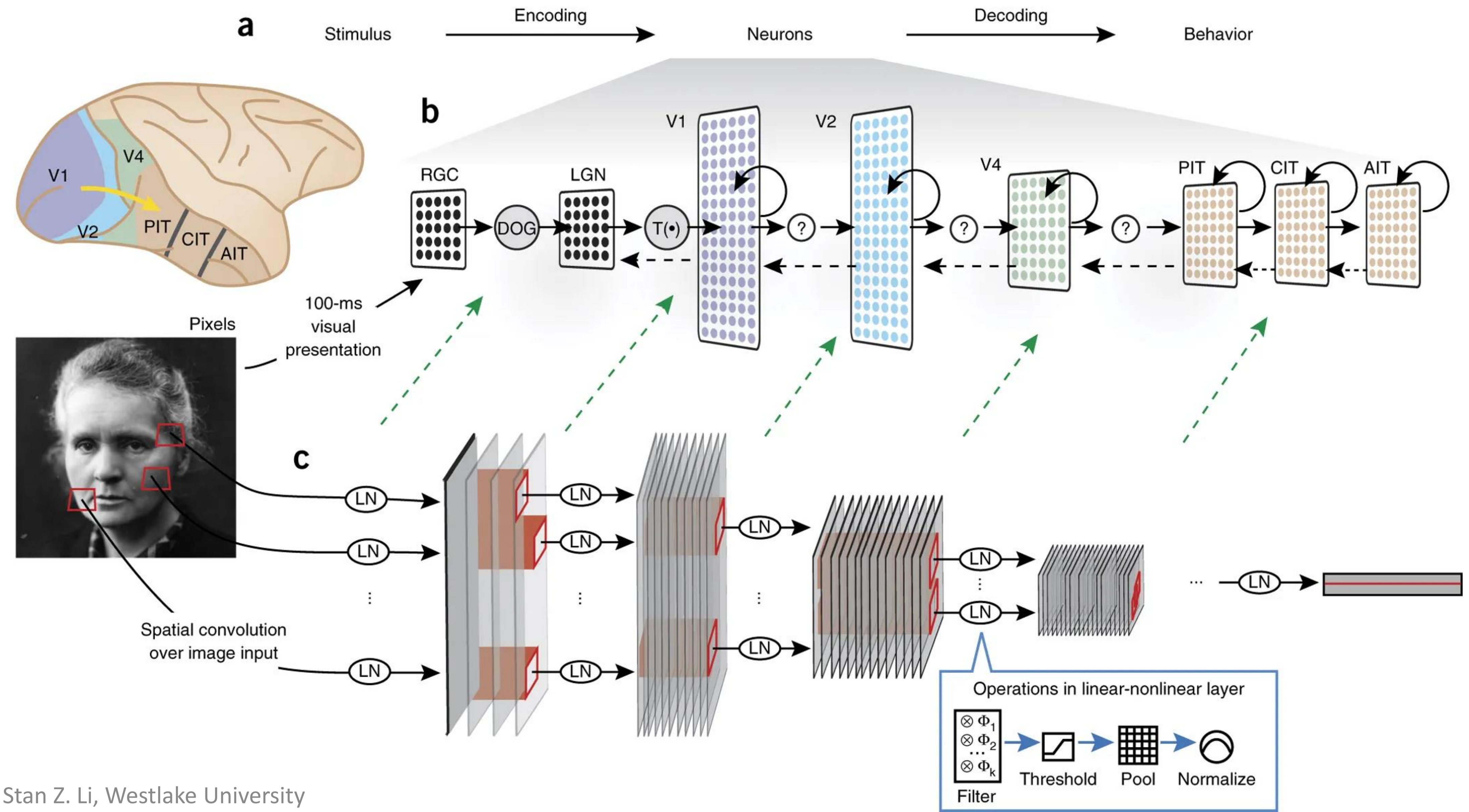


YouTube: Neural networks

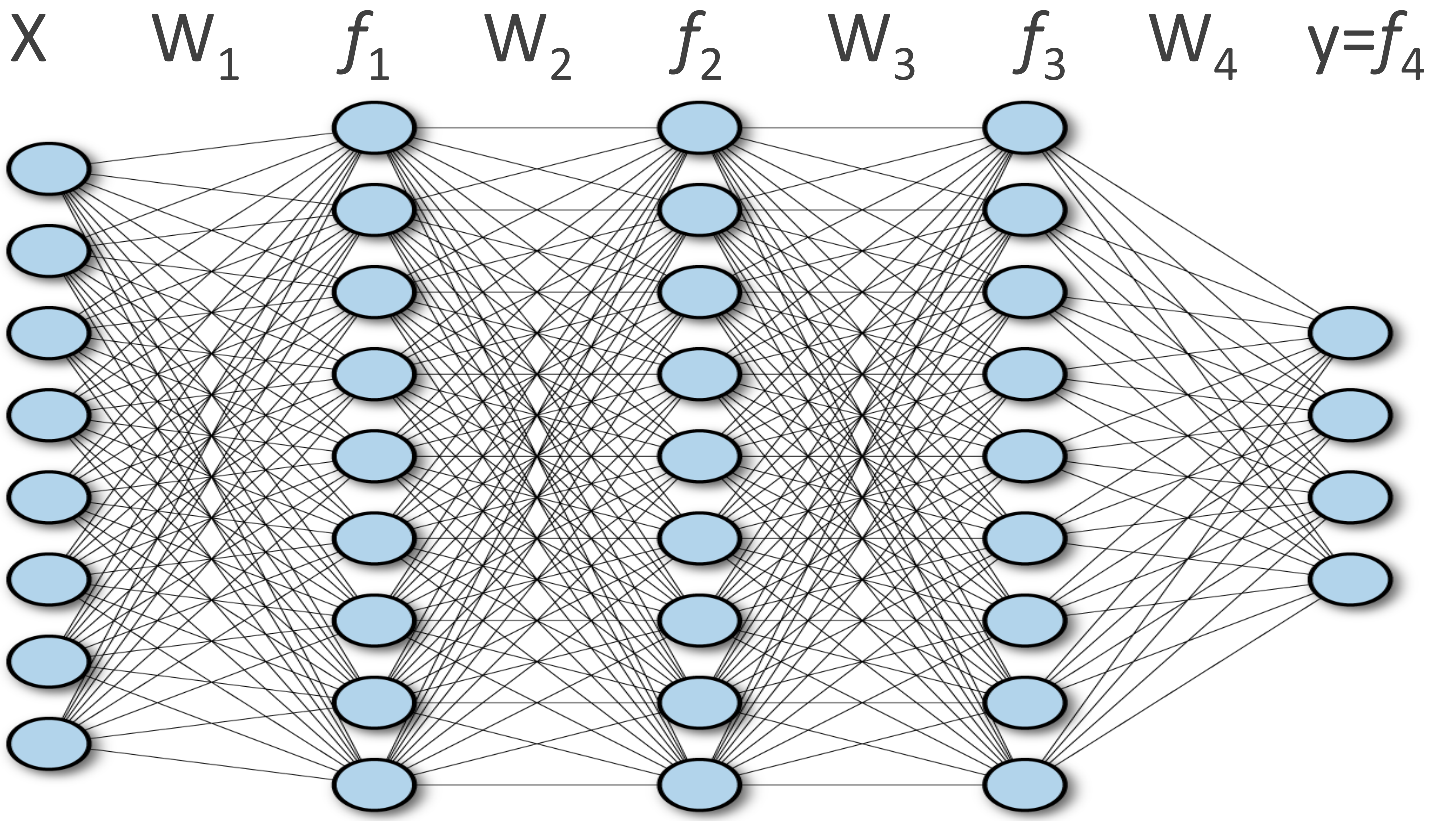
Learned Weights at Different Layers



Visual Neural Networks: Biological vs. Artificial



Multilayer Perceptron



$X_1 = f_1(x \mid w_1)$

$X_2 = f_2\,(f_1|w_2)$

......

$Y = f_4\,(f_3|\ w_4)$

$f_i\,(x \mid w_i\,) = \sigma(xw_i + b)$

Composite Function and Neural Network

- Composition of two functions $y=f(x)$ and $z=g(y)$

  $z=h(x) = g(f(x))$ is the composite function of f and g

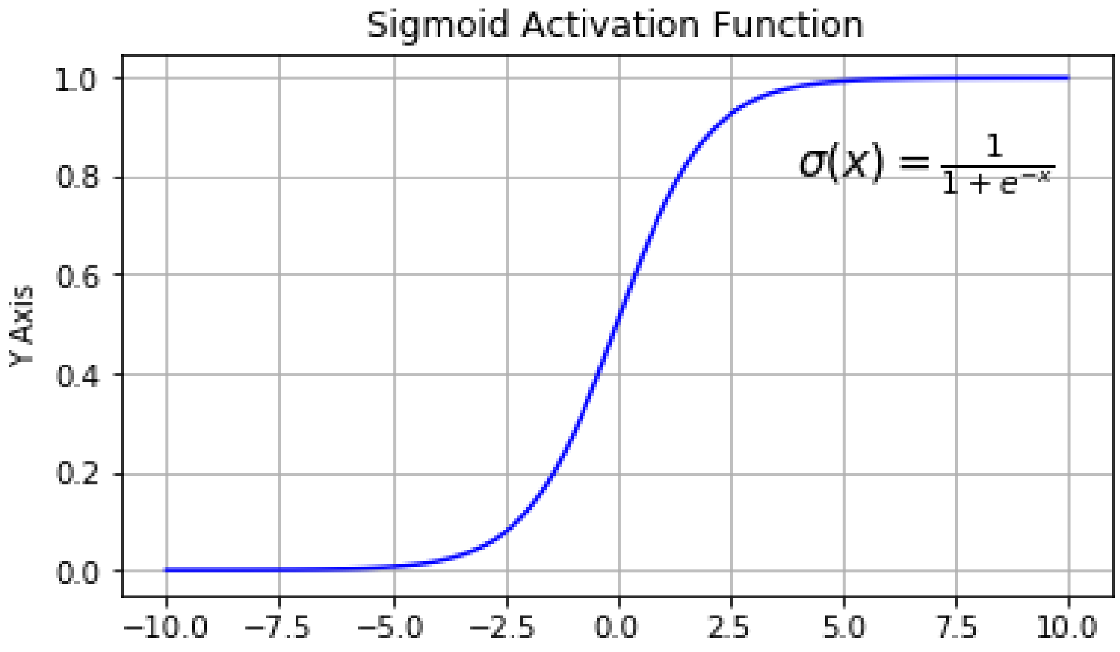
- Composite of K parametric functions

$$f_1(x \mid w_1)$$

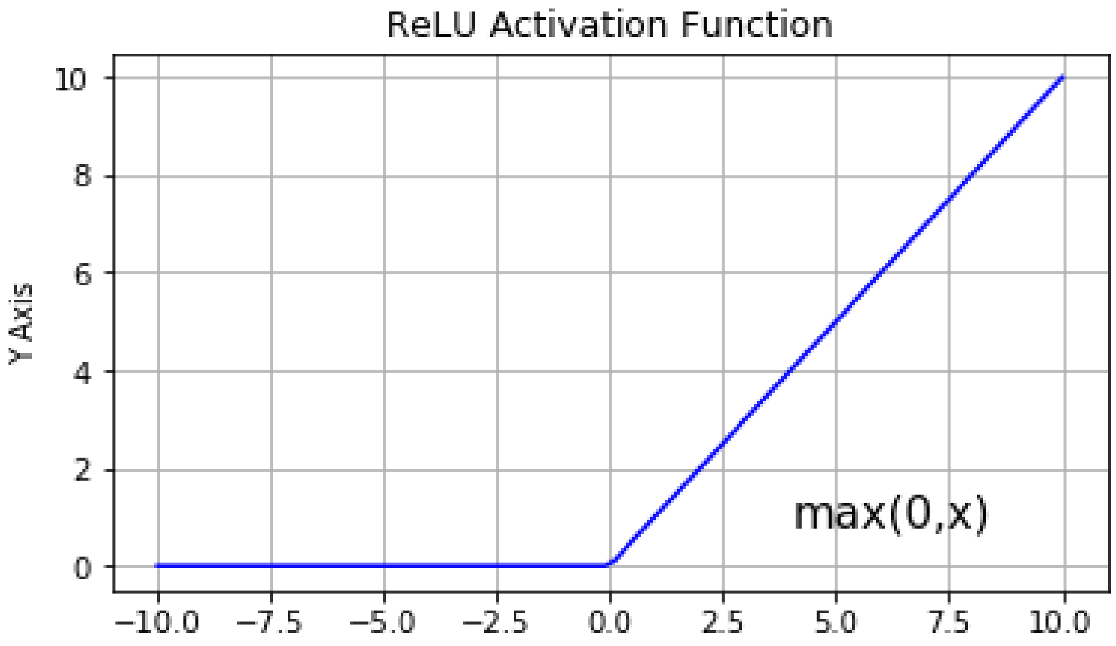
This is the form of a *K*-layer Neural Network

Overall $y = f(x \mid w)$ where $w=\{w_1, w_2, \ldots, w_K\}$

Activation Function to Achieve Nonlinearity



x= $\sum_{i=1}^{n} x_i w_i$



x= $\sum_{i=1}^{n} x_i w_i$

Supervised Learning of W

1. Design DNN structure, i.e. define the functional form f(x | w)
2. Design/define the loss function L(w | f , {(x,y)}), incorporating domain knowledge for solving the problem
3. Given a training set of $\{(x_i,y_i)\}_{i=1}^{N}$, each x_i with label y_i find best that minimizes the loss:

$$w^* = \arg\min_w L\,(w)$$

Minimizing Loss by Gradient Descent

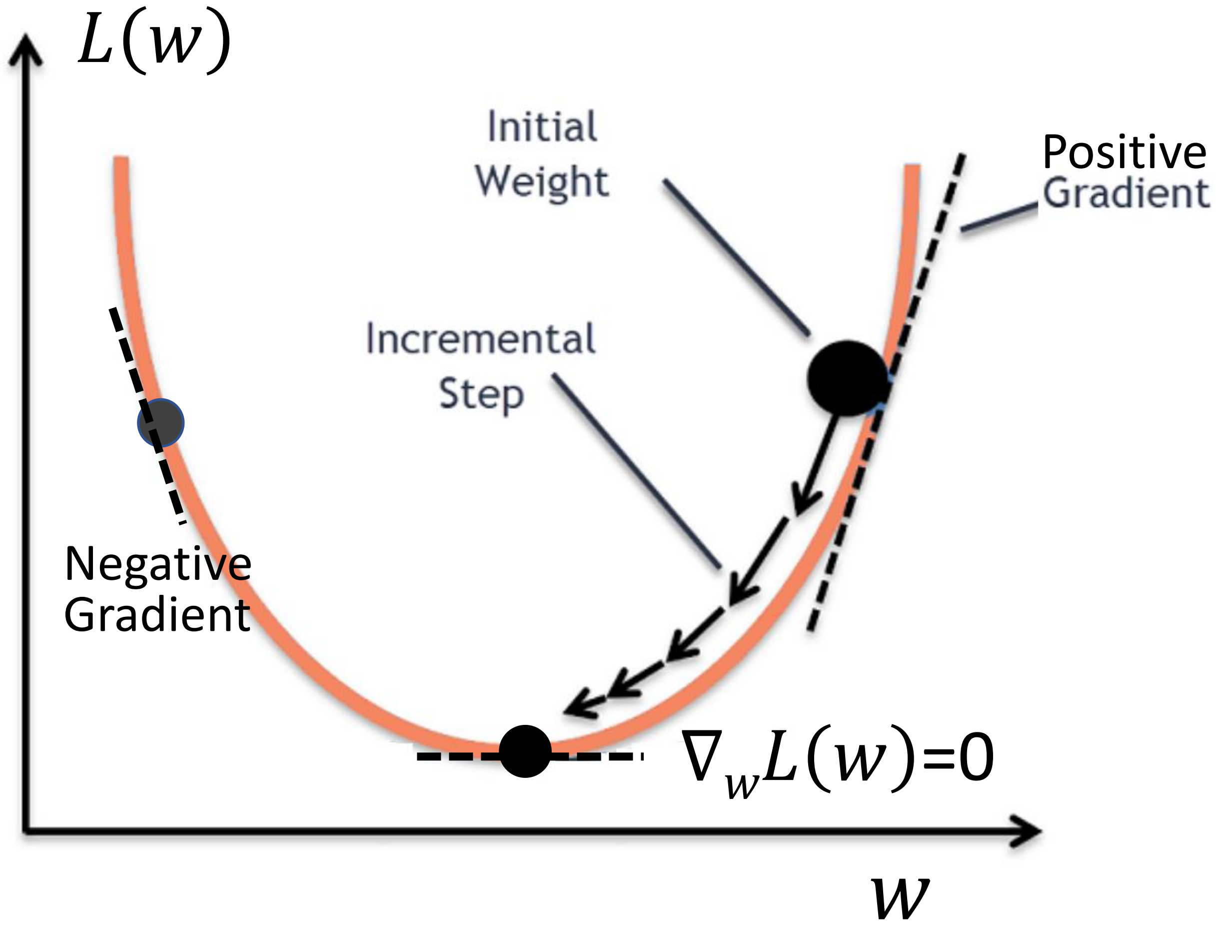
- Gradiant (using all $\{x_i\}$)

  $\nabla_w L(w) = \sum_{i=1}^{N} \nabla_w L_i(w)$

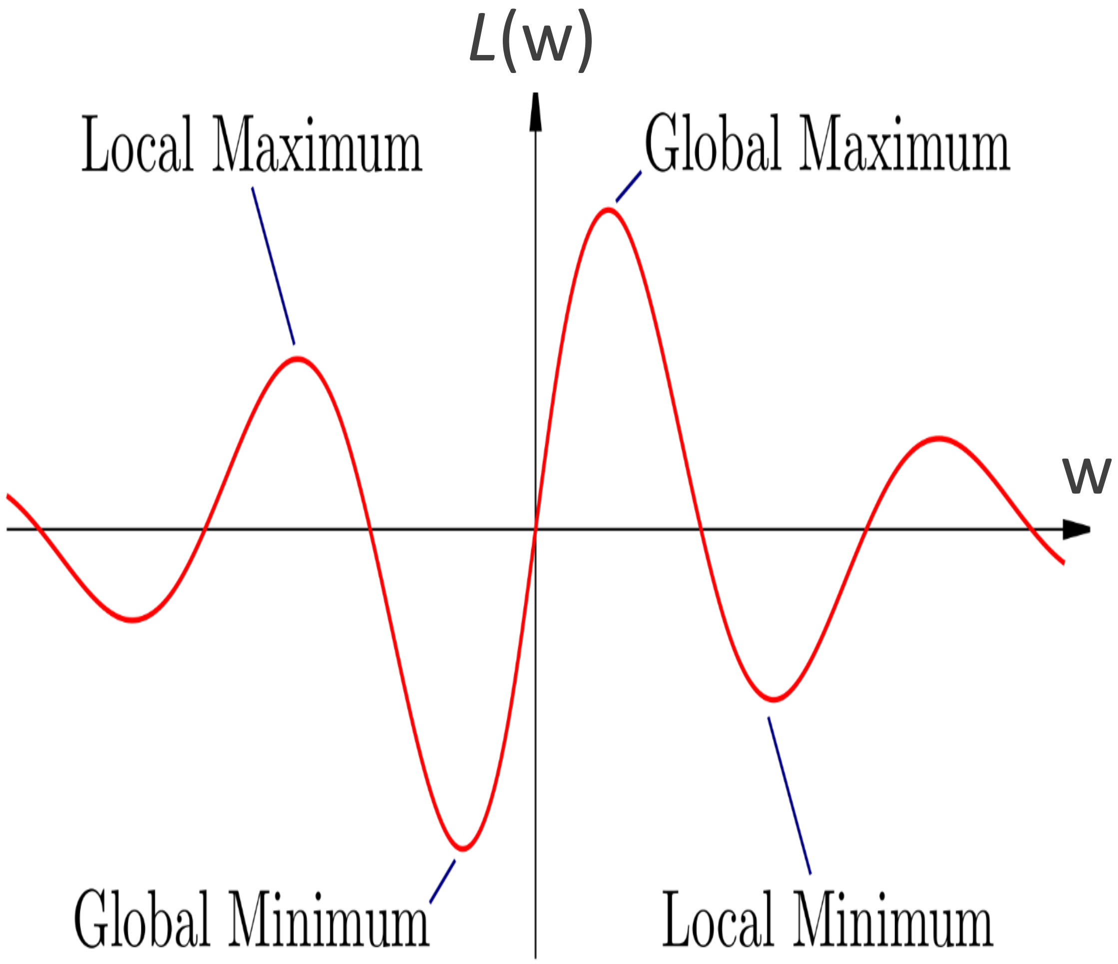
- Gradiant Descent

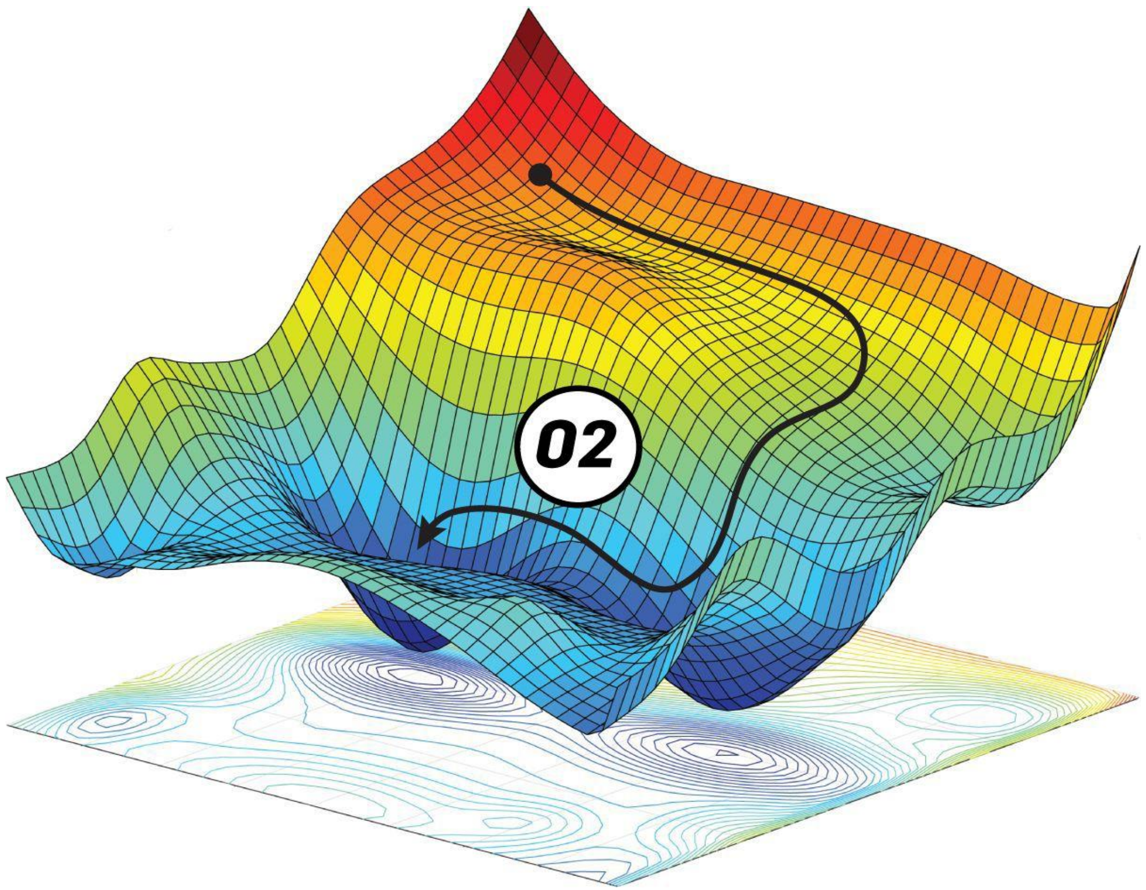
  $w^{(t+1)} \leftarrow w^{(t)} - \eta \nabla_w L(w)$

  $\eta > 0$: learning rate (step size)



Global vs Local Minima





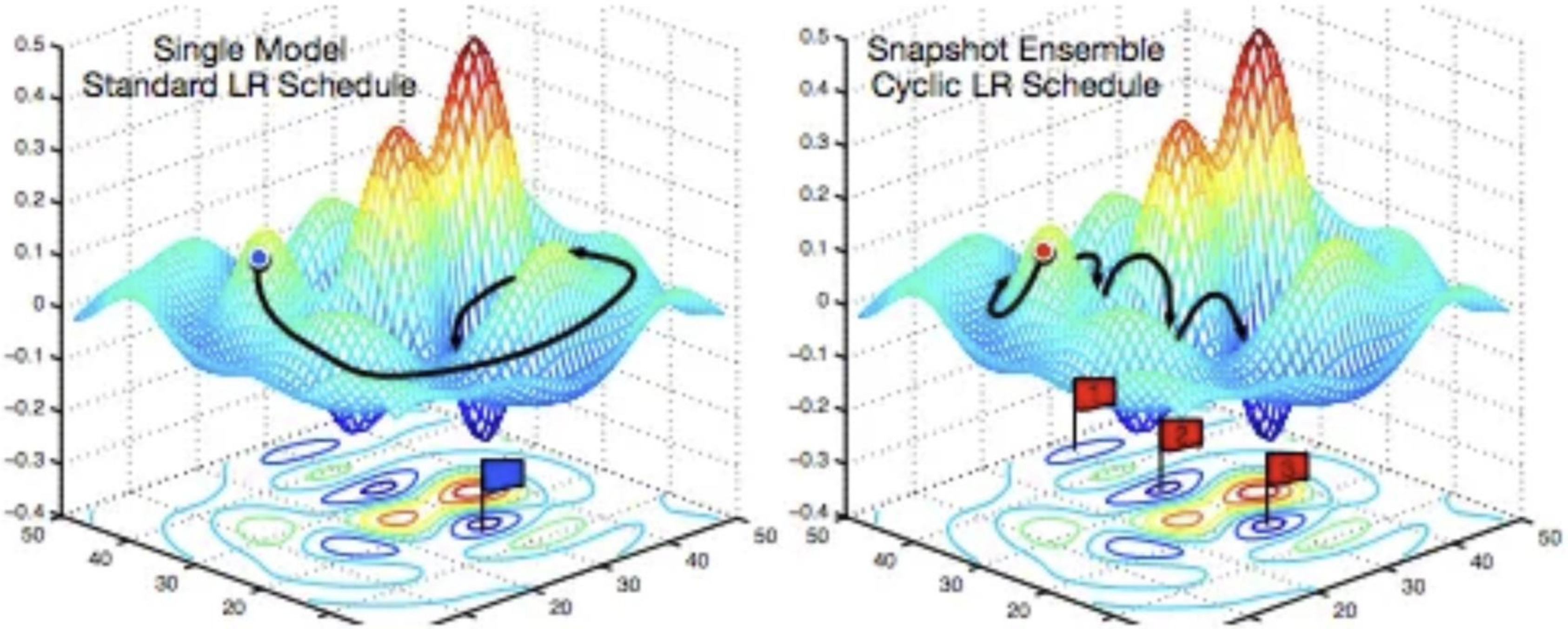
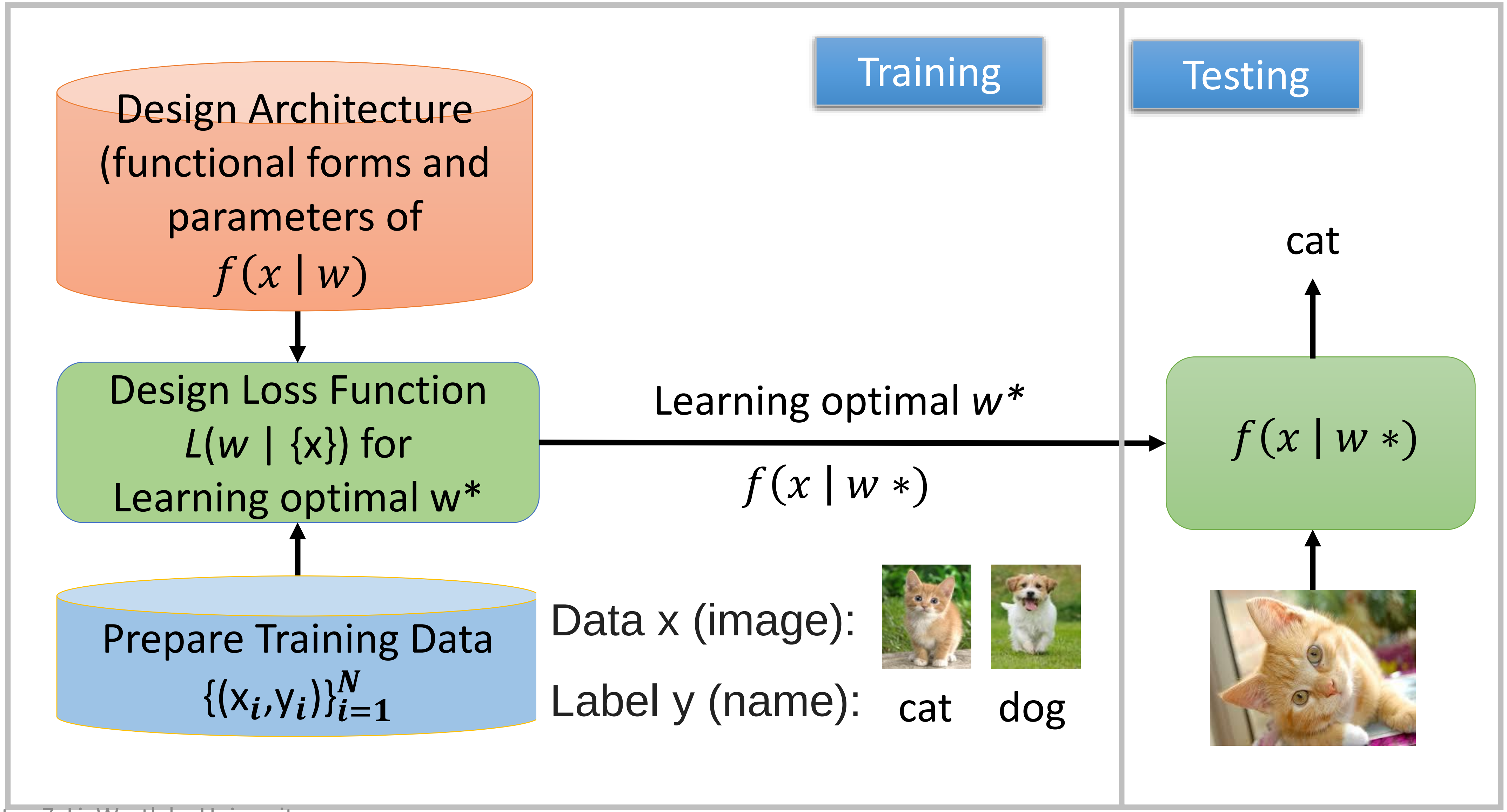
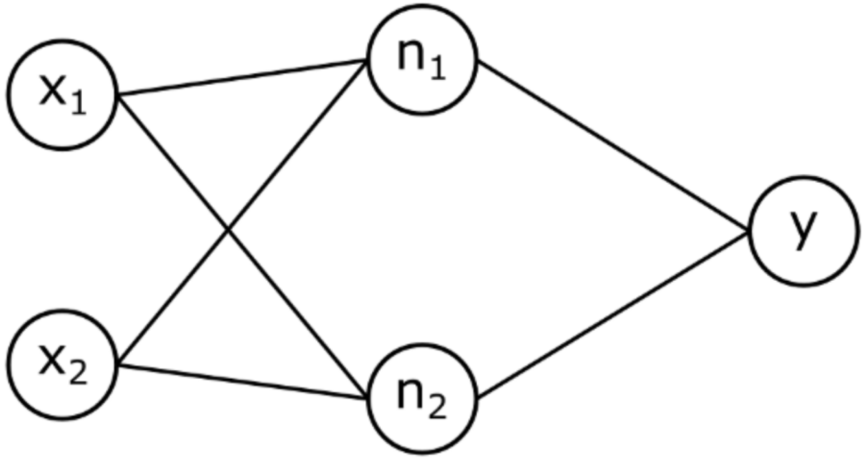


Figure : **Left:** Illustration of SGD optimization with a typical learning rate schedule. The model converges to a minimum at the end of training. **Right:** Illustration of Snapshot Ensembling. The model undergoes several learning rate annealing cycles, converging to and escaping from multiple local minima. We take a snapshot at each minimum for test-time ensembling.

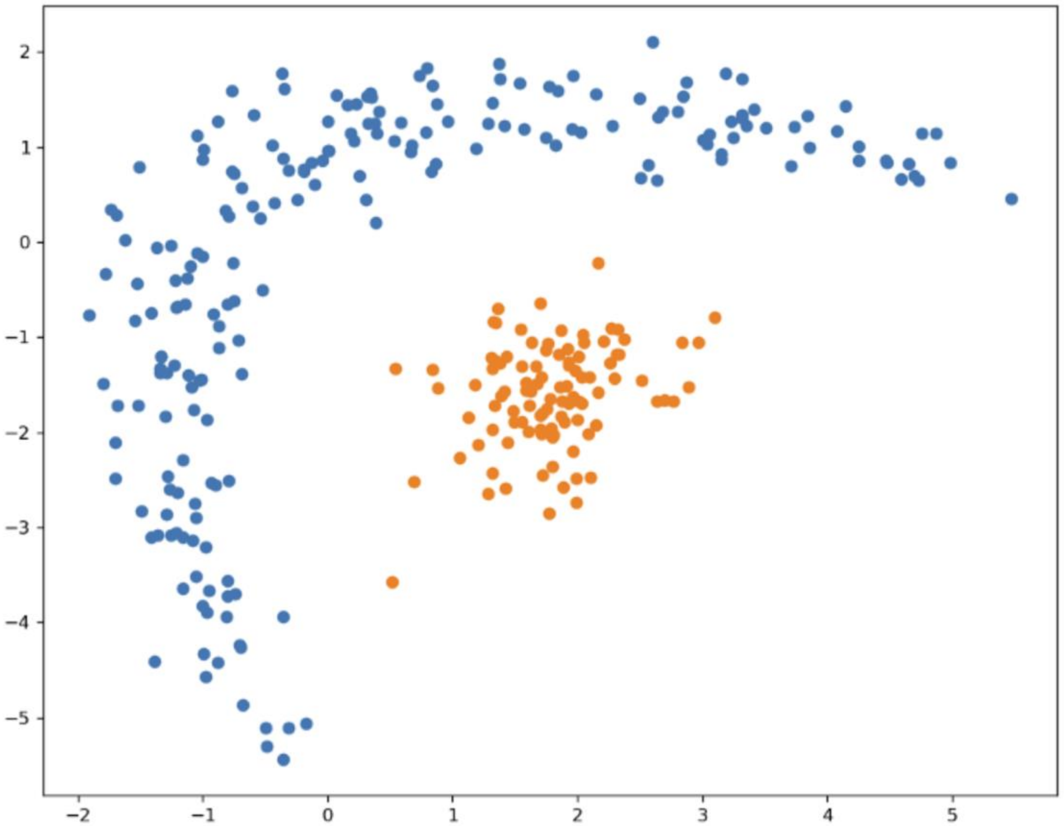
Summary: NN Training and Testing



Linear Transformation

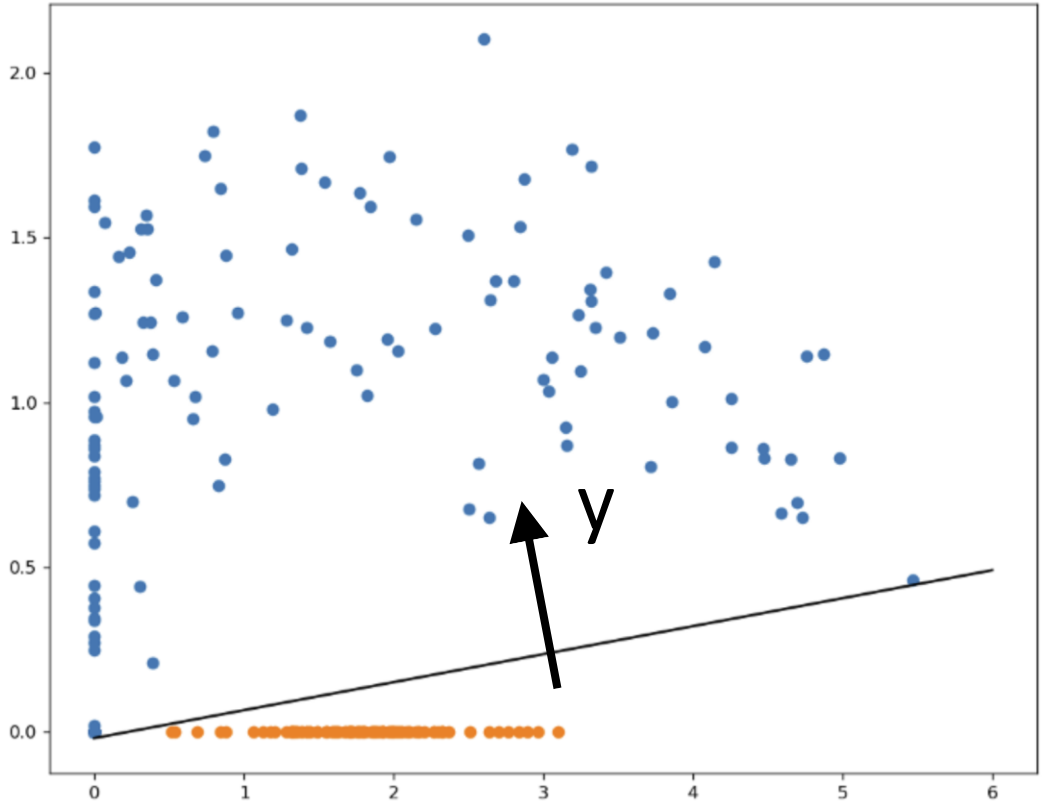


$$n_1 = 0.32 \cdot x_1 - x_2 - 0.5$$
$$n_2 = -0.32 \cdot x_1 - x_2 + 0.6$$

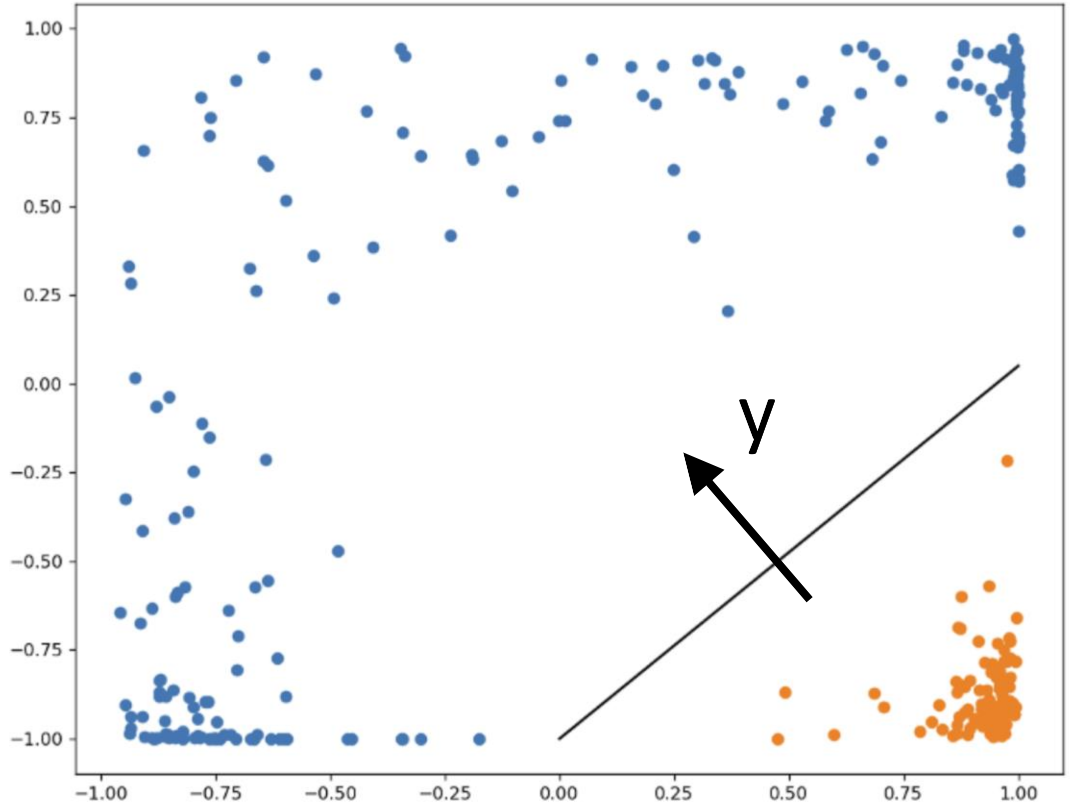


After f(n1) and f(n2) and recombine into y

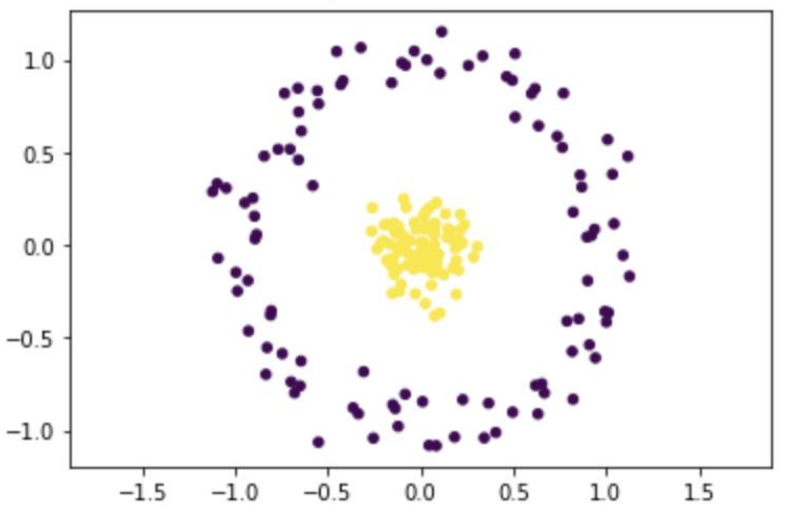
Using f=ReLU

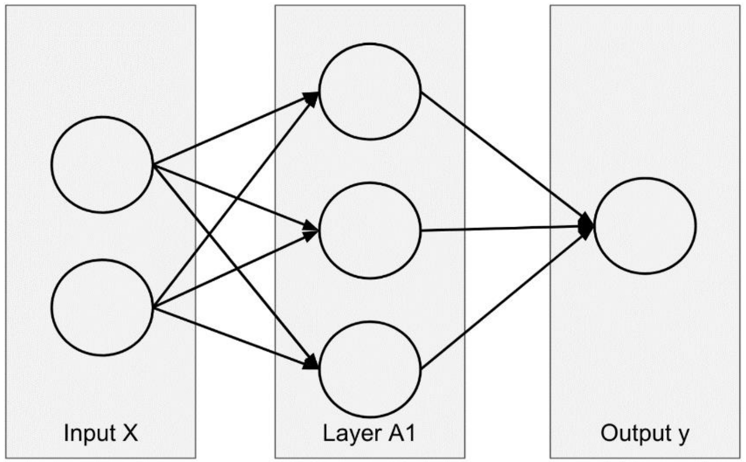


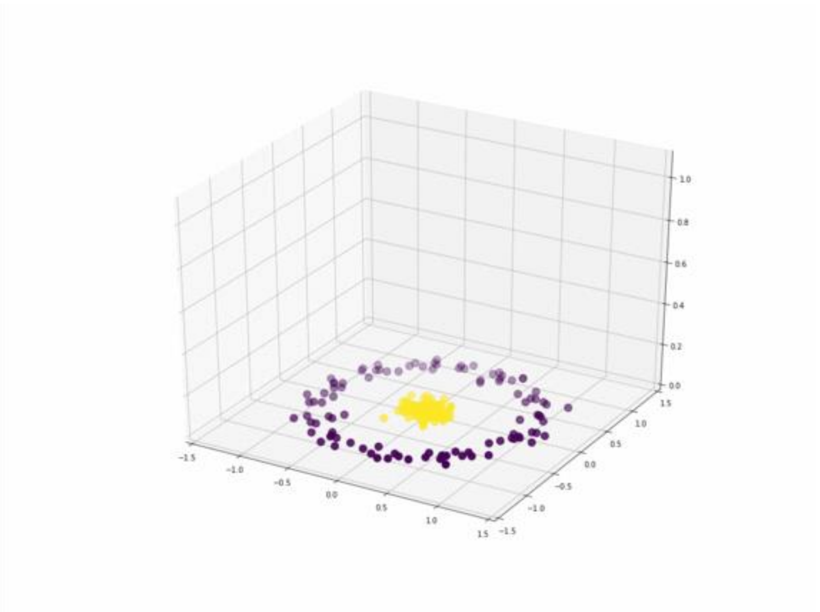
Using f=Tanh

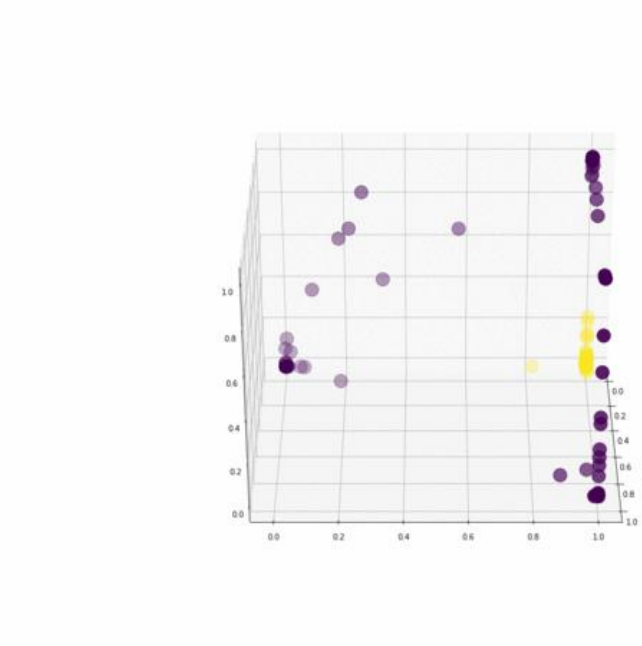


Learning W for Nonlinear Transformation



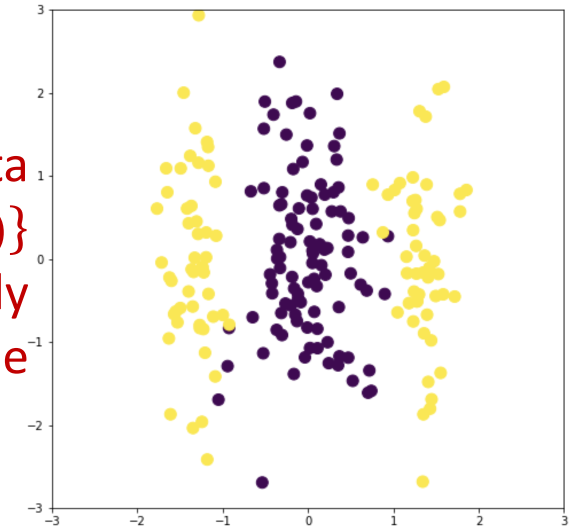




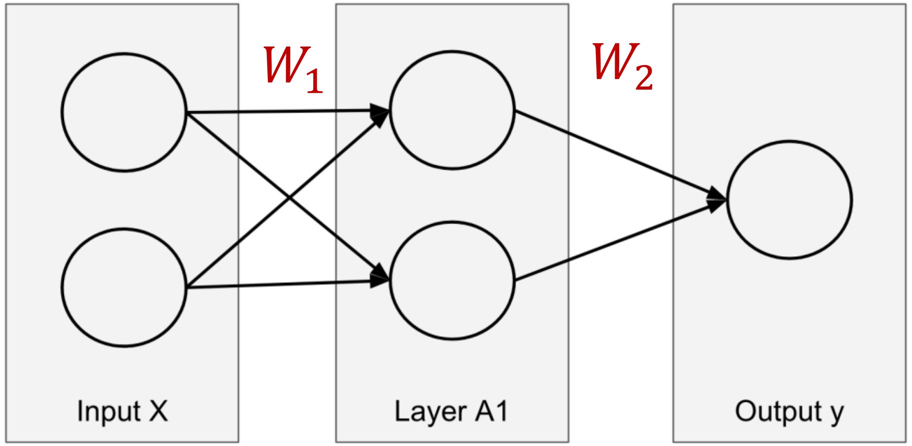


Learning W as Optimization

Training data $\{(X_i, y_i)\}$ not linearly separable

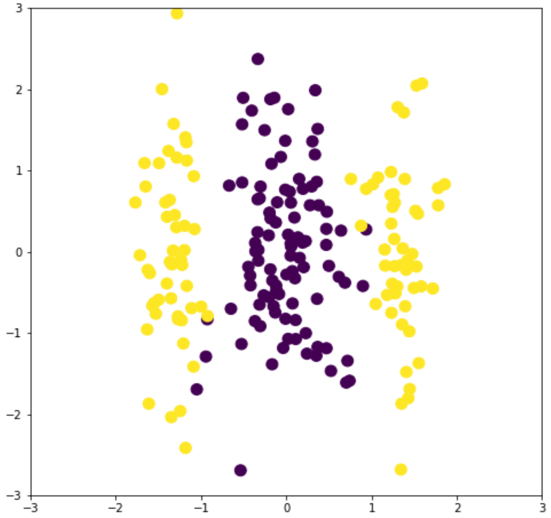


W_1 W_2

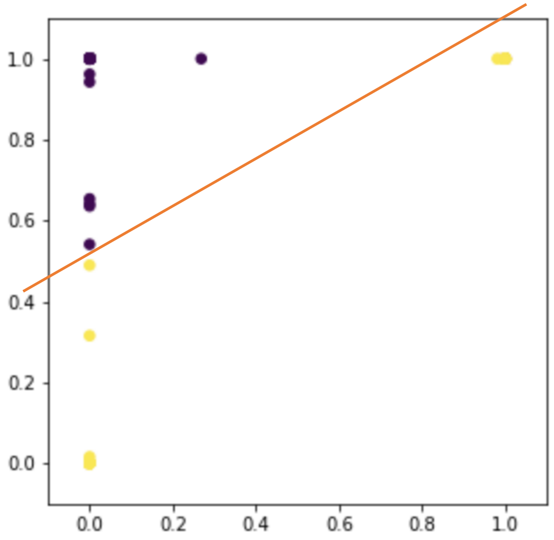


$$X_1 = \sigma(X\,W_1)$$
$$Y = X_1 W_2$$

$$L(W) = \sum_i \|y_i - X\,W_2\|^2$$



z_1



Thank You