

# Supervised Learning:

GOAL: LEARN A Predictor  $f_\theta: \mathcal{X} \rightarrow \mathcal{Y}$

WHERE  $x \in \mathcal{X}$  INPUTS

$y \in \mathcal{Y}$  TARGETS

$f_\theta$  is THE predictor WITH TRAINABLE WEIGHTS  $\theta \in \mathbb{R}^P$

INPUTS  $\mathcal{X}$ :

- IMAGES:  $\mathcal{X} = [0, 1]^{3n}$ ,  $n$  RGB pixels
- TIME SERIES:  $\mathcal{X} = [0, T]$
- TEXT
- GRAPHS

TARGETS  $\mathcal{Y}$ :

① CLASSIFICATION

$$\mathcal{Y} = \{0, 1, \dots, C\} \quad \text{CLASS LABELS}$$

② REGRESSION

$$\mathcal{Y} = \mathbb{R} \quad \text{or} \quad \mathcal{Y} = \mathbb{R}^n$$

## Risk Minimization:

$$R = E_{x,y} \left\{ \mathcal{L}(f_\theta(x), y) \right\} \quad \text{RISK}$$

where  $\mathcal{L}: Y \times Y \rightarrow \mathbb{R}^+$  is a loss function.

In practice, we only have access

$$\text{TO A SUPERVISED DATASET } \mathcal{D} = \left\{ (x_i, y_i) \right\}_{i=1, \dots, N}$$

$$\hat{R}(\theta) = \frac{1}{N} \sum_{i=1}^N \mathcal{L}(f_\theta(x_i), y_i) \quad \text{EMPIRICAL RISK}$$

Empirical Risk Minimization (ERM):

$$\hat{\theta} = \underset{\theta}{\operatorname{arg\,min}} \hat{R}(\theta)$$

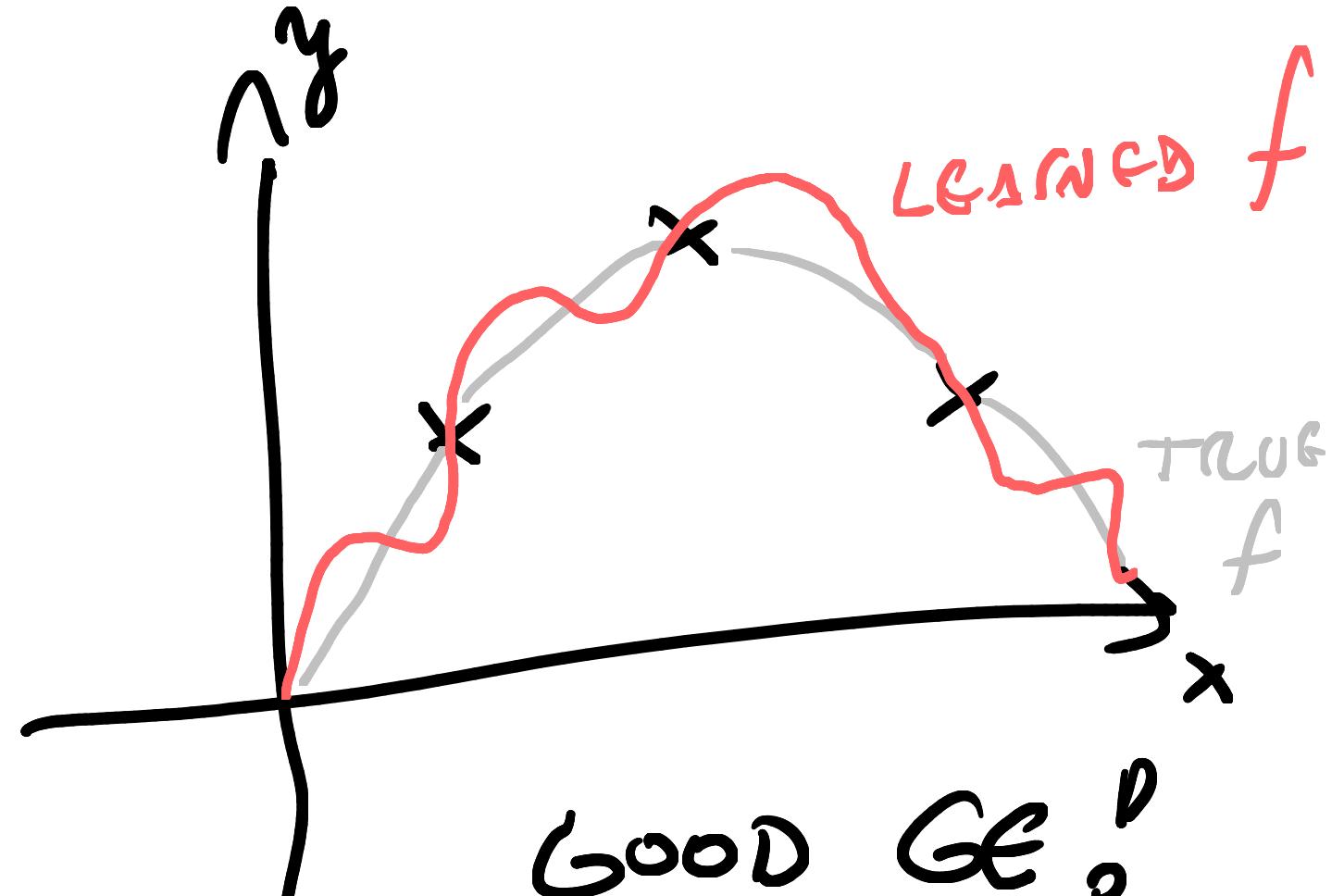
Idea: FIND THE PARAMETERS THAT MINIMIZE THE EMPIRICAL RISK

## IMPORTANT QUESTIONS:

- ① How do we choose  $\alpha$  for our task?
- ② How do we parameterize  $f_\theta(x)$ ?
- ③ How do we optimize the empirical risk?
- ④ Minimizing  $\hat{R}(\theta) = \text{minimizing } R(\theta)$ ?

## GENERALIZATION ERROR

$$GE = |R - \hat{R}|$$



NOTE THAT IN BOTH CASES WE HAVE  $\hat{R} = 0$  ZERO EMPIRICAL RISK

## optimization:

$$\hat{\theta} = \underset{\theta}{\operatorname{argmin}} \frac{1}{N} \sum_{i=1}^N \ell(f_{\theta}(x_i), y_i)$$

optimality condition (assuming differentiability)

$$\sum_{i=1}^N \frac{\partial \ell(\theta)}{\partial \theta} = 0$$

## GRADIENT DESCENT

$$\theta^{k+1} = \theta^k - \gamma \frac{\partial L(\theta^k)}{\partial \theta}$$

$\gamma$  step size or learning rate.

(more on this in Nelly's class)

# CHOOSING THE LOSS L

## ① BINARY CLASSIFICATION

- TWO CLASSES  $\gamma = \{-1, 1\}$

### 0-1 LOSS

$$L^{01}(\hat{y}, y) = \begin{cases} 1 & \text{if } \hat{y} \neq y \\ -1 & \text{if } \hat{y} = y \end{cases}$$

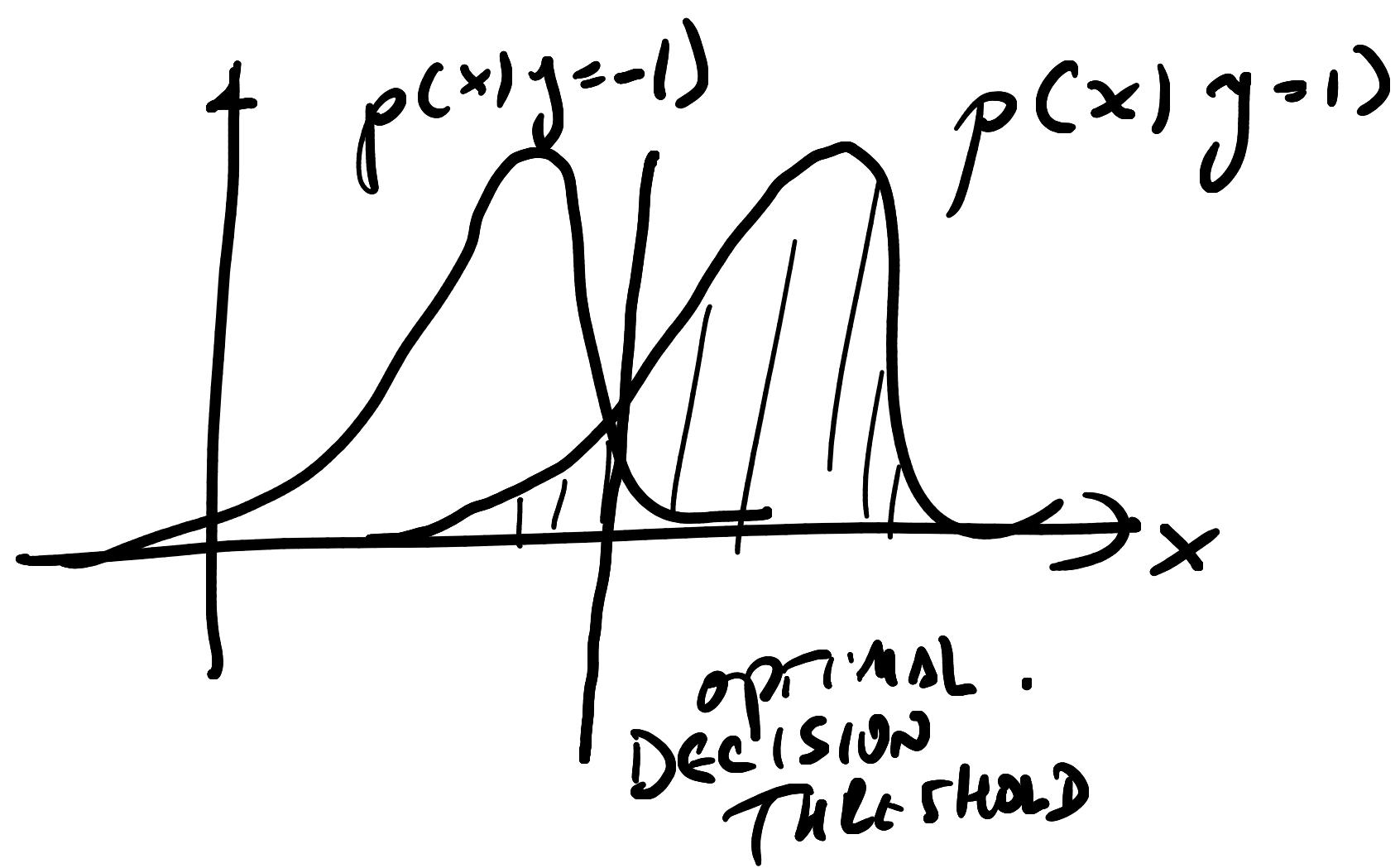
### THEOREM :

$$\hat{f} = \underset{f}{\operatorname{argmin}} \mathbb{E} \left\{ L^{01}(f(x), y) \right\}$$

$$\hat{f}(x) = \begin{cases} -1 & \text{if } P(y=-1/x) \geq P(y=1/x) \\ 1 & \text{if } P(y=-1/x) < P(y=1/x) \end{cases}$$

THUS WE NEED TO CHECK THE Ratio

$$\frac{P(y=-1/x)}{P(y=1/x)} = \underbrace{\frac{P(x/y=-1)}{P(x/y=1)}}_{\text{LIKELIHOOD RATIO TEST}} \cdot \frac{P(y=-1)}{P(y=1)}$$

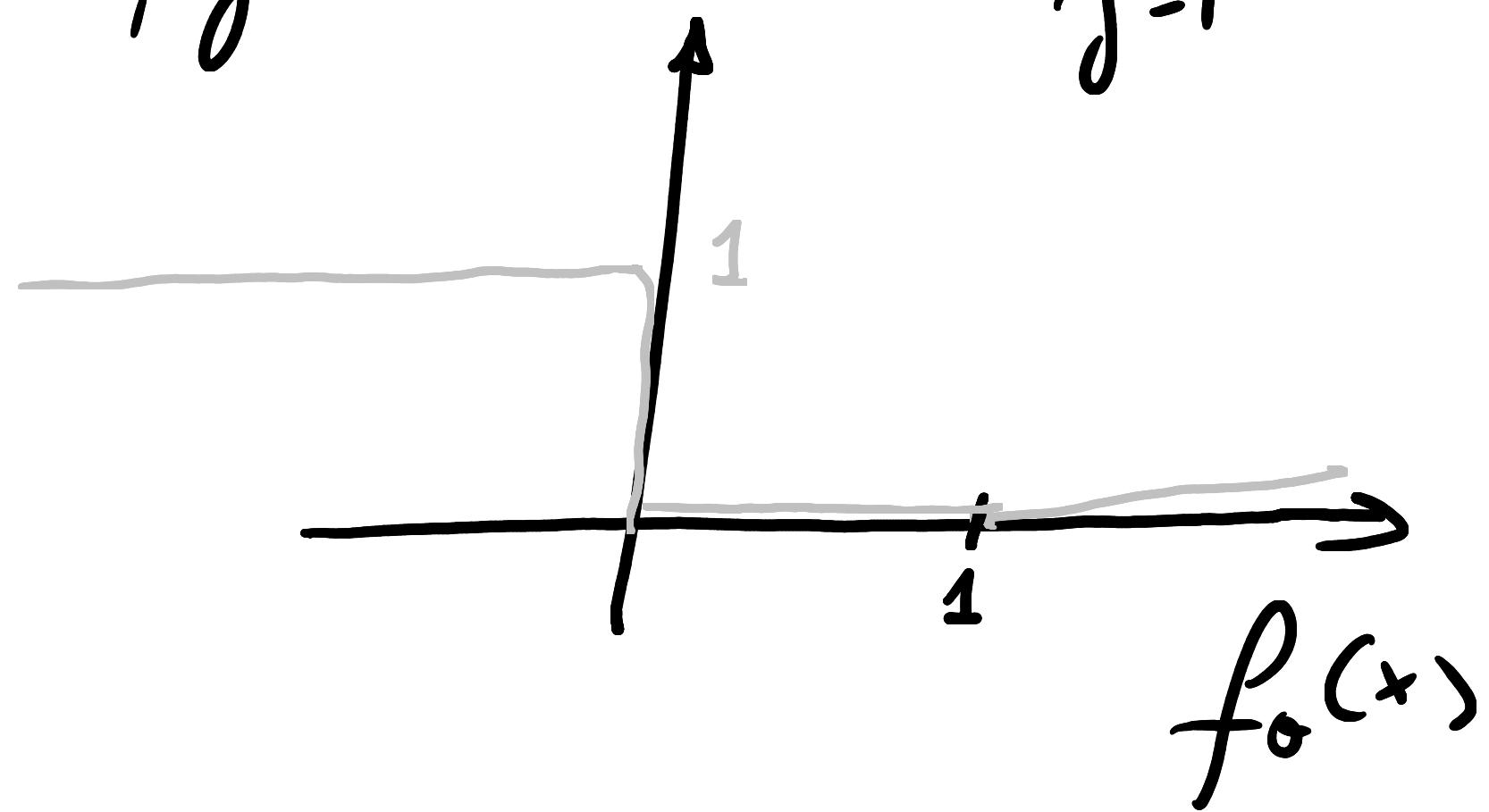


HOWEVER IN PRACTICE WE DON'T HAVE ACCESS  
TO  $p(x|y)$  ... ONLY A DATASET  $\{(x_i, y_i)\}$ .

IDEA : LEARN A FUNCTION  $f_0(x) : \mathcal{X} \rightarrow \mathbb{R}$   
AND CHOOSE CLASS AS  $\text{sign}(f_0(x))$ .

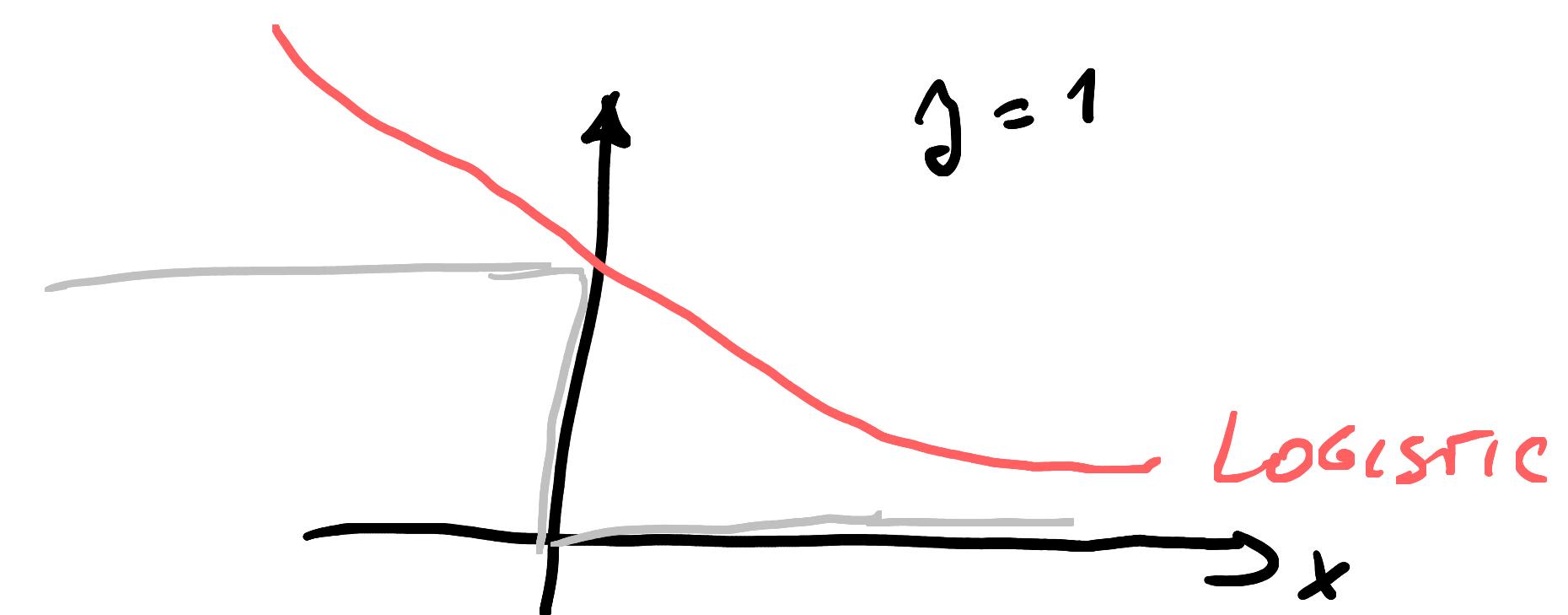
### 0-1 Loss

$$L(\text{sign } f_0(x), y)$$



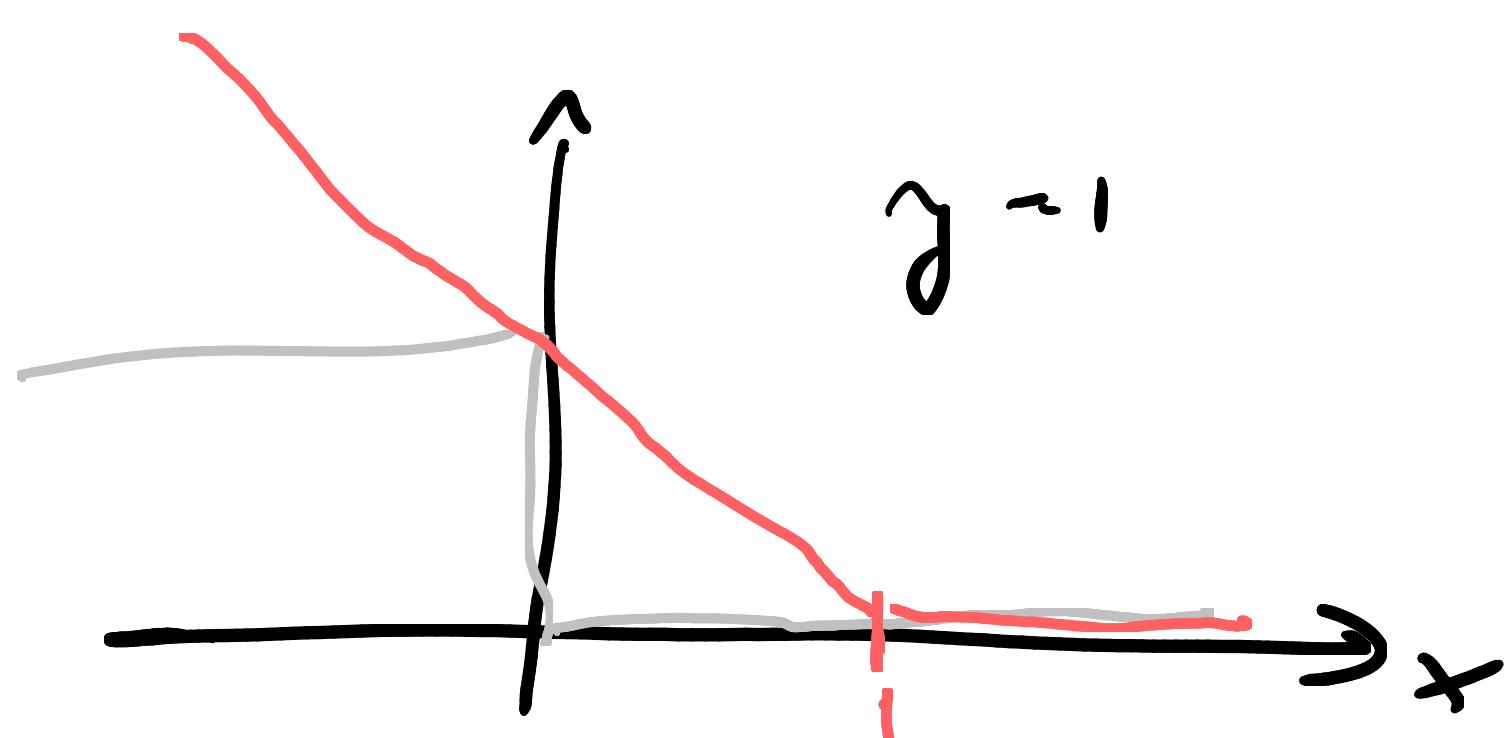
PROBLEM  $\Rightarrow$  THIS LOSS HAS  
ZERO GRADIENTS ALMOST  
EVERY WHERE ...  
 ↓  
SURROGATE LOSSES

$$\frac{\text{LOGISTIC}}{\text{LOSS}}: \quad L(f_0(x), y) = \log(1 + e^{-y f_0(x)})$$



Hinge Loss:

$$L(f_0(x), y) = \max(0, 1 - y f_0(x))$$



## ② REGRESSION

QUADRATIC LOSS

$$L(f_0(x), y) = \|f_0(x) - y\|^2$$

THEOREM: optimal predictor

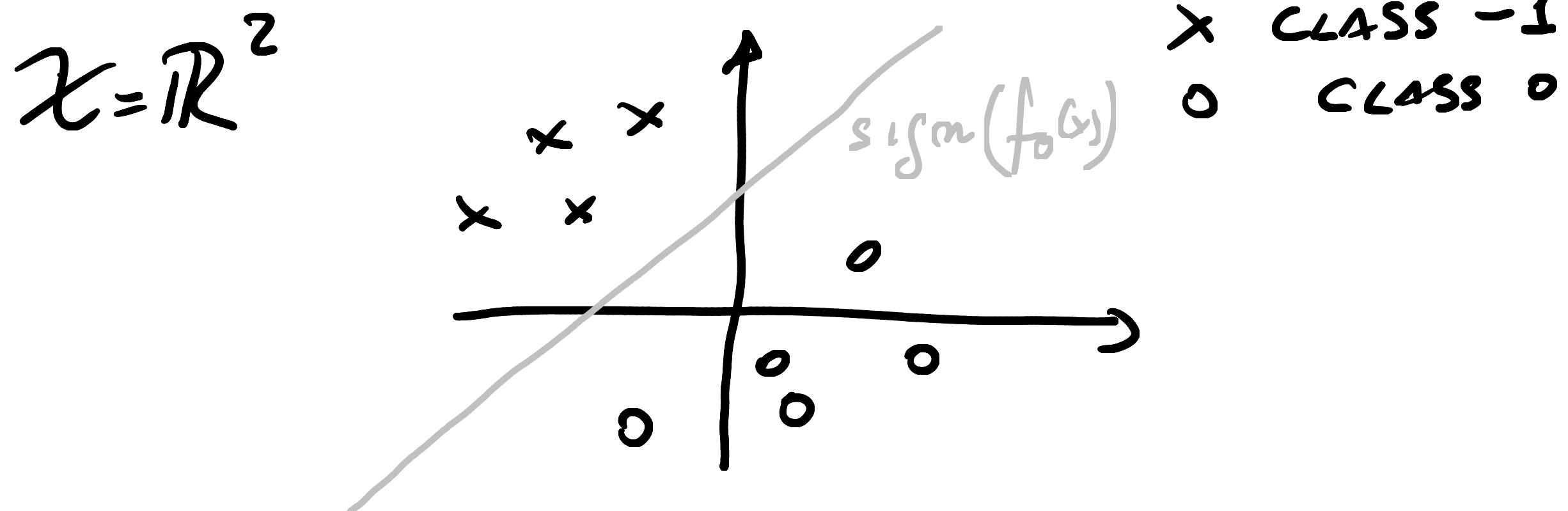
$$\hat{f} = \underset{f}{\operatorname{arg\min}} \mathbb{E}_{x,y} \left\{ \|f(x) - y\|^2 \right\}$$

$$\Rightarrow \hat{f}(x) = \mathbb{E}\{y|x\} \quad \begin{matrix} \text{KNOWN AS} \\ \text{"POSTERIOR" } \\ \text{MEAN"} \end{matrix}$$

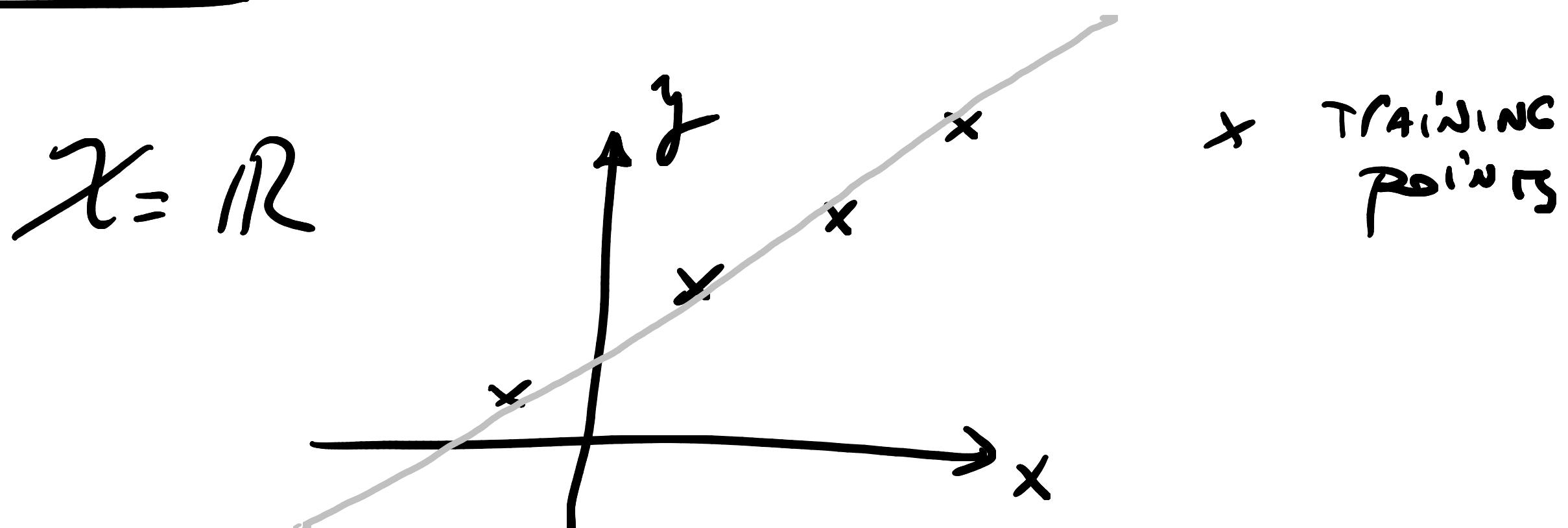
# Choosing the predictor $f_\theta$

LINEAR predictor  $f_\theta(x) = \theta_1^T x + \theta_2$   $\theta = \begin{bmatrix} \theta_1 \\ \theta_2 \end{bmatrix}$

CLASSIFICATION: LINEAR BOUNDARY  $\theta_1^T x > \theta_2$



REGRESSION: LINEAR function



Simple solution if  $L$  = quadratic loss

$$\hat{\theta} \underset{\theta}{\operatorname{arg\,min}} \sum_{i=1}^N \| \theta^T x_i - y_i \|^2 \Leftrightarrow \underset{\theta}{\operatorname{arg\,min}} \| X \theta - \bar{y} \|^2$$

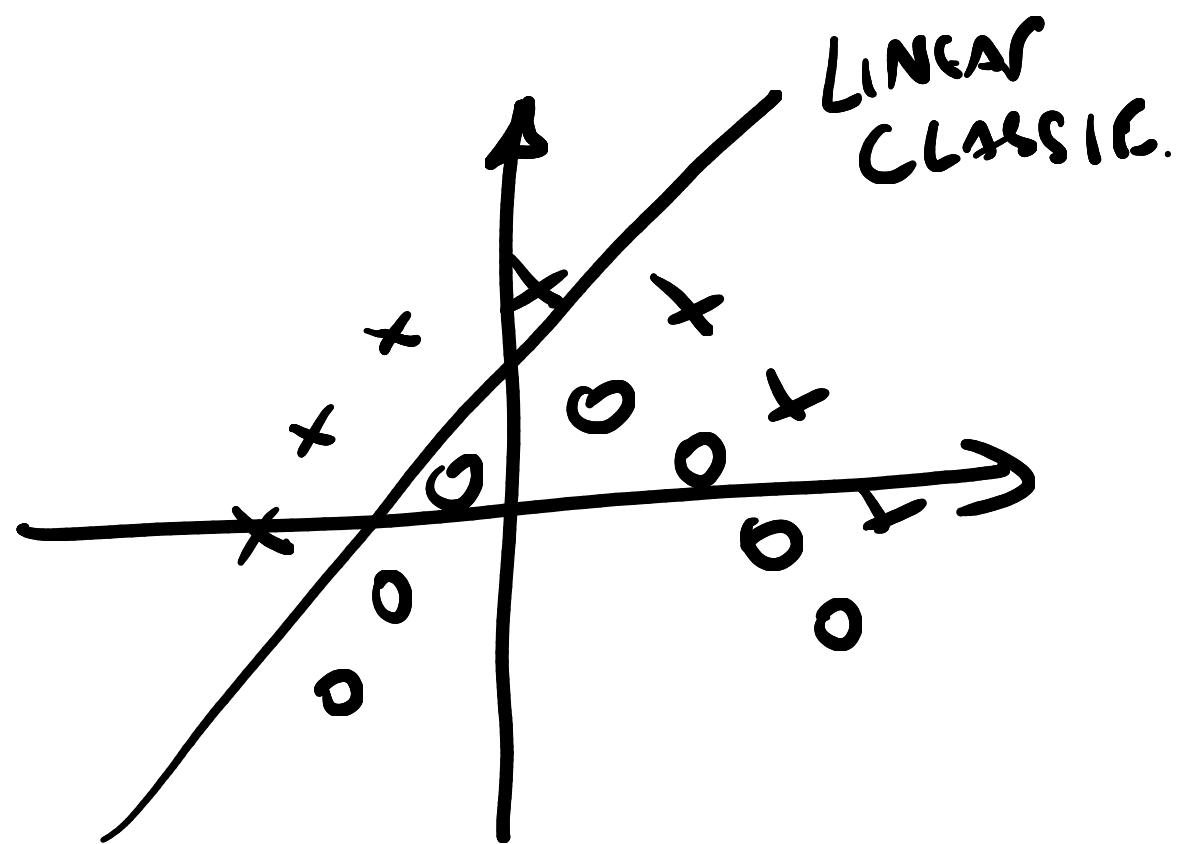
if  $N > m$   $\hat{\theta} = (X^T X)^{-1} X^T \bar{y}$

$$X = \begin{bmatrix} x_1 & \dots & x_N \end{bmatrix}$$

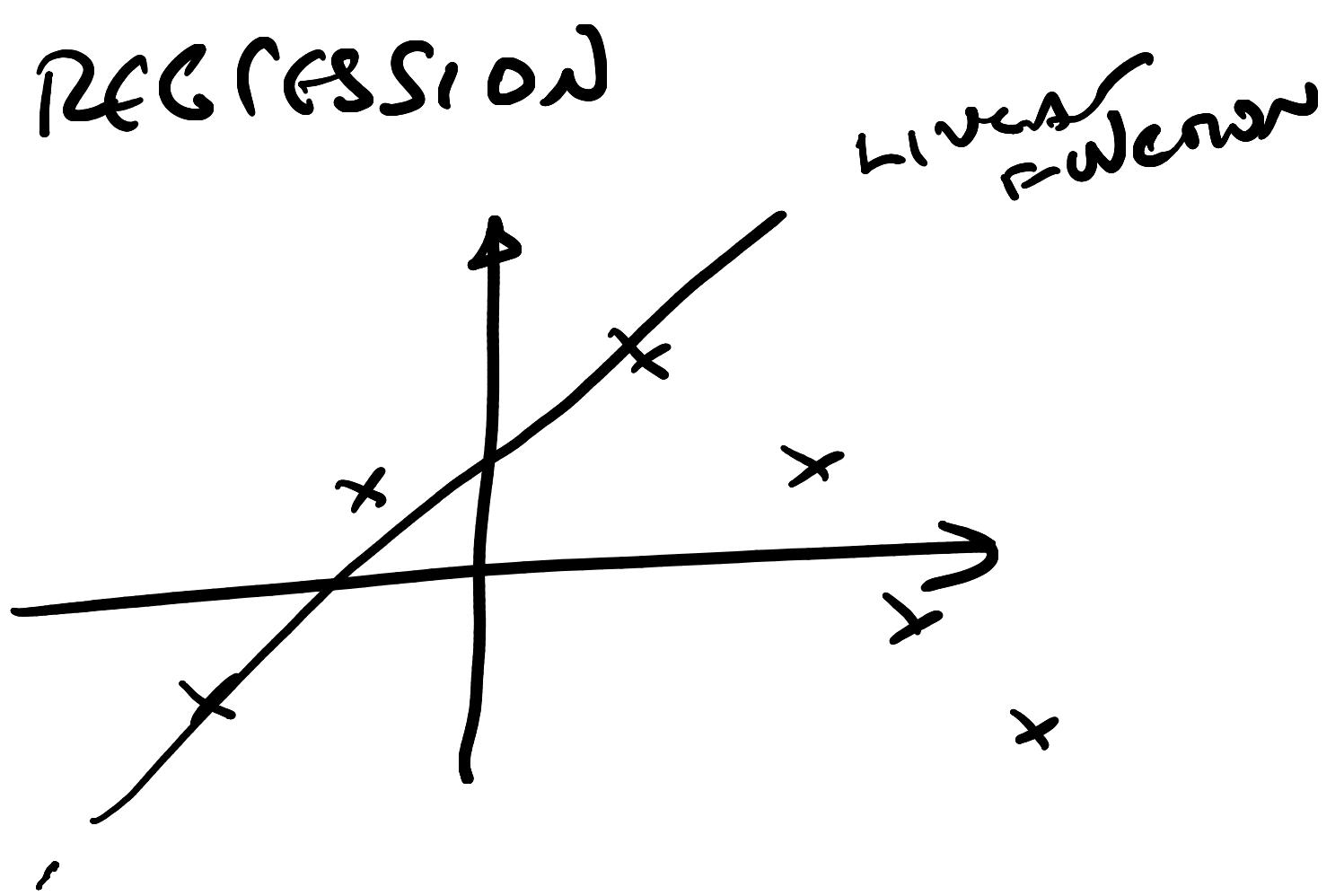
$$N > m \quad \bar{y} = \begin{bmatrix} y_1 \\ \vdots \\ y_N \end{bmatrix}$$

## Limitations of Linear Regression :

CLASSIFICATION



REGRESSION



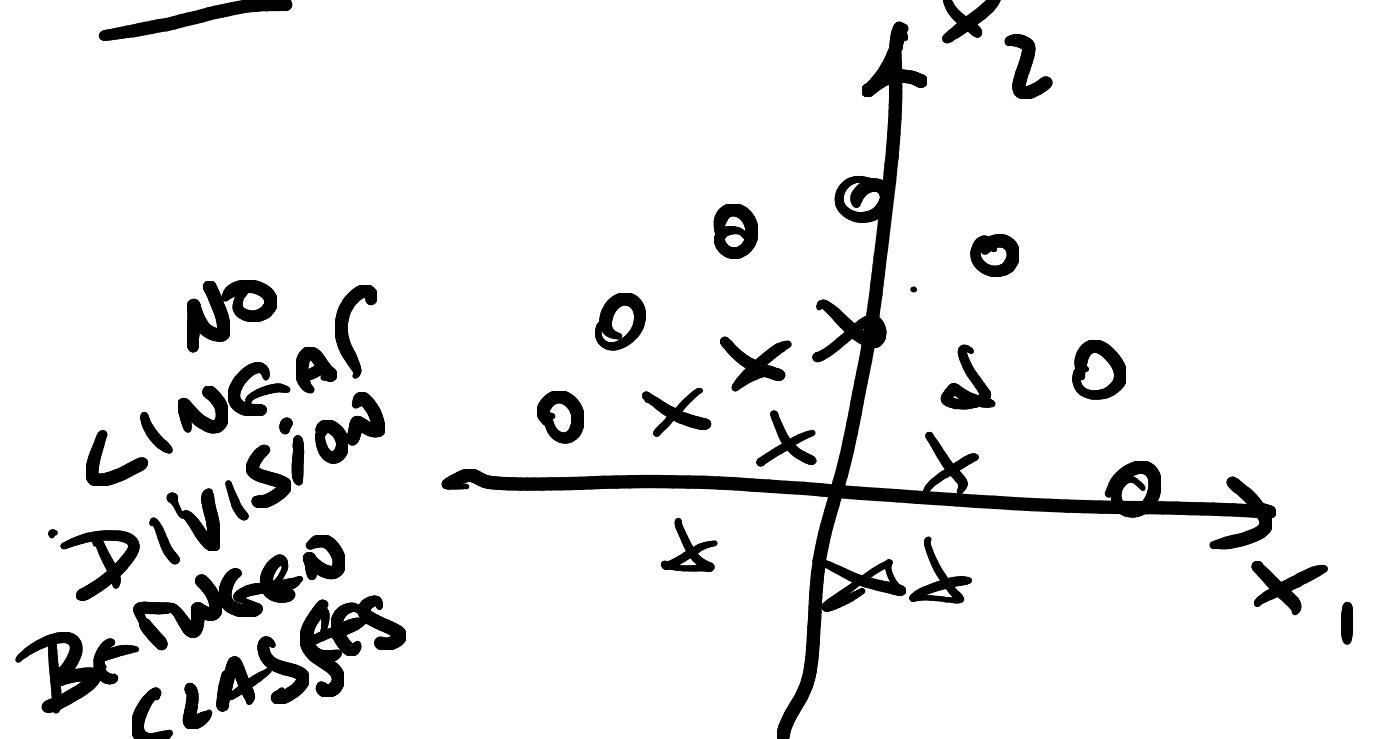
LIFTING

$$f_{\theta}(x) = \theta^T \Phi(x)$$

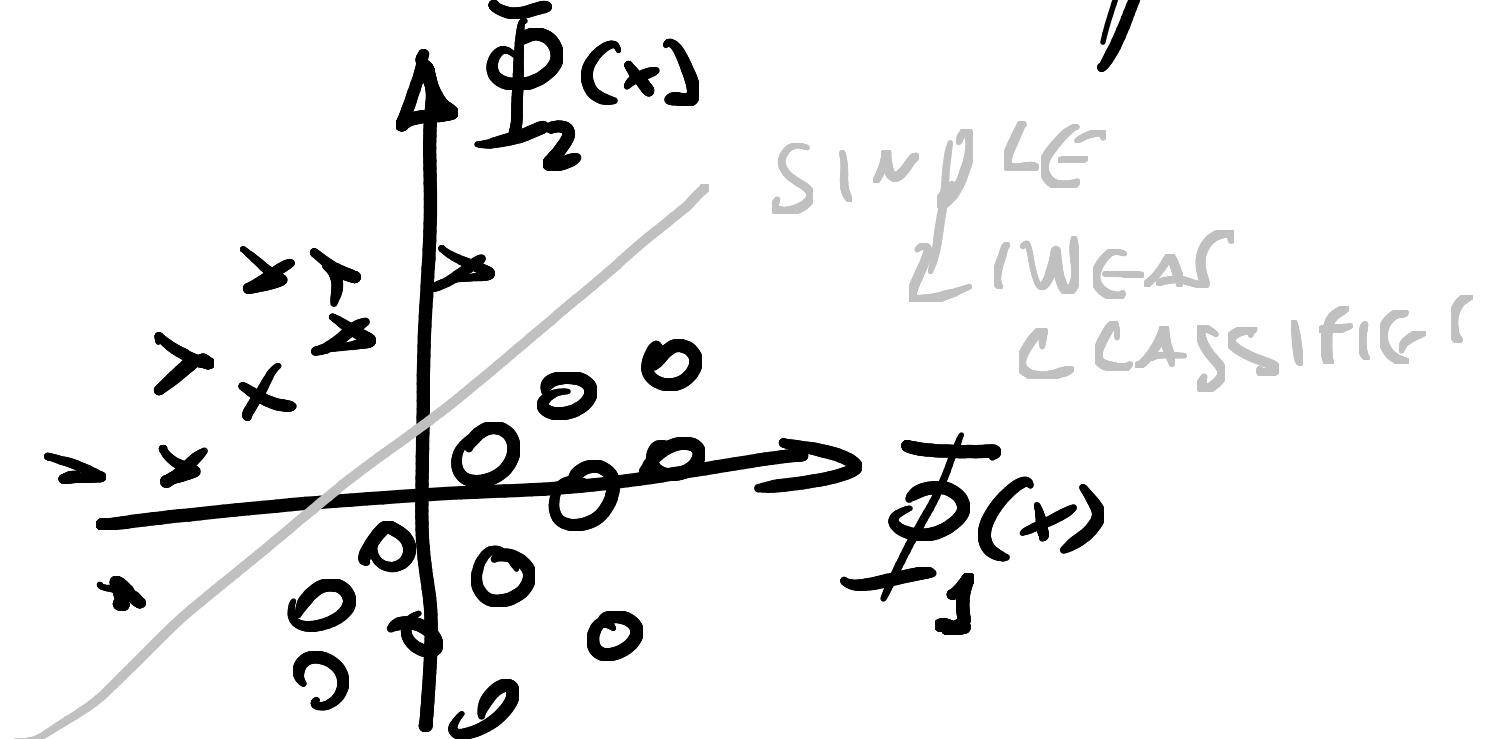
where  $\Phi: \mathcal{X} \rightarrow \mathbb{R}^P$  is called the feature mapping

- In a way, we replace dataset  $\{(x_i, y_i)\}$  by  $\{(\Phi(x_i), y_i)\}$
- $f_{\theta}$  is still linear in  $\theta$ ?

GOAL : LINEARIZE THE problem in THE LIFTED SPACE.



$\Phi$



POLYNOMIAL  $\Phi$  (order s)

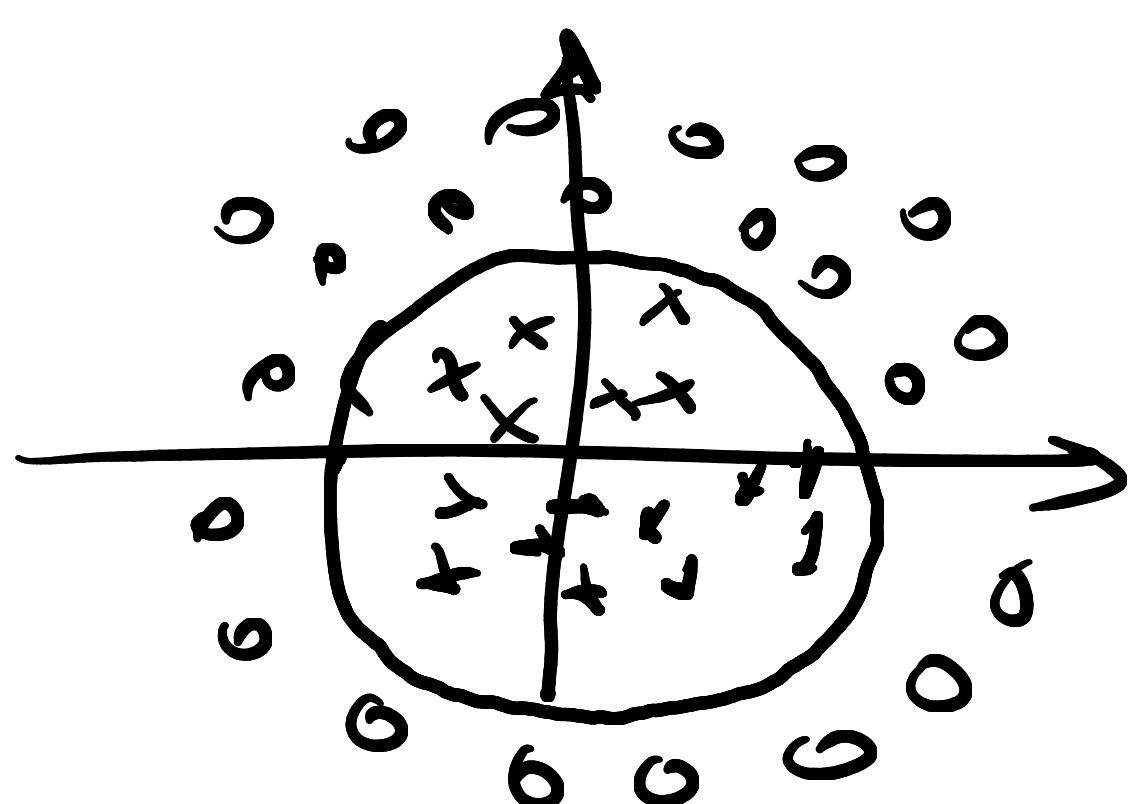
$\Phi(x)$  contains all the polynomials up to order s

EXAMPLE

$$X = \mathbb{R}^2 \quad x = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}$$

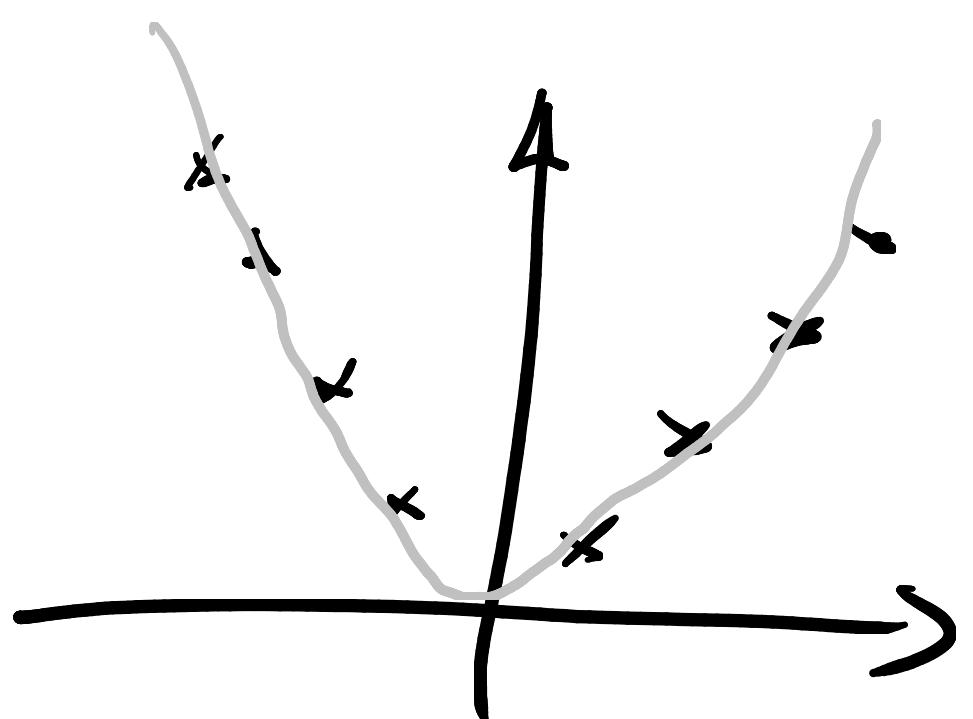
$$\Phi(x) = \begin{bmatrix} x_1 \\ x_2 \\ x_1 x_2 \\ x_1^2 + x_2^2 \end{bmatrix} \quad s=2$$

CLASSIFICATION



choose  $\theta = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \end{bmatrix} \Rightarrow f_\theta(x) = x_1^2 + x_2^2$

REGRESSION WITH  $X = \mathbb{R}$



$$\Phi(x) = \begin{bmatrix} x_1 \\ x_1^2 \end{bmatrix}$$

choose  $\theta = \begin{bmatrix} 0 \\ 1 \end{bmatrix} \Rightarrow f_\theta(x) = x^2$

# HOW MANY FEATURES DO YOU NEED?

- MORE FEATURES  $\rightarrow$  HIGHER DIMENSIONAL SPACE
 

$\downarrow$   
 more CHANCES OF LINEARIZATION

## A REGRESSION ANALYSIS:

$$Z = R^m$$

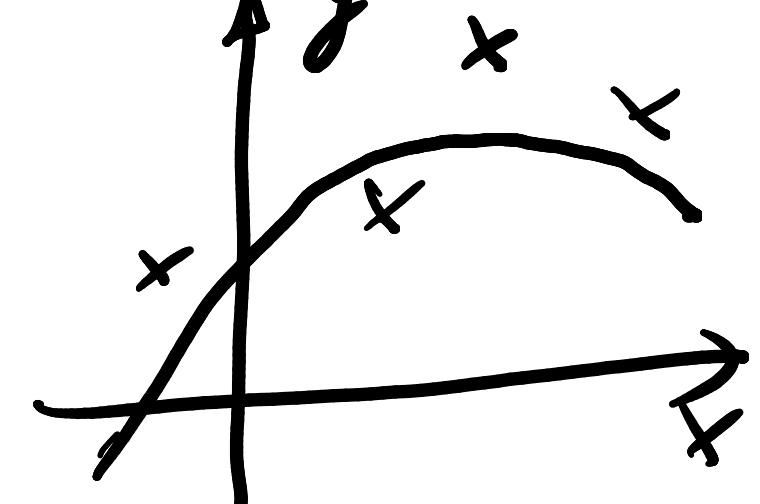
ERM  $\underset{\theta}{\operatorname{argmin}} \|\Phi^\top \theta - \bar{y}\|^2$  where  $\Phi = \begin{bmatrix} \Phi(x_1), \dots, \Phi(x_N) \end{bmatrix}$

if  $N > p$  UNDERPARAMTERIZED

$$\Rightarrow \hat{\theta} = (\Phi^\top \Phi)^{-1} \Phi^\top y \quad \text{UNIQUE SOLUTION}$$

IN GENERAL

$$f_{\hat{\theta}}(x_i) \neq y_i$$



if  $N \leq p$  OVERPARAMETERIZED

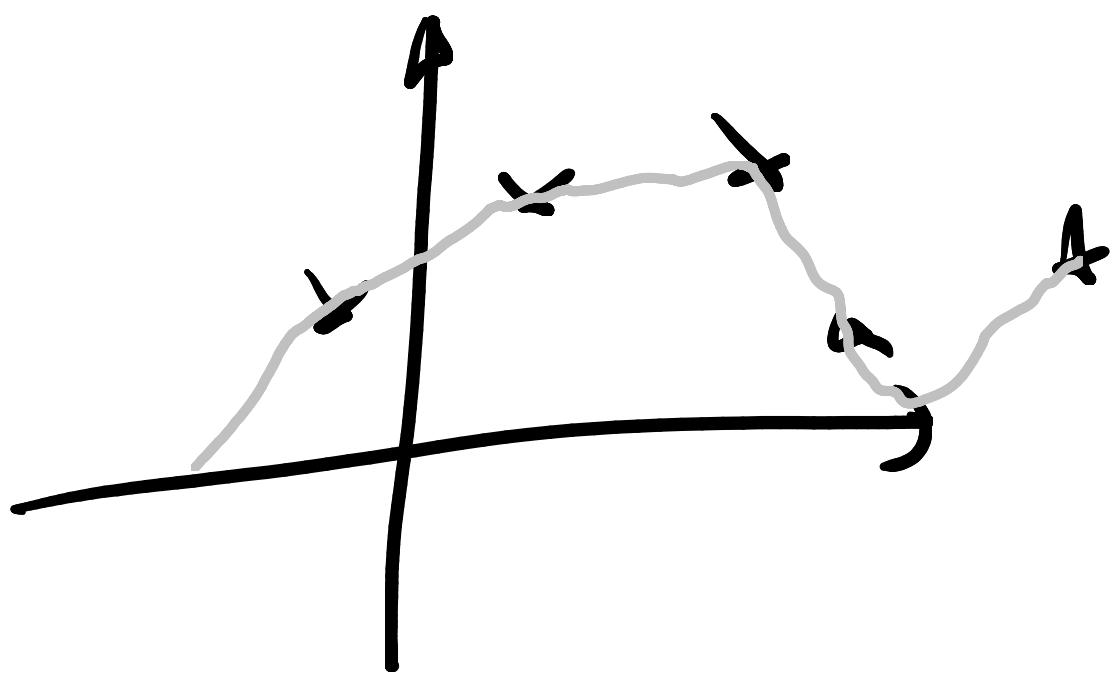
- THERE ARE INFINITE SOLUTIONS TO THE ERM PROBLEM

- ALL THE SOLUTIONS VERIFY

PERFECTLY FIT DATASET  $f_{\hat{\theta}}(x_i) = y_i \quad \forall i = 1, \dots, N$

$$\Leftrightarrow \hat{R}(\hat{\theta}) = 0$$

## INTERPOLATING SOLUTION



Example

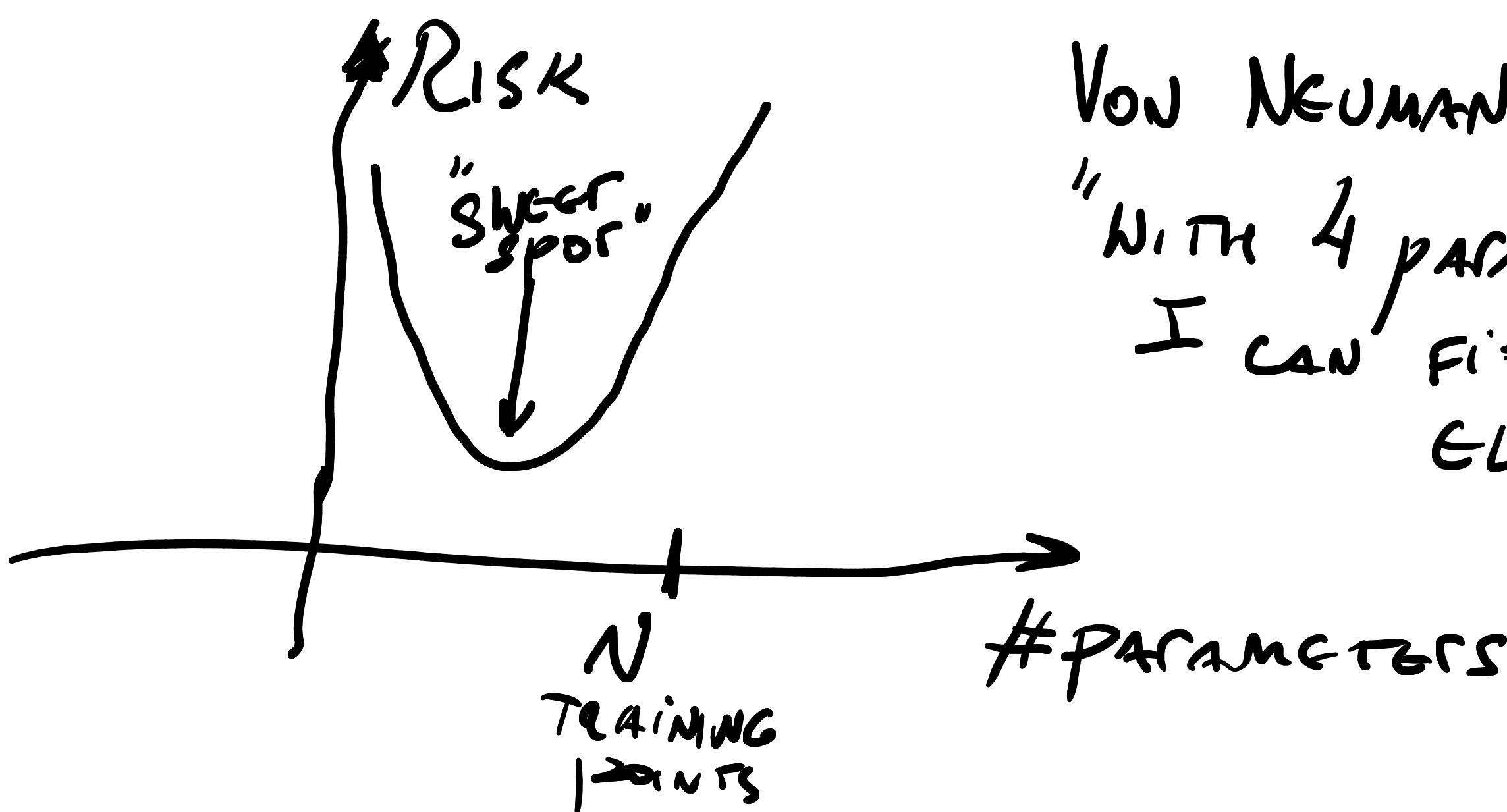
$\hat{\theta} = \Phi^T (\Phi^T \Phi)^{-1} \bar{y}$  is one solution  
(called minimum norm solution).

$$\begin{aligned} & \| \Phi^T \hat{\theta} - \bar{y} \|^2 \\ &= \| \cancel{\Phi^T} (\cancel{\Phi^T \Phi})^{-1} \bar{y} - \bar{y} \|^2 = \| \bar{y} - \bar{y} \|^2 = 0 \end{aligned}$$

zero empirical risk  
interpolating solution.

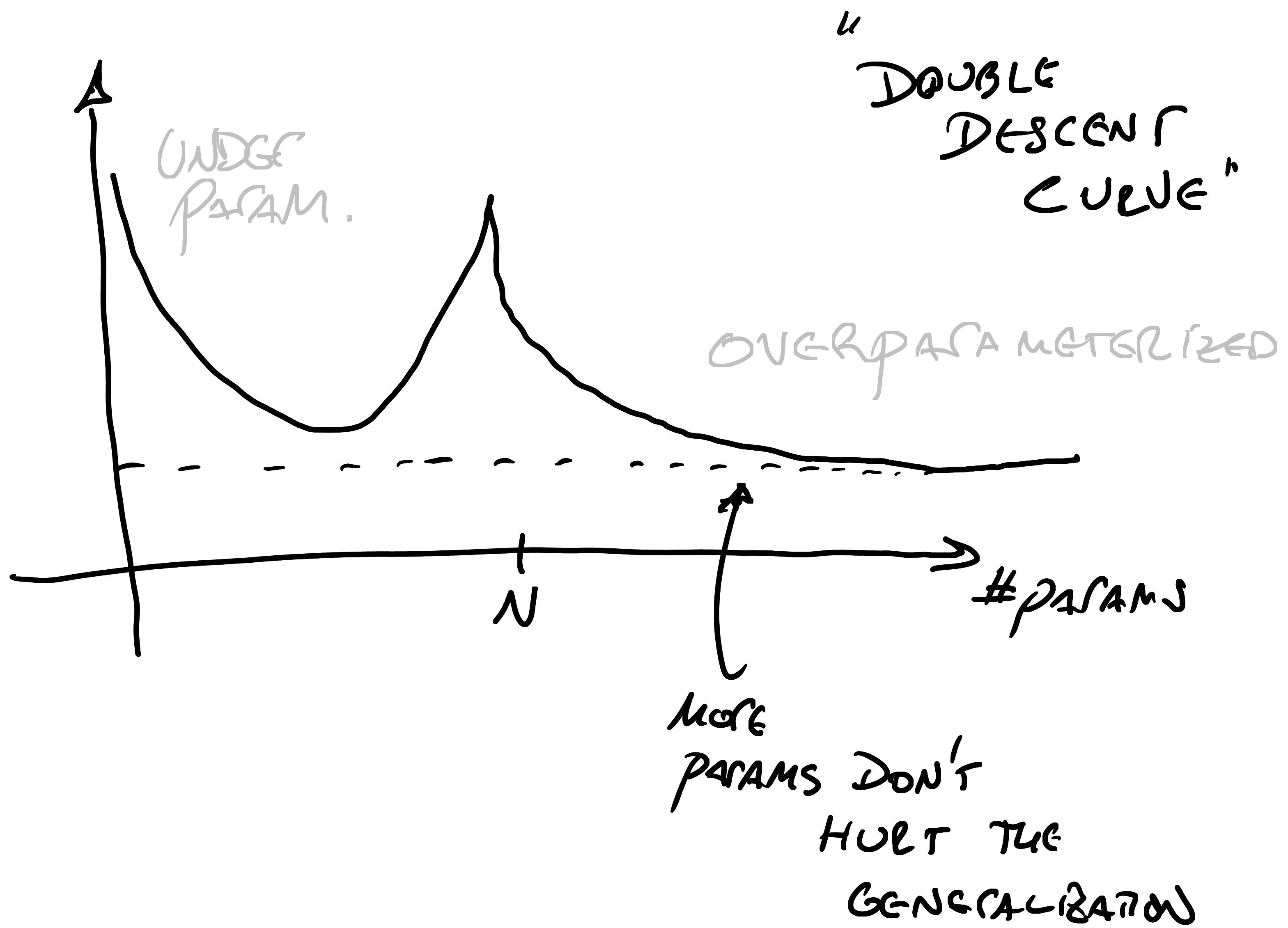
INTERPOLATION = OVERFITTING?

"TRADITIONAL" STATISTICAL BIAS vs VARIANCE



VON NEUMANN:  
"WITH 4 PARAMETERS  
I CAN FIT AN  
ELEPHANT".

# MODERN MACHINE LEARNING



## BASIS FUNCTIONS

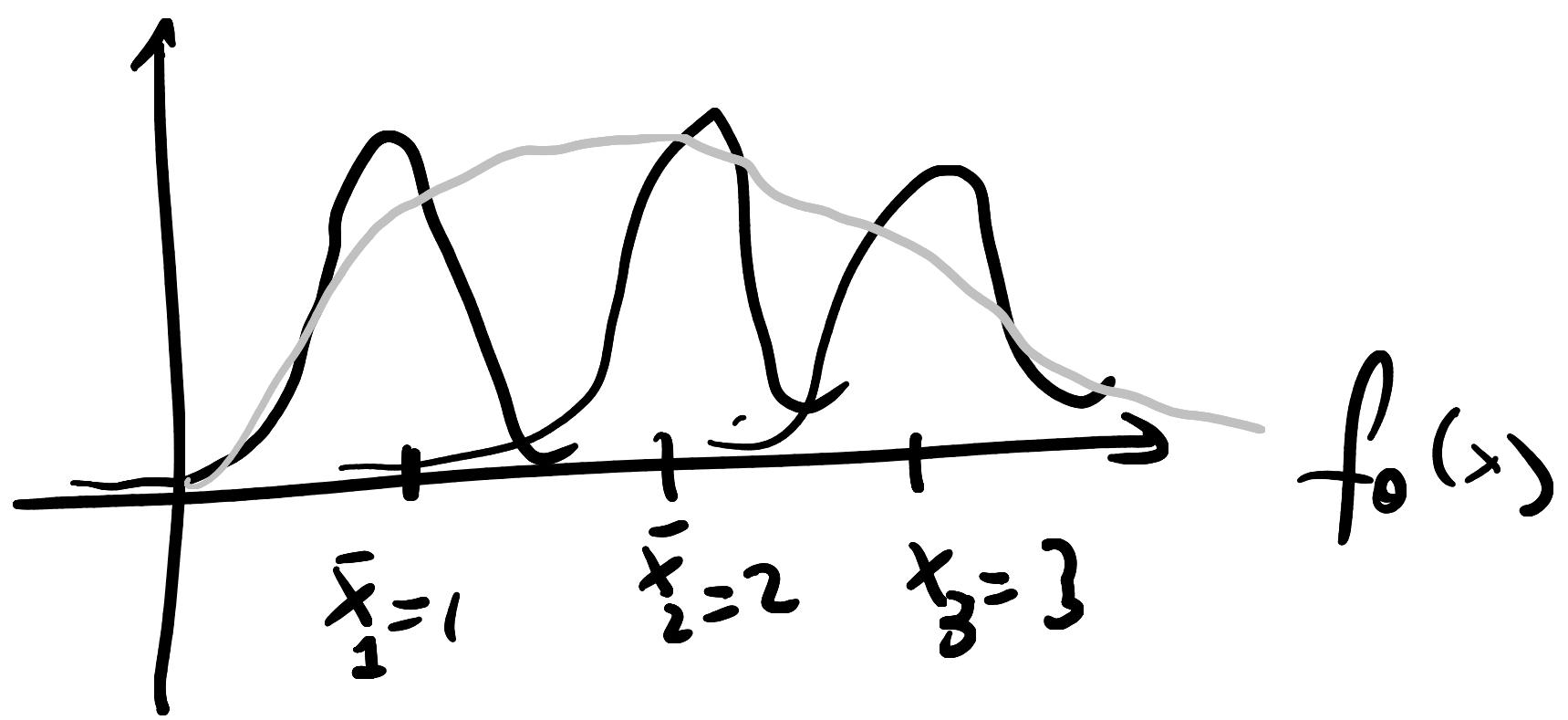
$$\left[ \Phi(x) \right]_j = \omega \left( \|x - \bar{x}_j\| \right) \quad \text{where}$$

$\bar{x}_j$  is a knot point

Example : Radial Basis Functions (RBFs)

$$\omega(t) = e^{-\frac{t^2}{\sigma^2}}$$

$$\Rightarrow f_0(x) = \sum_{j=1}^P \theta_j e^{-\frac{\|x - \bar{x}_j\|^2}{\sigma^2}}$$



- HOW SHOULD WE CHOOSE  $\bar{x}_j$  IN HIGH DIMENSIONS?

### KERNELS :

If  $p \gg N$ , computing the explicit feature map might be costly! However we saw that:

$$\hat{\theta} = \bar{\Phi} (\bar{\Phi}^T \bar{\Phi})^{-1} \bar{y}$$

AND  $f_0(x) = \hat{\theta}^T \bar{\Phi}(x) = \bar{y}^T (\bar{\Phi}^T \bar{\Phi})^{-1} \bar{\Phi}^T \bar{\Phi}(x)$

DEFINE  $K = \bar{\Phi}^T \bar{\Phi}$  Gram matrix

$$K_{ij} = \bar{\Phi}(x_i)^T \bar{\Phi}(x_j)$$

$\triangleq k(x_i, x_j)$  KERNEL FUNCTION

$$\Rightarrow f_0(x) = \bar{y}^T K^{-1} \begin{bmatrix} k(x_1, x) \\ \vdots \\ k(x_m, x) \end{bmatrix}$$

SINCE  $\bar{\Phi}^T \bar{\Phi}(x) = \begin{bmatrix} \bar{\Phi}(x_1)^T \bar{\Phi}(x) \\ \vdots \\ \bar{\Phi}(x_m)^T \bar{\Phi}(x) \end{bmatrix} = \begin{bmatrix} k(x_1, x) \\ \vdots \\ k(x_m, x) \end{bmatrix}$

## OBSERVATIONS :

① Often, computing  $K(x_i, x_j)$  is cheaper than  $\underline{\Phi}(x_i)^\top \underline{\Phi}(x_j)$ !

For example:

$$a) \underline{\Phi}(x) = \begin{bmatrix} 1 \\ \sqrt{2}x \\ x^2 \end{bmatrix} \Leftrightarrow K(x, u) = (1+xu)^2$$

$$\begin{aligned} b) K(x, u) &= e^{-\frac{(x-u)^2}{\sigma^2}} = e^{-\frac{x^2}{\sigma^2}} e^{-\frac{u^2}{\sigma^2}} e^{2\frac{xu}{\sigma^2}} \\ (x, u \in \mathbb{R}) &= \sum_{j=0}^{\infty} \frac{(2xu)^j}{j!} e^{-\frac{x^2}{\sigma^2}} e^{-\frac{u^2}{\sigma^2}} \\ &= \underline{\Phi}(x)^\top \underline{\Phi}(u) \end{aligned}$$

$$\text{where } [\underline{\Phi}(x)]_j = e^{-\frac{x^2}{\sigma^2}} \frac{\sqrt{2}x^j}{j!}$$

here  $p = \infty$  infinitely many features!

↓  
it would be  
impossible to compute  
the feature map  
explicitly.

## ② KNOT POINTS

since  $\hat{f}_0(x) = \bar{y}^T K^{-1} \begin{bmatrix} k(x_1, x) \\ \vdots \\ k(x_n, x) \end{bmatrix}$

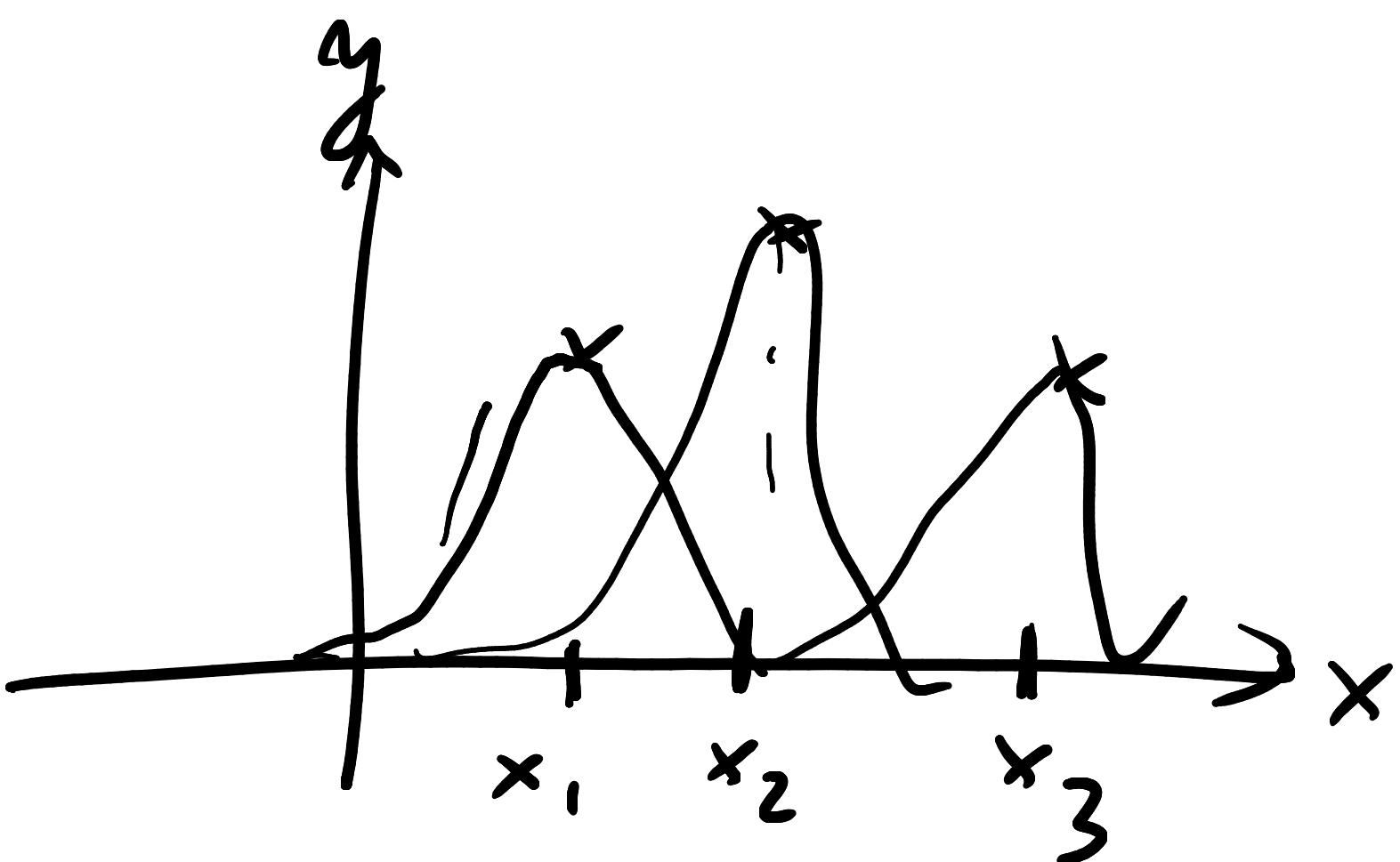
$\hat{f}_0(x) = \sum_{i=1}^n \alpha_i k(x_i, x)$  Representation Theorem

THE SOLUTION ALWAYS HAS THIS FORM.

where  $\alpha_i = [\bar{y}^T K^{-1}]_i$

WE CAN SEE  $k(x_i, x)$  AS A BASIS FUNCTION

WITH KNOT POINTS  $\bar{x}_i = x_i$   
GIVEN BY TRAINING POINTS



$$k(x, x_i) = e^{-\frac{\|x - x_i\|^2}{\sigma^2}}$$

## ③ INTERPREATION OF KERNEL.

- IS THE INNER PRODUCT IN FEATURE SPACE
- MODELS A "DISTANCE" BETWEEN TWO INPUT POINTS  $x_i$  AND  $x_j$ .

# NEURAL NETWORKS

IDEA: IN many problems, CHOOSING THE RIGHT FEATURE MAP, KERNEL OR KNOT POINTS IS HARD.

↳ Why NOT LEARN THEM FROM DATA?

One hidden layer network:

$$f_0(x) = \theta_2^\top \sigma(\Theta_1 x + \theta_1)$$

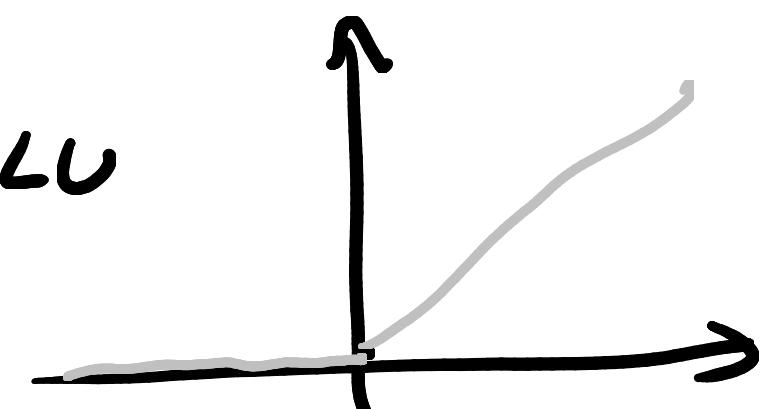
FEATURE mapping

WITH LEARNING PARAMETERS

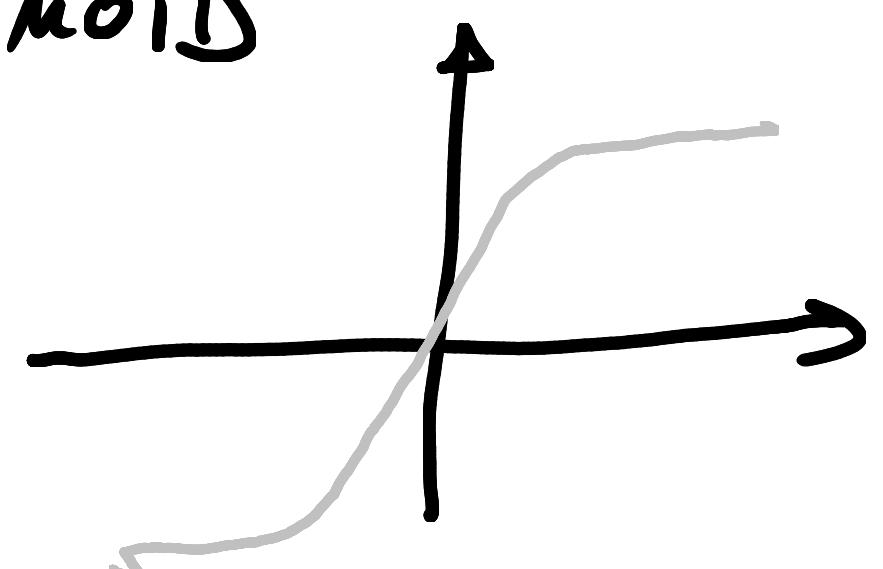
$$\Theta_1 \in \mathbb{R}^{P \times n} \text{ and } \theta_2 \in \mathbb{R}^m$$

WHERE  $\sigma$  IS A NON-LINEARITY

- $\sigma(x) = \max(x, 0)$  RELU



- $\sigma(x) = \frac{1}{1 + e^{-x}}$  SIGMOID



- UNIVERSAL APPROXIMATION THEOREM: WITH SUFFICIENT NEURONS  $\rho$ ,  $f_\theta(x)$  CAN APPROXIMATE ANY CONTINUOUS FUNCTION.
- IF WE FIX THE FIRST LAYER, THE NN BECOMES A SIMPLE FEATURE MAP.
- THERE IS NO CLOSED FORM SOLUTION TO THE LEARNING PROBLEM

$$\text{optim} \sum_{i=1}^N (f_\theta(x_i) - y_i)^2$$

EVEN WORSE, THERE ARE MULTIPLE (POSSIBLY LOCAL) MINIMA.

⇒ WE GENERALLY USE GRADIENT DESCENT TO OPTIMIZE THEM.

Deep Neural Networks ("AKA" Deep Learning):

- IDEA: CONCATENATE MANY LAYERS

$$f_\theta(x) = \theta_L^\top \sigma(\theta_{L-1}^\top \dots \sigma(\theta_1^\top x + \theta_1) + \theta_{L-1})$$

$\nwarrow$  L TIMES

LEARNABLE PARAMETERS

$$\theta = \{\theta_1, \dots, \theta_L, \theta_1, \dots, \theta_{L-1}\}$$

- MOST NEURAL NETWORKS USED IN PRACTICE DO NOT TAKE THIS SIMPLE FORM.

$\hookrightarrow$  CNNs (CONVOLUTIONAL NEURAL NETWORKS)  
 $\hookrightarrow$  GNNs (GRAPH NEURAL NETWORKS)  
 $\hookrightarrow$  TRANSFORMERS (ATTENTION-BASED NETWORKS)

- THIS ARCHITECTURES CAN BE (PARTLY) UNDERSTOOD THROUGH THE LENS OF EQUIVARIANCE.

Some MATHEMATICAL CONCEPTS:

Group Action: The action of a group  $G$  is the set of invertible transformations  $\{T_g\}_{g \in G}$ .  
 We must have

$$\textcircled{1} \text{ identity } T_e = \text{Id}$$

$$\textcircled{2} \text{ inverse } T_g^{-1} = T_{g^{-1}}$$

$$\textcircled{3} \text{ composition } T_{g_1} T_{g_2} = T_{g_1 \cdot g_2}$$

## EXAMPLES:

- IMAGES
- GRAPHS
- TRANSLATIONS (in 1D, 2D, ...)
  - ROTATIONS (in 2D, 3D)
  - REFLECTIONS
  - PERMUTATIONS

## EQUIVARIANCE:

$$f_0(x) : \mathbb{R}^{n_{in}} \rightarrow \mathbb{R}^{n_{out}}$$

EQUIVARIANCE TO  $\{\tau_g\}_{g \in G}$  if

$$f_0(\tau_g x) = \tilde{\tau}_g f_0(x)$$

↳ valid group action

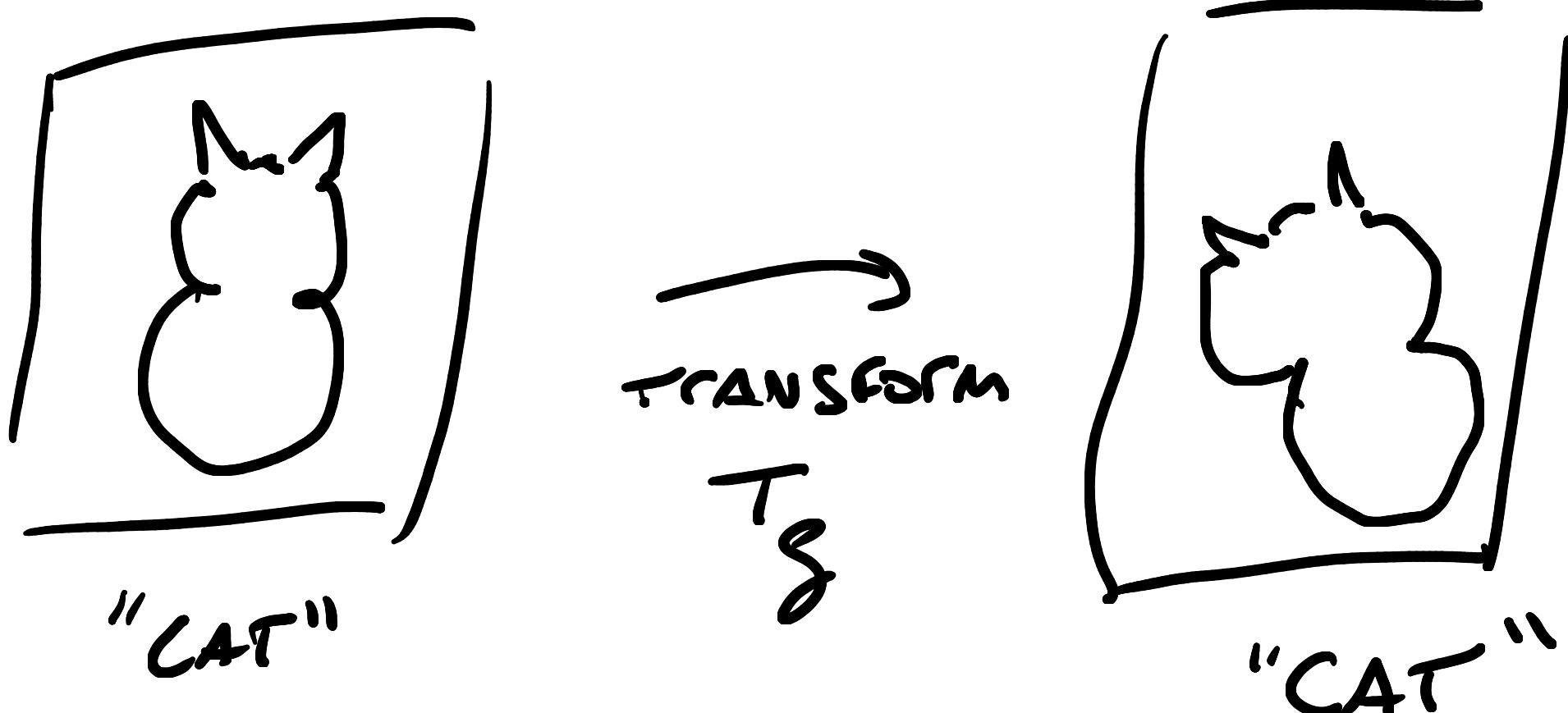
INVARIANCE: A PARTICULAR CASE OF EQUIVARIANCE

WHERE  $\tilde{\tau}_g = \text{Id}$   $\forall g \in G$  (TRIVIAL GROUP ACTION)

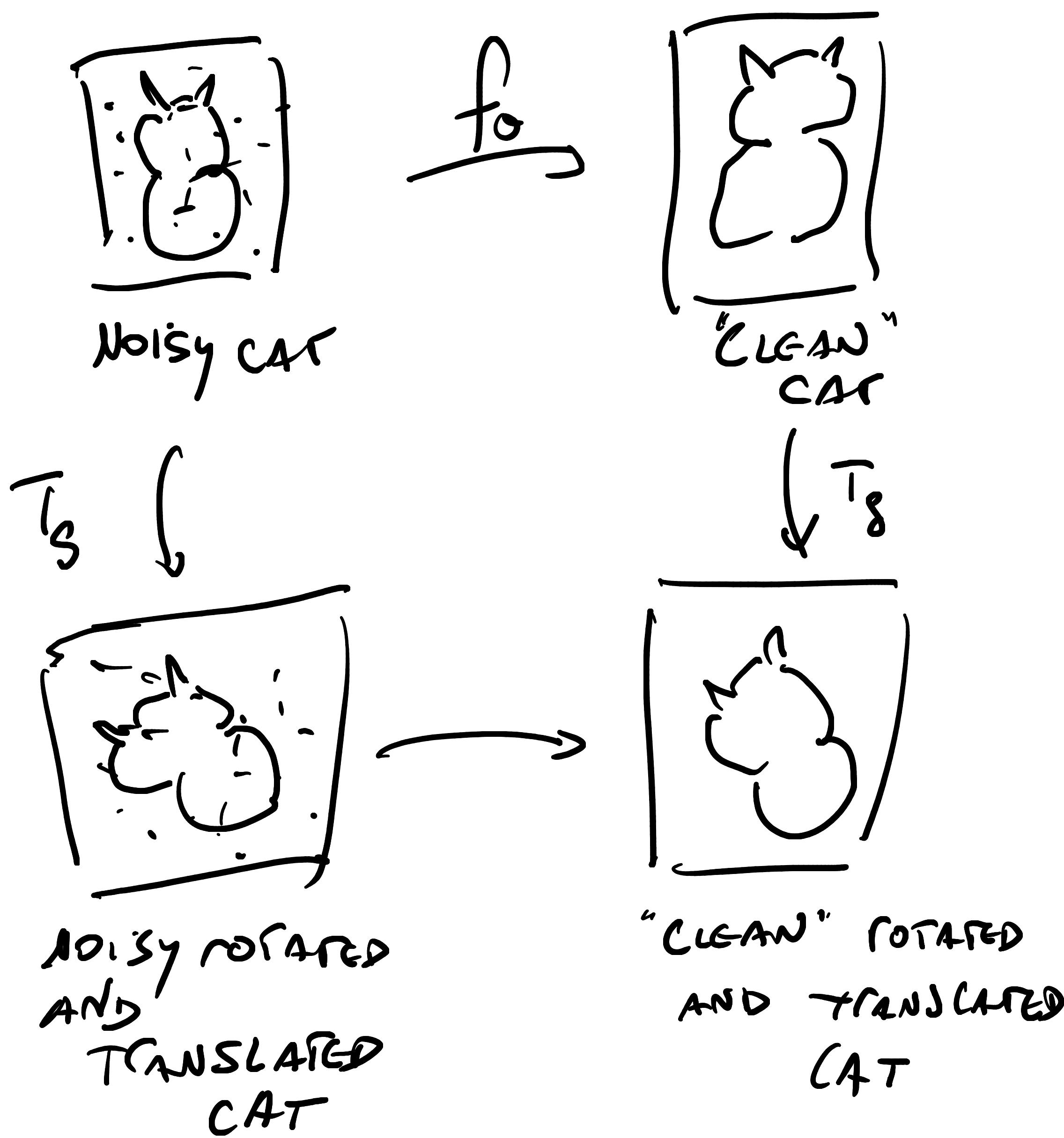
OR IN OTHER WORDS

$$f_0(\tau_g x) = f_0(x)$$

IN CLASSIFICATION TASKS, WE WANT AN INVARIANT FUNCTION  $f_\theta(\bar{T}_g x) = x$



IN REGRESSION TASKS, WE WANT AN EQUIVARIANT FUNCTION:



Idea : Parameterize  $f_\theta$  so that it is invariant/equivariant for any possible values of  $\theta$   
 (i.e. even without any training)

- THE COMPOSITION OF TWO EQUIVARIANT FUNCTIONS IS EQUIVARIANT.

MATHEMATICALLY  $f = f_1 \circ f_2$

if  $f_1$  AND  $f_2$  ARE EQUIV.  
 $\Rightarrow f$  is equiv.

A NEURAL NETWORK HAS 3 OPERATIONS

- BIAS :  $f(x) = x + \theta$  WHICH IS ALWAYS EQUIVARIANT. ✓

- NON-LINEARITY :  $f(x) = \sigma(x)$   
 WHICH IS EQUIVARIANT  
 TO ROTATIONS IF  
 APPLIED PIXEL-WISE

$$\sigma(x) = \begin{bmatrix} \sigma(x_1) \\ \vdots \\ \sigma(x_m) \end{bmatrix}$$

- LINEAR TRANSFORMATION

$f(x) = \Theta x$  which is NOT necessarily equivariant  
for all  $\Theta \in \mathbb{R}^{P_1 \times P_2}$

- we can look for matrices that verify

$$S = \left\{ \Theta \in \mathbb{R}^{P_1 \times P_2} : \Theta \tilde{T}_g = \tilde{T}_g \Theta \quad \forall g \in G \right\}$$

Theorem:  $S$  is a linear subspace of  $\mathbb{R}^{P_1 \times P_2}$   
AND thus equiv.  $\Theta$  can be parameterized as

$$\Theta^{\text{equiv}} = \sum_{i=1}^K \theta_i \psi$$

where  $\psi_i \in \mathbb{R}^{P_1 \times P_2}$  are some basis

PARAMETER EFFICIENCY :  $\frac{K}{P_1 P_2} \ll 1$

↑  
equivariant  
matrices  
↓  
any matrix.

If  $T_g$  ARE TRANSLATIONS, WE GET

$$\Theta^{\text{conv.}} = \text{circ}(\Theta)$$

CIRCULAR MATRIX

$$\text{circ}(\Theta)x = \Theta * x \quad \text{CONVOLUTION}$$

→ ONLY  $m$  PARAMETERS  
INSTEAD OF  $m^2$

CNN

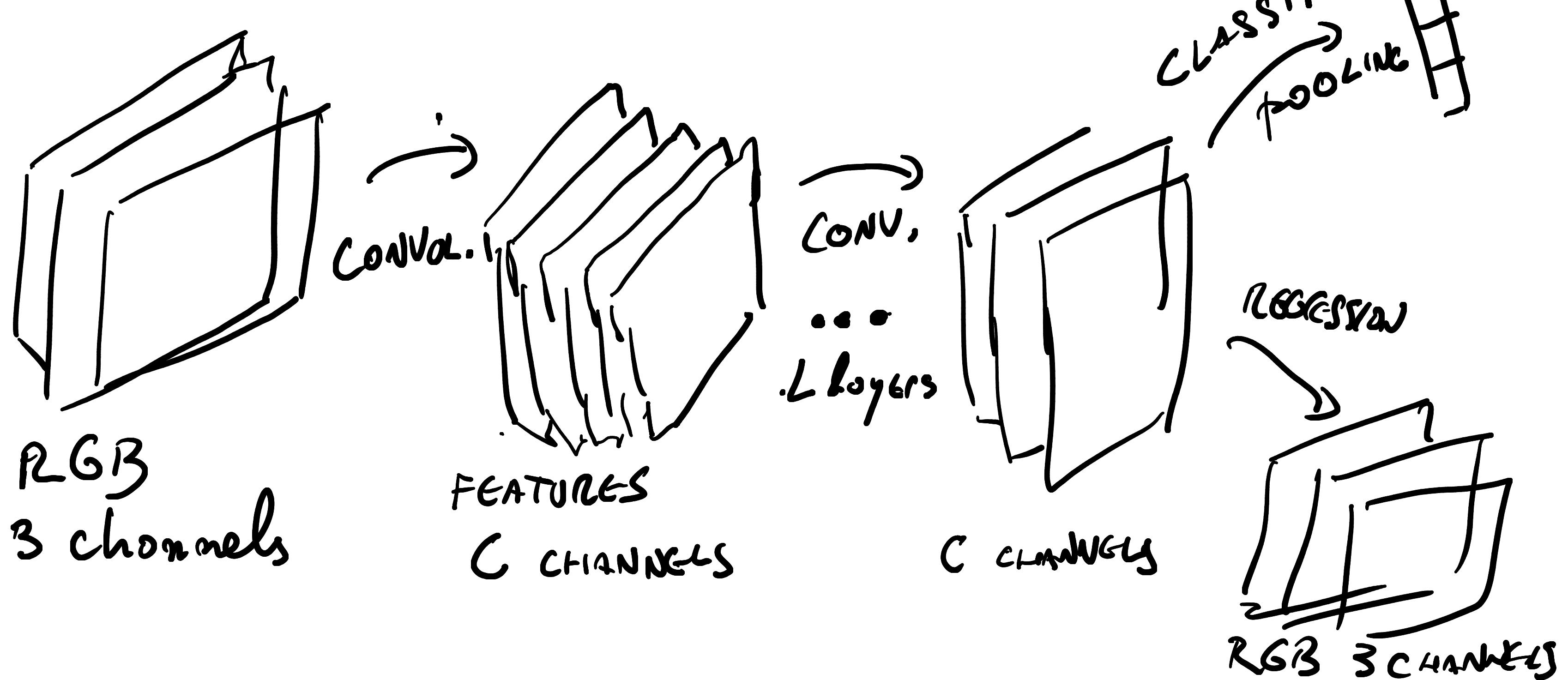
$$f_l(x) = \theta_l^\top \sigma(\underbrace{\theta_{L-1} + \dots + \theta_1}_\text{EQUIVARIANT STEPS} * x + \tilde{\theta}_l)$$

- parameters  $\theta_l$  ARE CALLED FILTERS/WEIGHTS.

- WE GENERALLY CHOOSE THEM OF SMALL SIZE

e.g.  $3 \times 3$  FOR IMAGES INSTEAD OF  $n$  PIXELS.

↳ "LOCAL" FUNCTIONS.



## POOLING

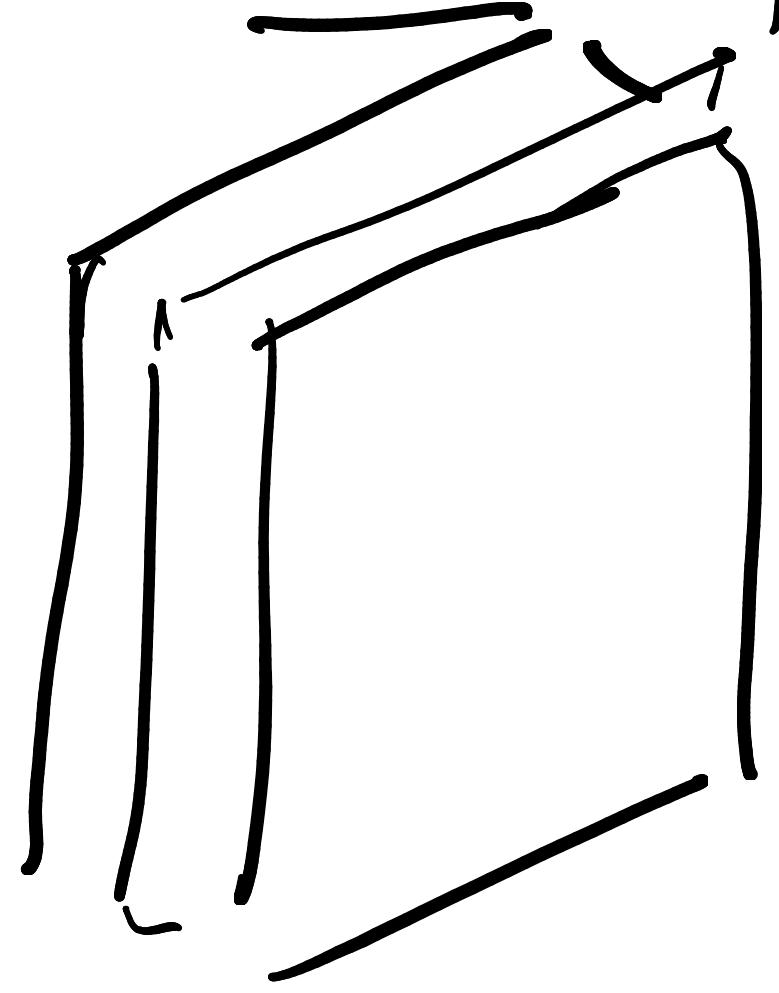
AVERAGE ACROSS THE IMAGE

FROM EQUIVARIANT  $\rightarrow$  INVARIANT  
FUNCTIONS



TAKE AN AVERAGE OF THE WHOLE IMAGE

### GLOBAL POOLING:



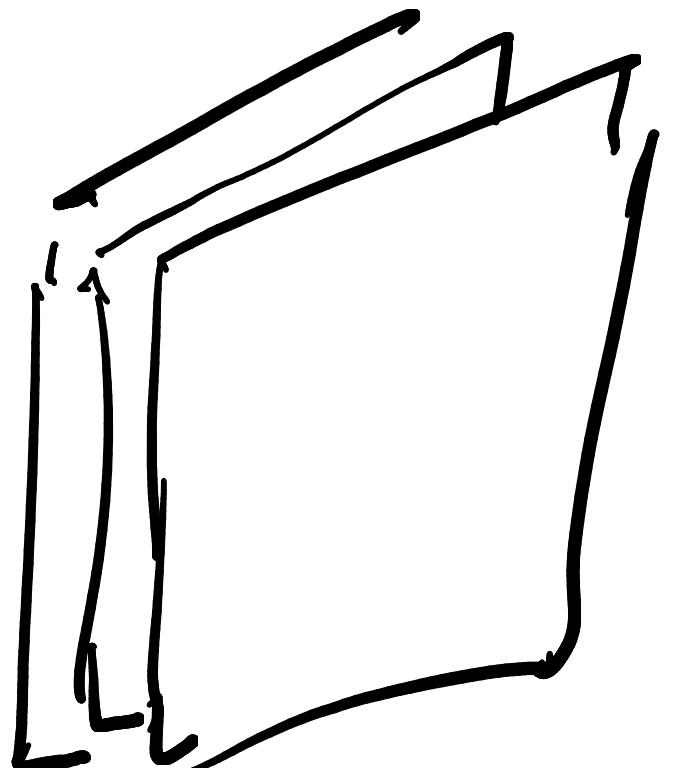
AVERAGE  
EVERY  
image  
separately, C VALUES

" $1 \times 1$ " pixels.

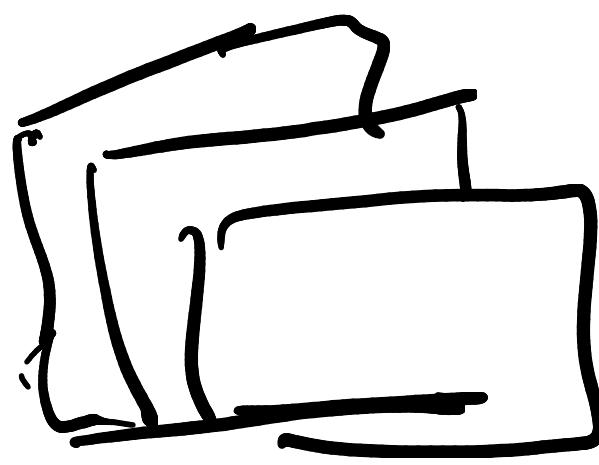
C CHANNELS

H x W pixels

### LOCAL POOLING



pool every  $2 \times 2$



C CHANNELS

$\frac{H}{2} \times \frac{W}{2}$  pixels

C CHANNELS

$\frac{H}{2} \times \frac{W}{2}$  pixels