https://introml.mit.edu/

6.390 Intro to Machine Learning

Lecture 5: Features, Neural Networks I

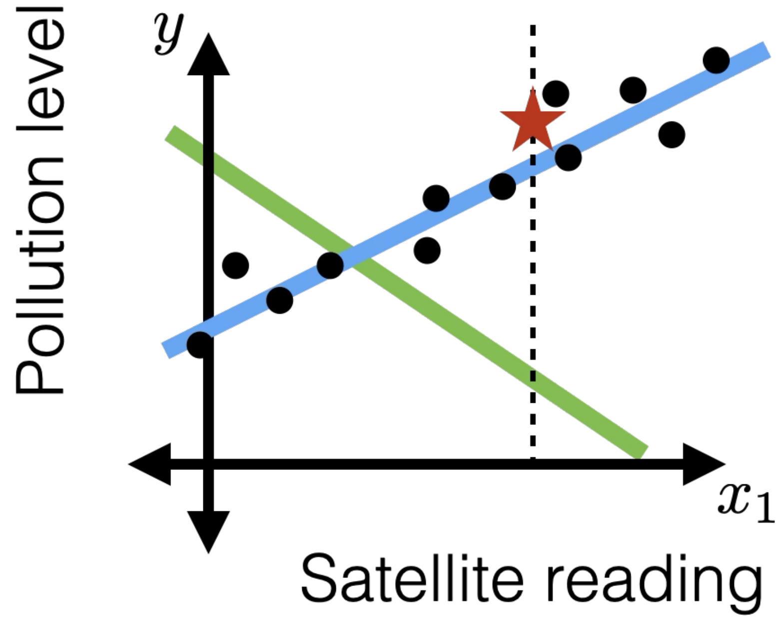
Shen Shen

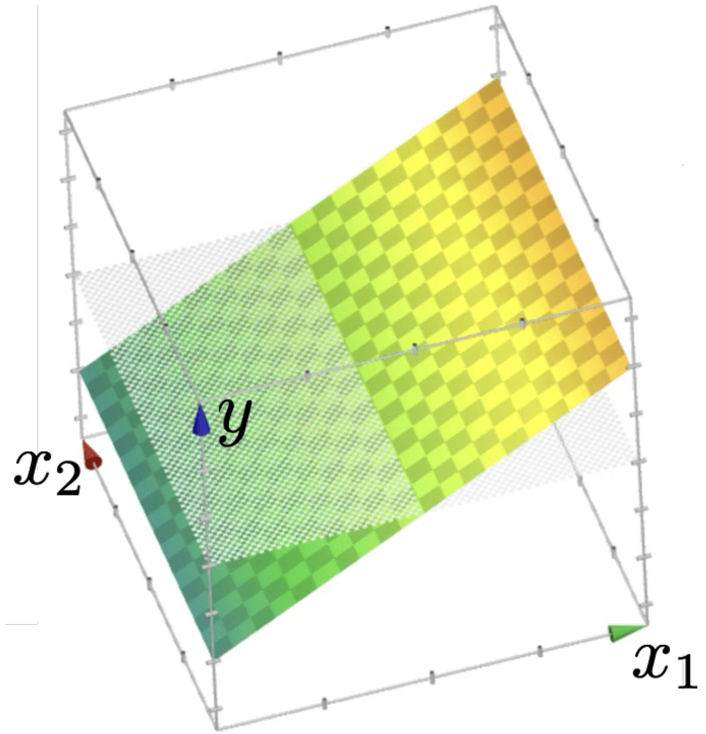
Feb 28, 2025

11am, Room 10-250

Recap:

linear regressor



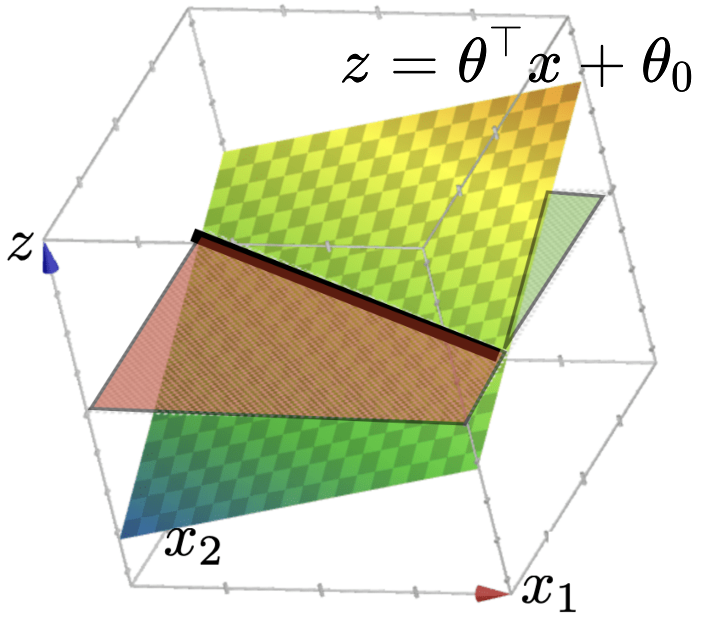


$$y = \theta^\top x + \theta_0$$

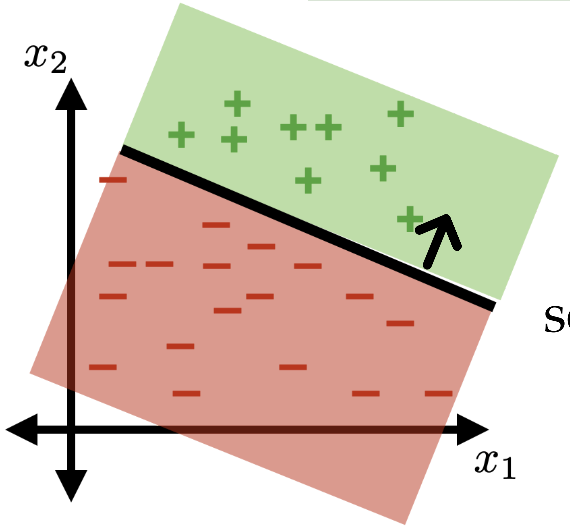
the regressor is **linear** in the feature x

Recap:

linear classifier



$\{x : \theta^\top x + \theta_0 > 0\}$



separator

$\{x : \theta^\top x + \theta_0 = 0\}$

$\{x : \theta^\top x + \theta_0 < 0\}$

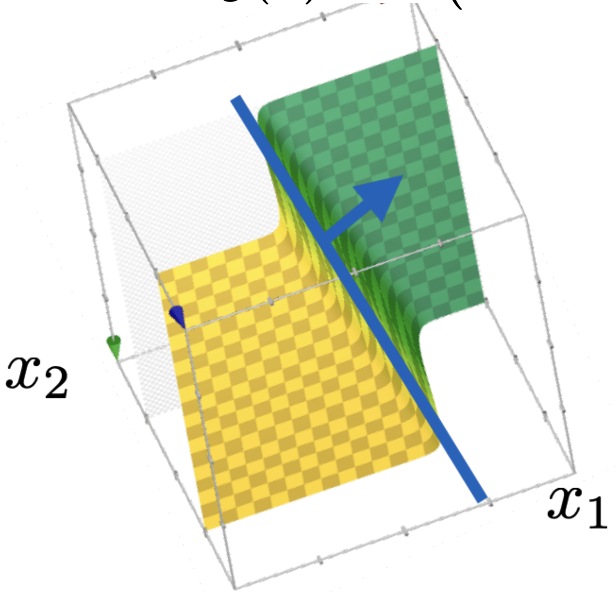
the separator is **linear** in the feature x

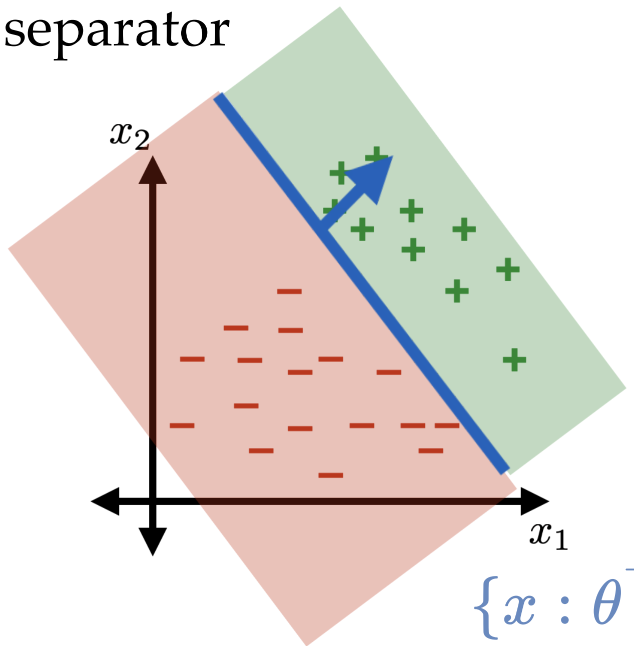
Recap:

linear logistic classifier

$\{x : \sigma(\theta^\top x + \theta_0) > 0.5\}$

$g(x) = \sigma\left(\theta^\top x + \theta_0\right)$



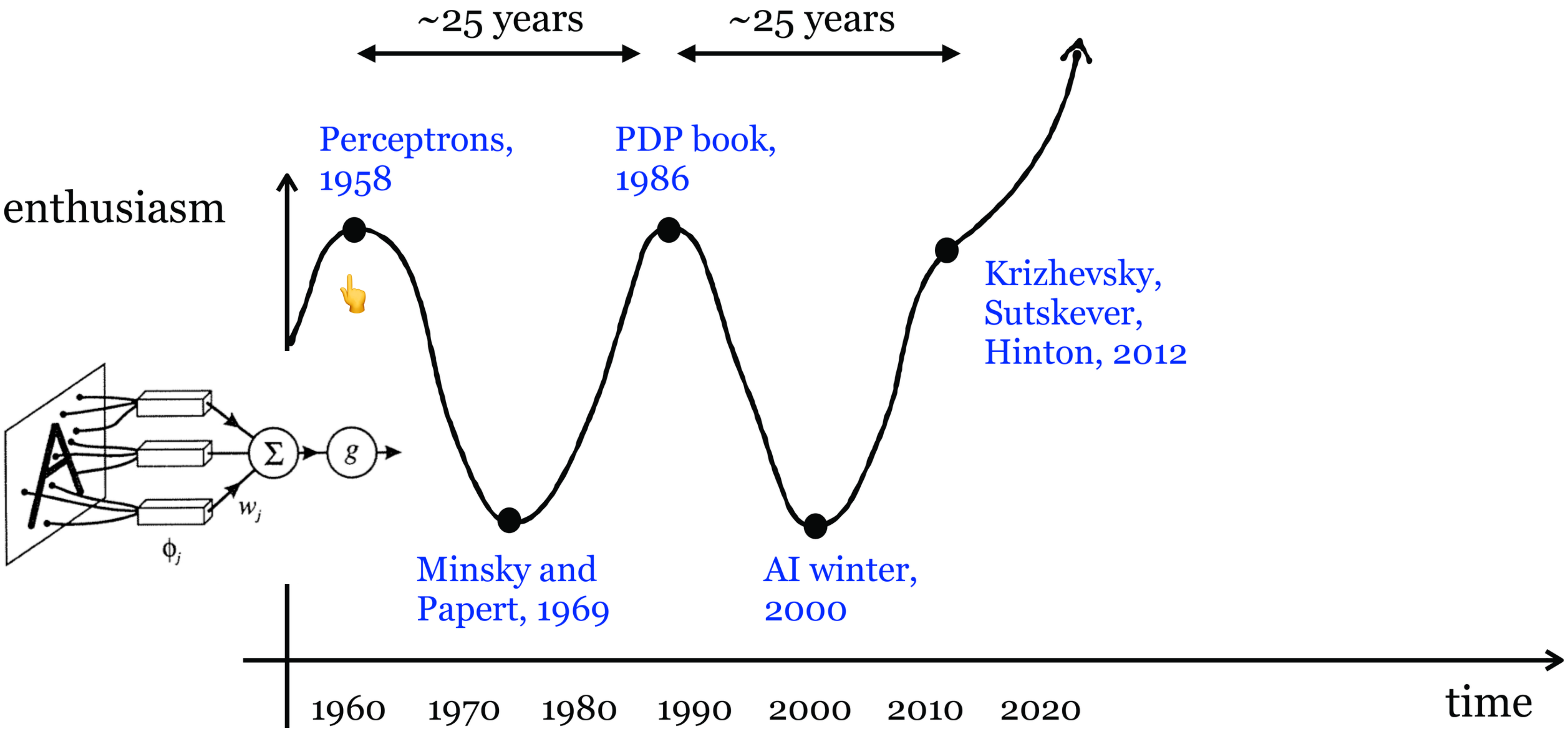


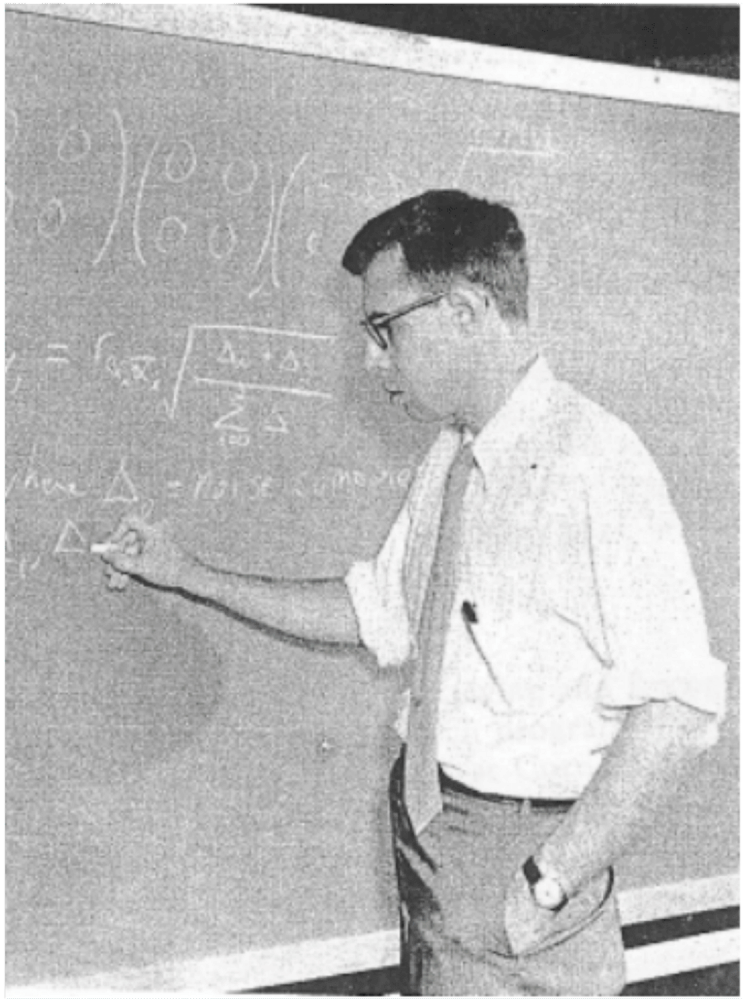
$\{x : \theta^\top x + \theta_0 = 0\}$

$\{x : \sigma(\theta^\top x + \theta_0) < 0.5\}$

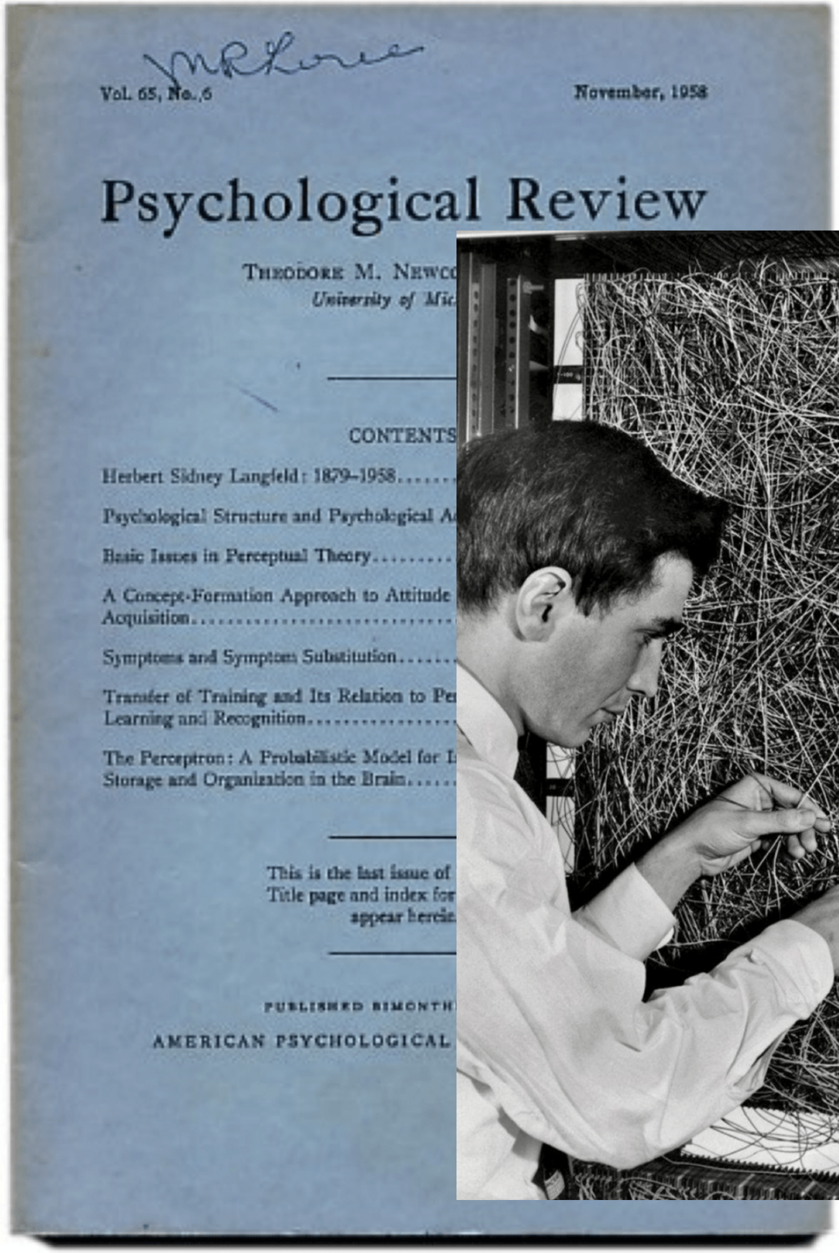
the separator is **linear** in the feature x

Linear classification played a pivotal role in kicking off the first wave of AI enthusiasm.





http://www.ecse.rpi.edu/homepages/nagy/PDF_chrono/2011_Nagy_Pace_FR.pdf. Photo by George Nagy



Vol. 65, No. 6 November, 1958

Psychological Review

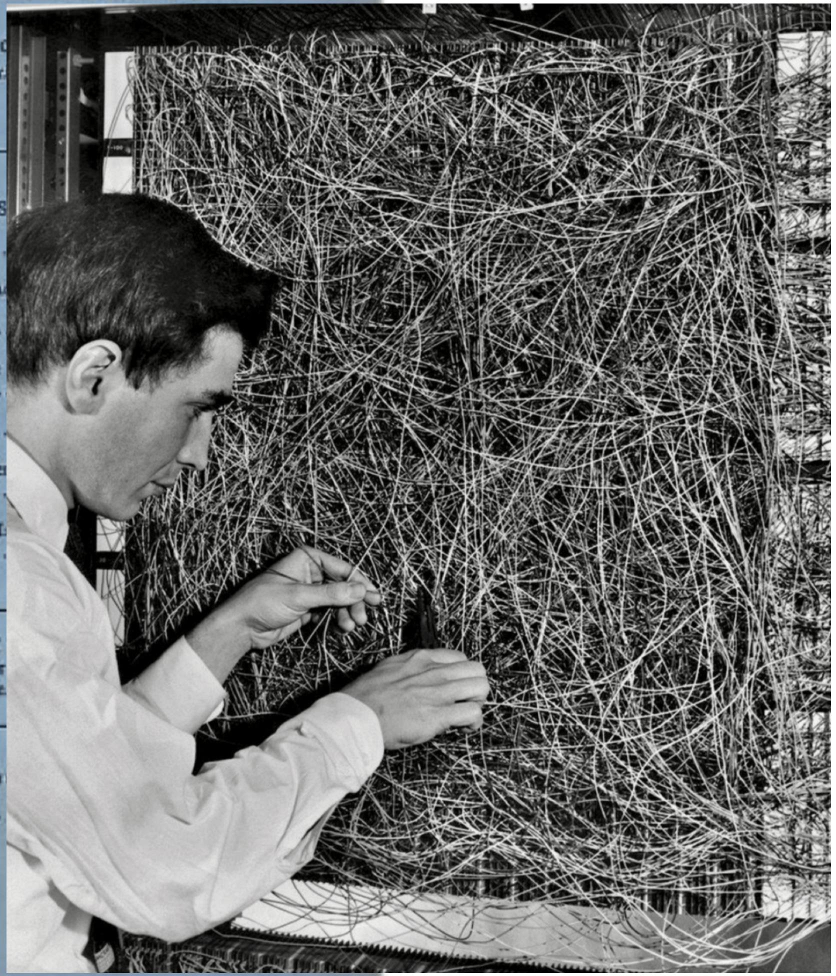
THEODORE M. NEWC
University of Mic

CONTENTS

Herbert Sidney Langfeld: 1879–1958......
Psychological Structure and Psychological A
Basic Issues in Perceptual Theory..........
A Concept-Formation Approach to Attitude Acquisition..............................
Symptoms and Symptom Substitution.......
Transfer of Training and Its Relation to Pe Learning and Recognition...................
The Perceptron: A Probabilistic Model for I Storage and Organization in the Brain......

This is the last issue of
Title page and index for
appear herein

PUBLISHED BIMONTH
AMERICAN PSYCHOLOGICAL



http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.335.3398&rep=rep1&type=pdf

1958 New York Times...

NEW NAVY DEVICE

Psychologist Shows Embryo of Computer Designed to Read and Grow Wiser

WASHINGTON, July 7 (UPI) —The Navy revealed the embryo of an electronic computer today that it expects will be able to walk, talk, see, write, reproduce itself and be conscious of its existence.

The embryo—the Weather

ducted the demonstration. He said the machine would be the first device to think as the human brain. As do human be-

said.

Dr. R
psycholog
Aeronaut
falo, said
fired to t
cal space

Witho

The Na
would be
mechanis
ing, recog
its surro
human tr

The "
remembe

duce themselves on an assembly line and which would be conscious of their existence.

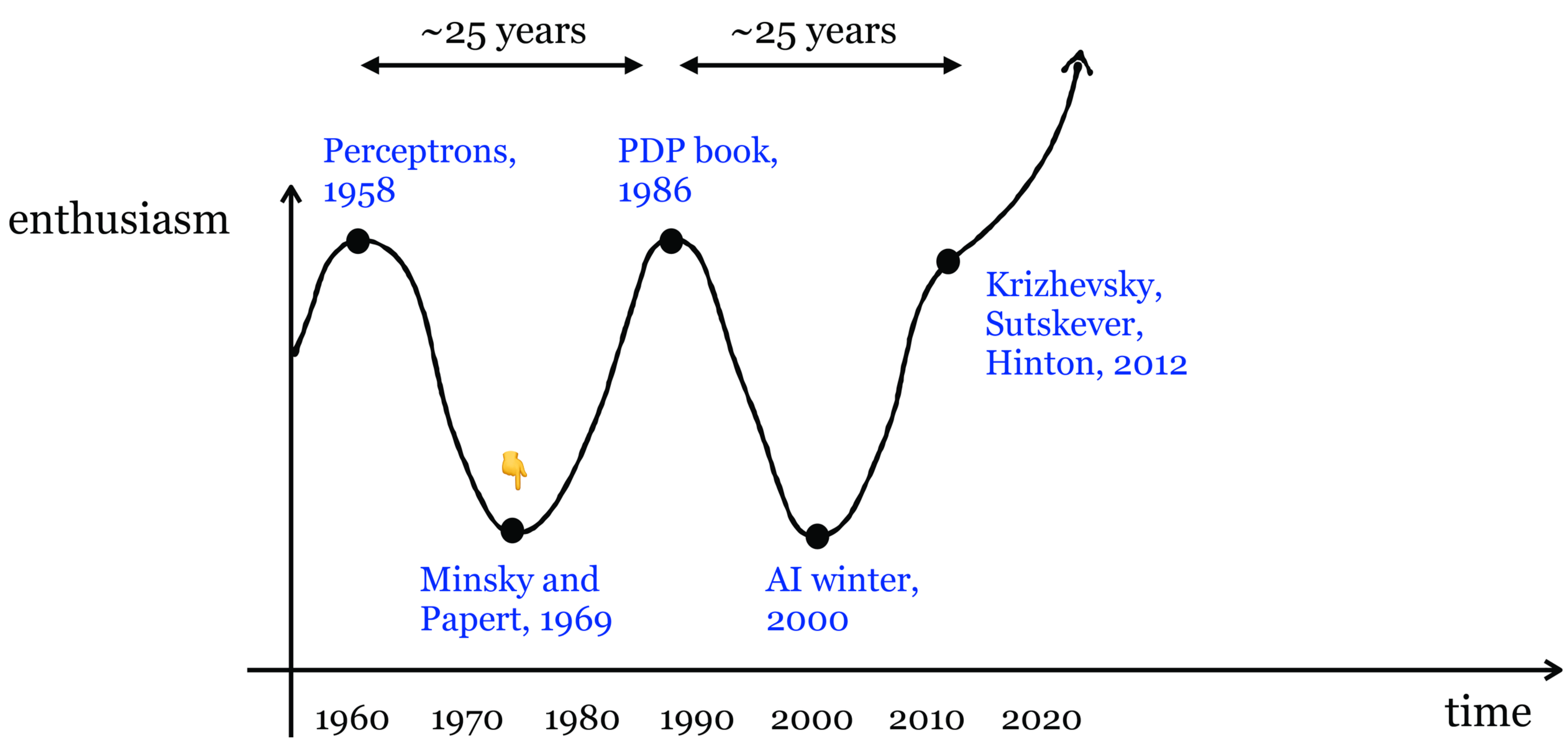
today's demonstration, the " was fed two cards, one squares marked on the left and the other with squares the right side.

Learns by Doing

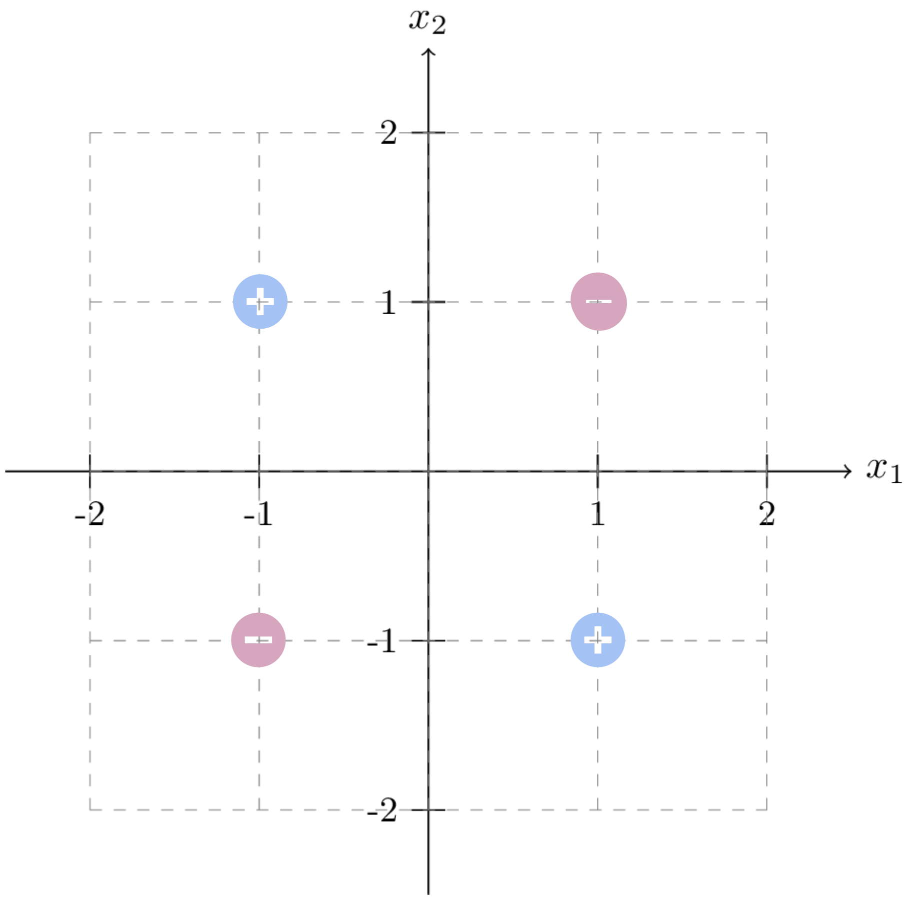
the first fifty trials, the hine made no distinction between them. It then started stering a "Q" for the left res and "O" for the right res.

r. Rosenblatt said he could ain why the machine ned only in highly technical s. But he said the computer undergone a "self-induced ge in the wiring diagram."

he first Perceptron will about 1,000 electronic ociation cells" receiving trical impulses from an eye-scanning device with 400 to-cells. The human brain 10,000,000,000 responsive cells, including 100,000,000 connections with the eyes.


~25 years
~25 years
Perceptrons, 1958
PDP book, 1986
enthusiasm
Krizhevsky, Sutskever, Hinton, 2012
Minsky and Papert, 1969
AI winter, 2000
1960
1970
1980
1990
2000
2010
2020
time

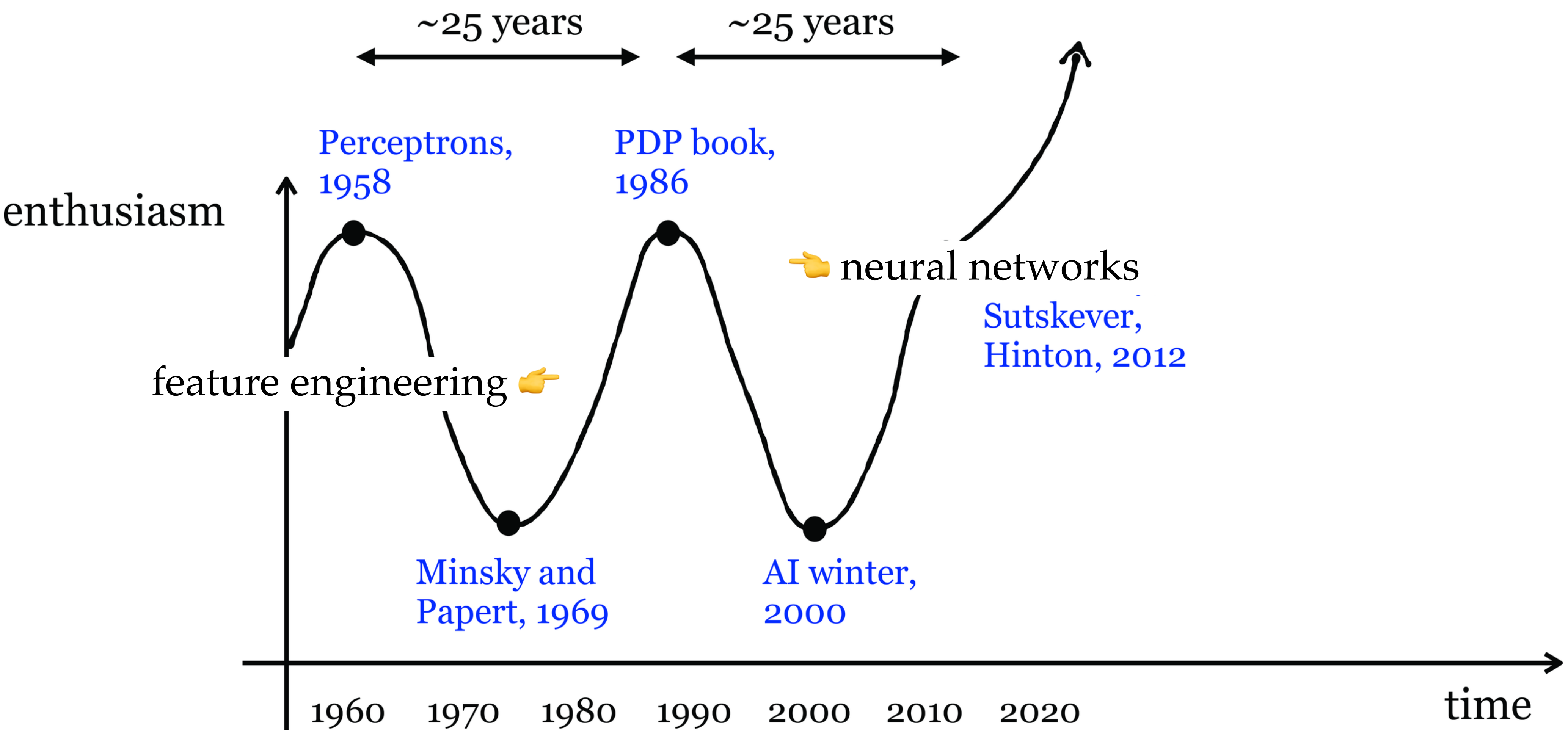
XOR dataset



Not **linearly** separable.

~~Linear tools cannot solve interesting tasks.~~

Linear tools cannot, *by themselves,* solve interesting tasks.

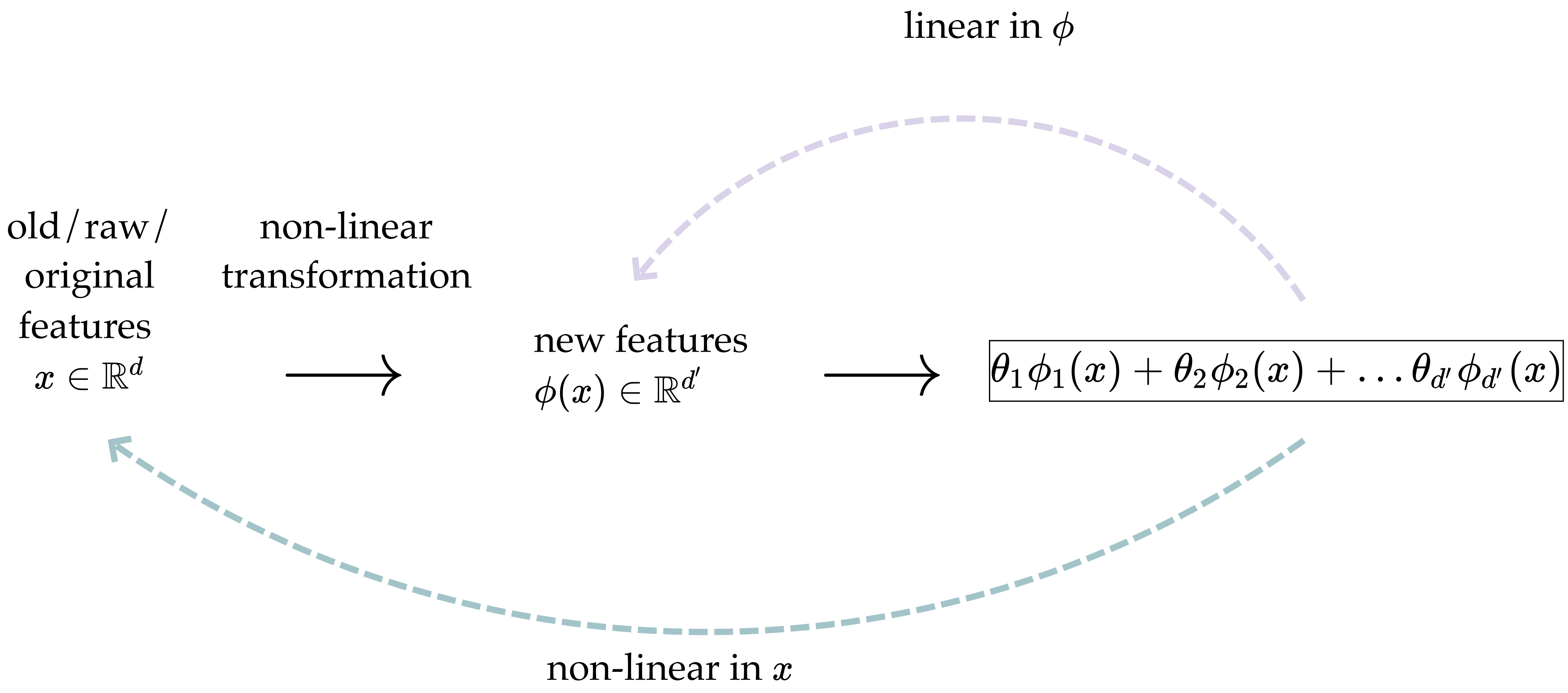

~25 years
~25 years
Perceptrons, 1958
PDP book, 1986
enthusiasm
👈 neural networks
Sutskever, Hinton, 2012
feature engineering 👉
Minsky and Papert, 1969
AI winter, 2000
1960
1970
1980
1990
2000
2010
2020
time

Outline

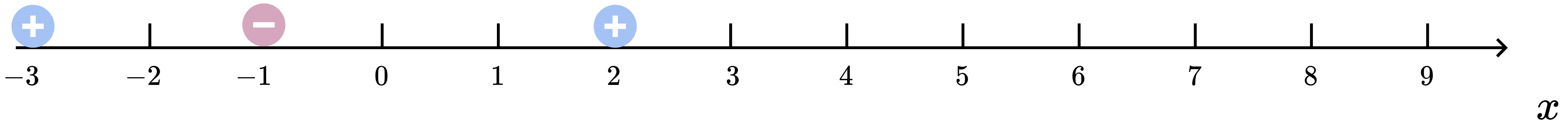
- Systematic feature transformations
  - Engineered features
  - Polynomial features
  - Expressive power
- Neural networks
  - Terminologies
    - neuron, activation function, layer, feedforward network
  - Design choices

Outline

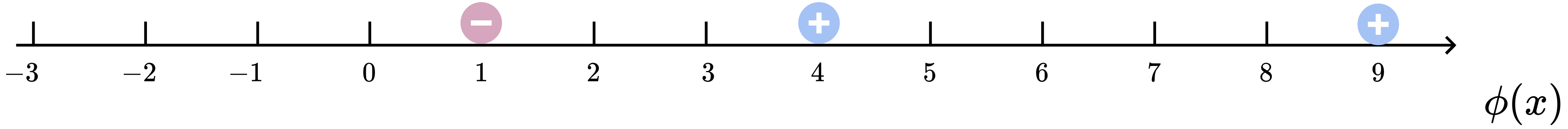
- Systematic feature transformations
  - Engineered features
  - Polynomial features
  - Expressive power
- Neural networks
  - Terminologies
    - neuron, activation function, layer, feedforward network
  - Design choices


linear in ϕ
old / raw / original features
$x \in \mathbb{R}^d$
non-linear transformation
new features
$\phi(x) \in \mathbb{R}^{d'}$
$\theta_1 \phi_1(x) + \theta_2 \phi_2(x) + \ldots \theta_{d'} \phi_{d'}(x)$
non-linear in x

Not linearly separable in x space

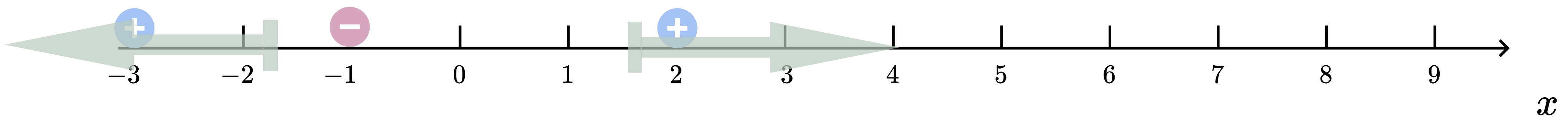


$\Downarrow$ transform via $\phi(x) = x^2$

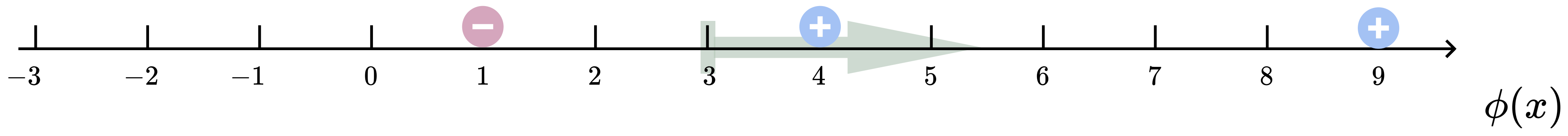


Linearly separable in $\phi(x) = x^2$ space

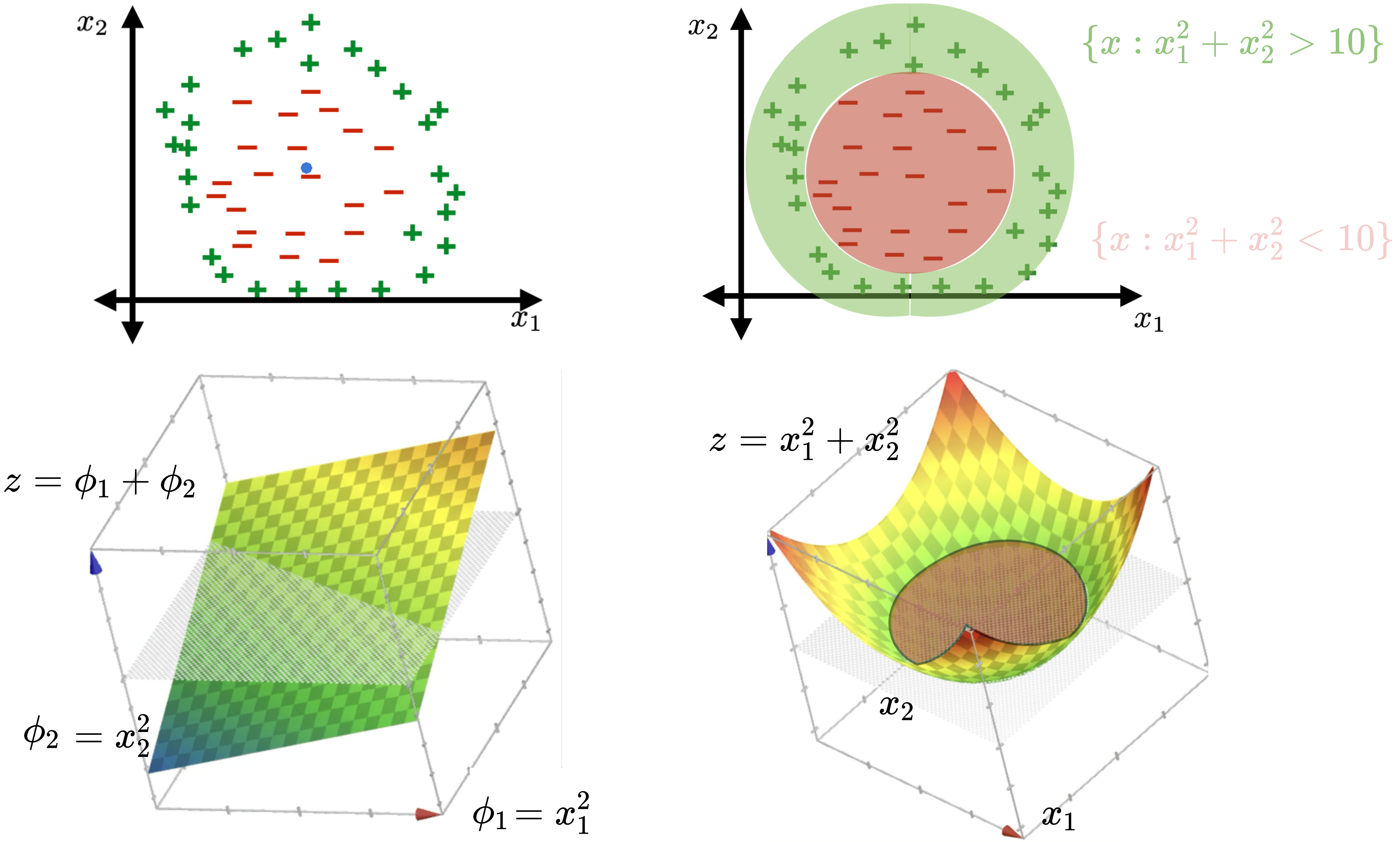
Non-linearly separated in x space, e.g. predict positive if $x^2 \geq 3$



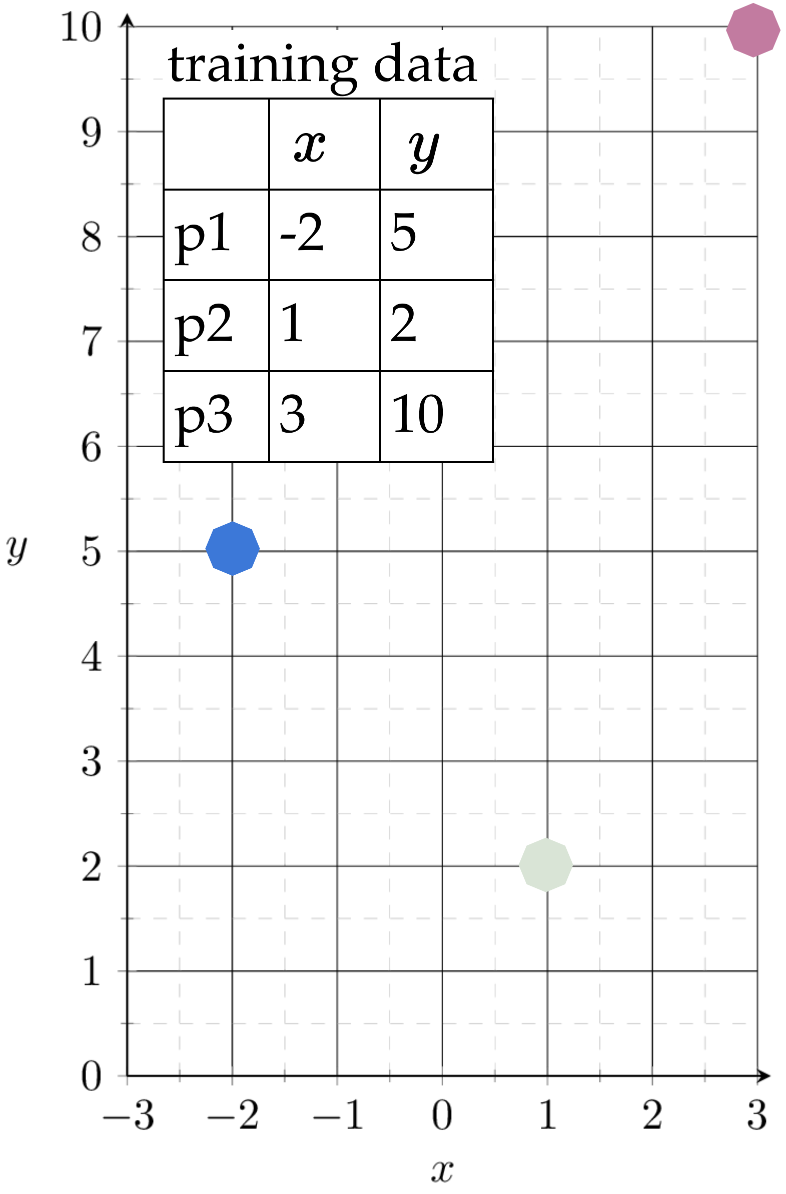
$\Downarrow$ transform via $\phi(x) = x^2$



Linearly separable in $\phi(x) = x^2$ space, e.g. predict positive if $\phi \geq 3$



x_2
x_1
x_2
x_1
$\{x : x_1^2 + x_2^2 > 10\}$
$\{x : x_1^2 + x_2^2 < 10\}$
$z = \phi_1 + \phi_2$
$\phi_2 = x_2^2$
$\phi_1 = x_1^2$
$z = x_1^2 + x_2^2$
x_2
x_1



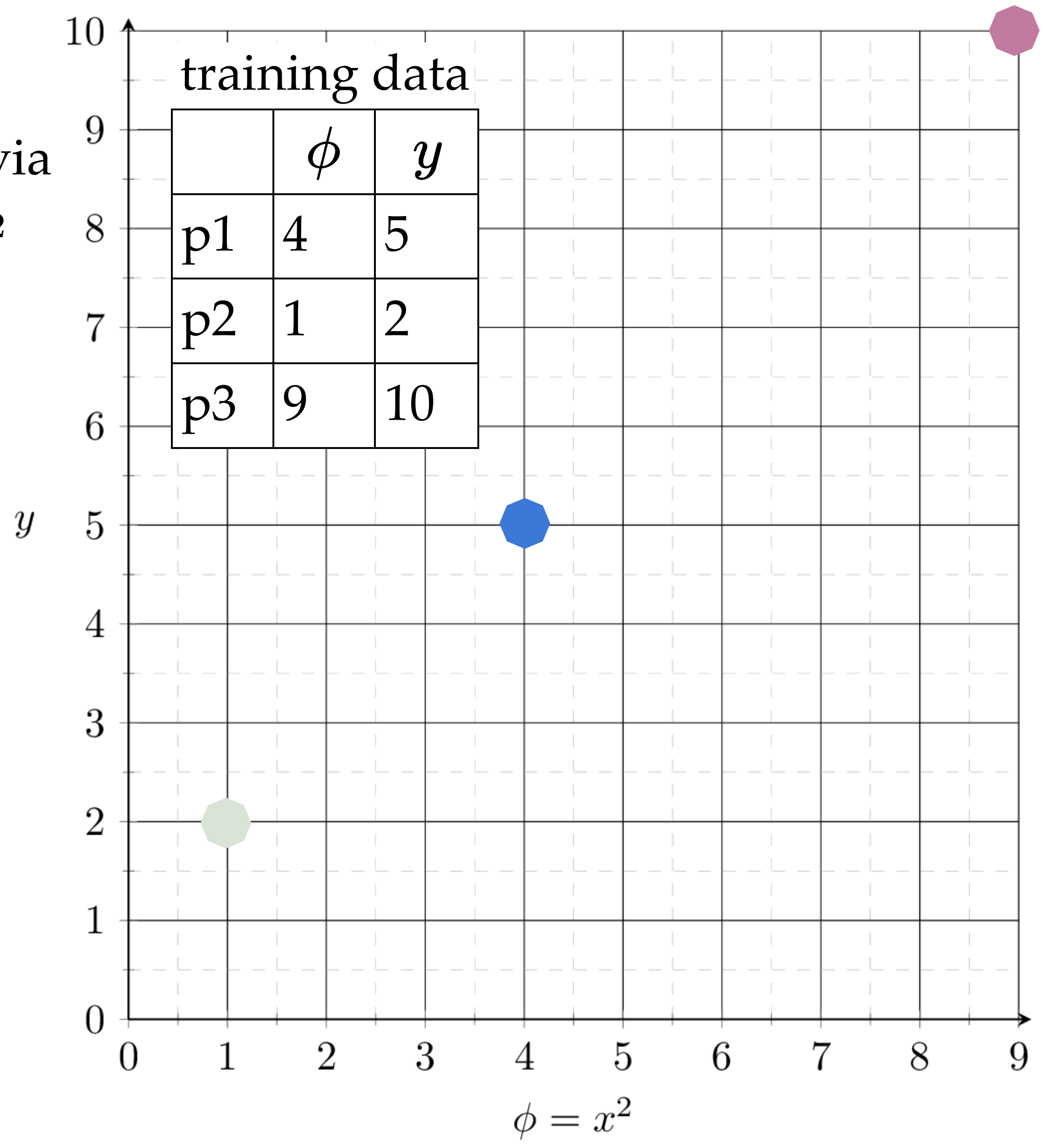
training data

	x	y
p1	-2	5
p2	1	2
p3	3	10

transform via

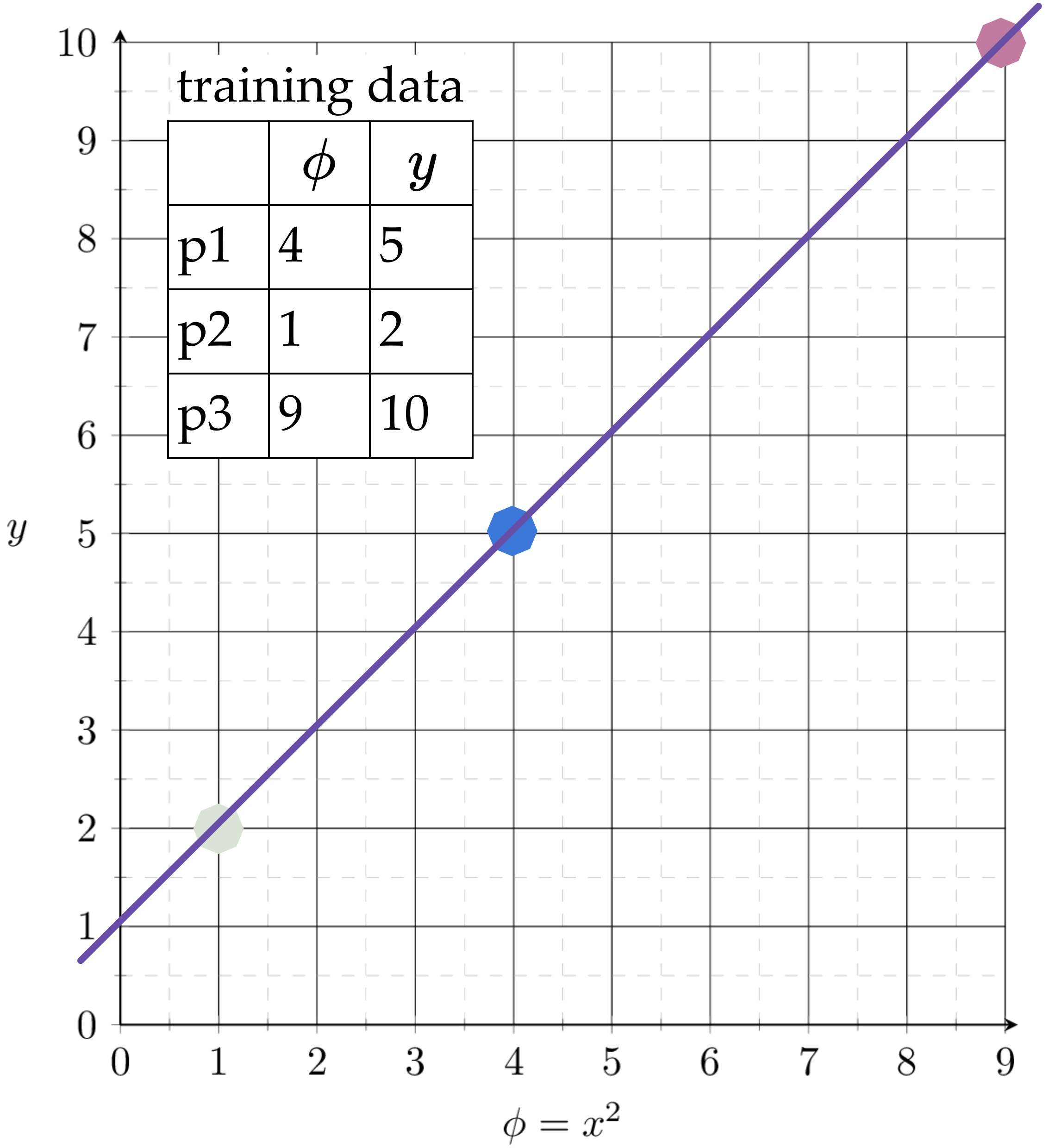
$\phi(x) = x^2$

$\Longrightarrow$



training data

	ϕ	y
p1	4	5
p2	1	2
p3	9	10

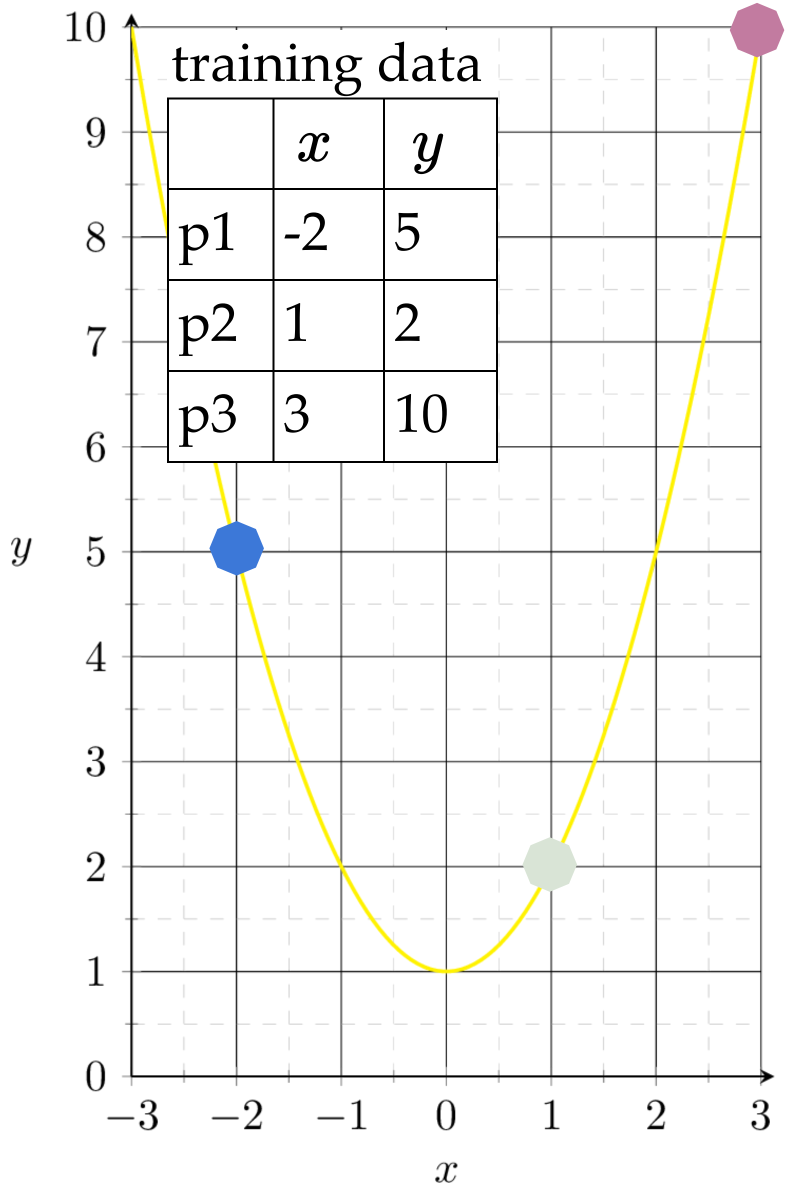


	ϕ	y
p1	4	5
p2	1	2
p3	9	10

$$y = \phi + 1$$

$$= x^2 + 1$$

$\Rightarrow$



	x	y
p1	-2	5
p2	1	2
p3	3	10

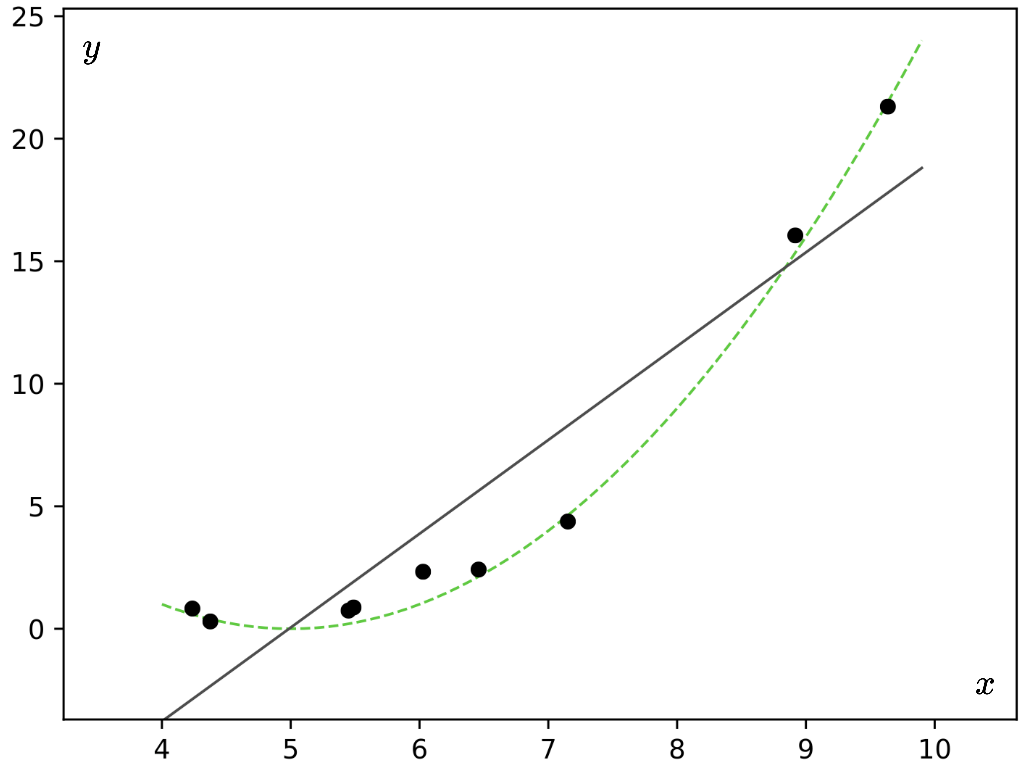
systematic polynomial features construction

	$d = 1$	$d = 2$
	x_1	x_1, x_2
$k = 0$	1	1
$k = 1$	$1, x_1$	$1, x_1, x_2$
$k = 2$	$1, x_1, x_1^2$	$1, x_1, x_2, x_1^2, x_1 x_2, x_2^2$
$k = 3$	$1, x_1, x_1^2, x_1^3$	$1, x_1, x_2, x_1^2, x_1 x_2, x_2^2, x_1^3, x_1^2 x_2, x_1 x_2^2, x_2^3$
$\ldots$		

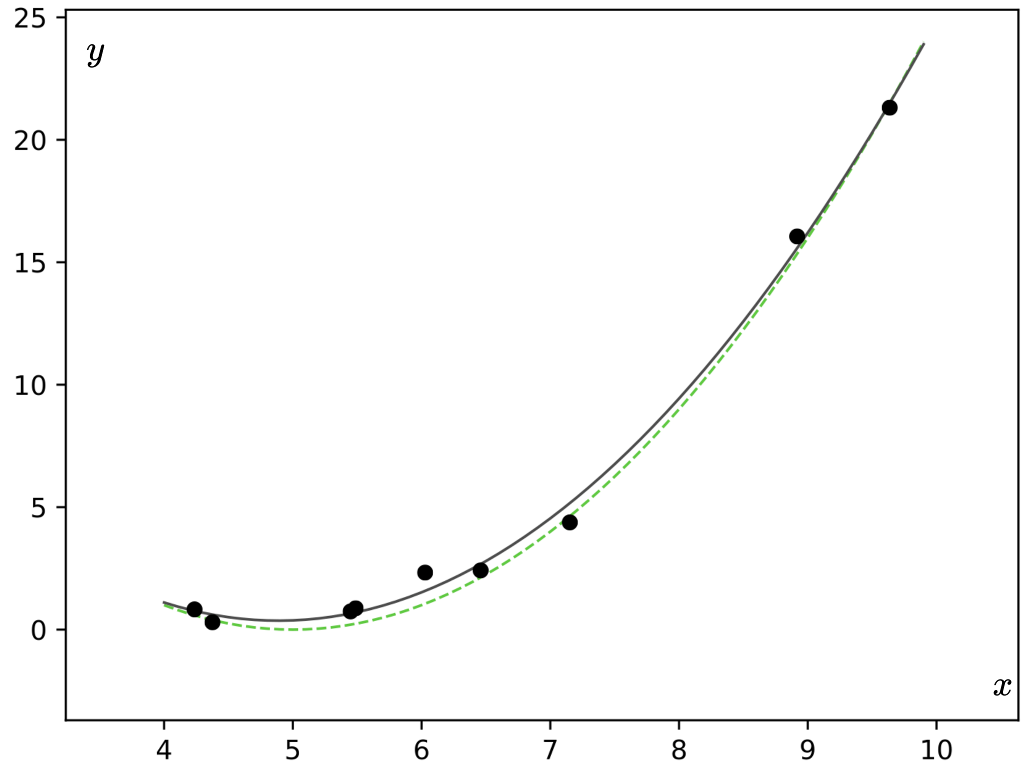
- Elements in the basis are the monomials of original features raised up to power k
- With a given d and a fixed k, the basis is **fixed**.

9 data points; each data point has a feature $x \in \mathbb{R}$, label $y \in \mathbb{R}$

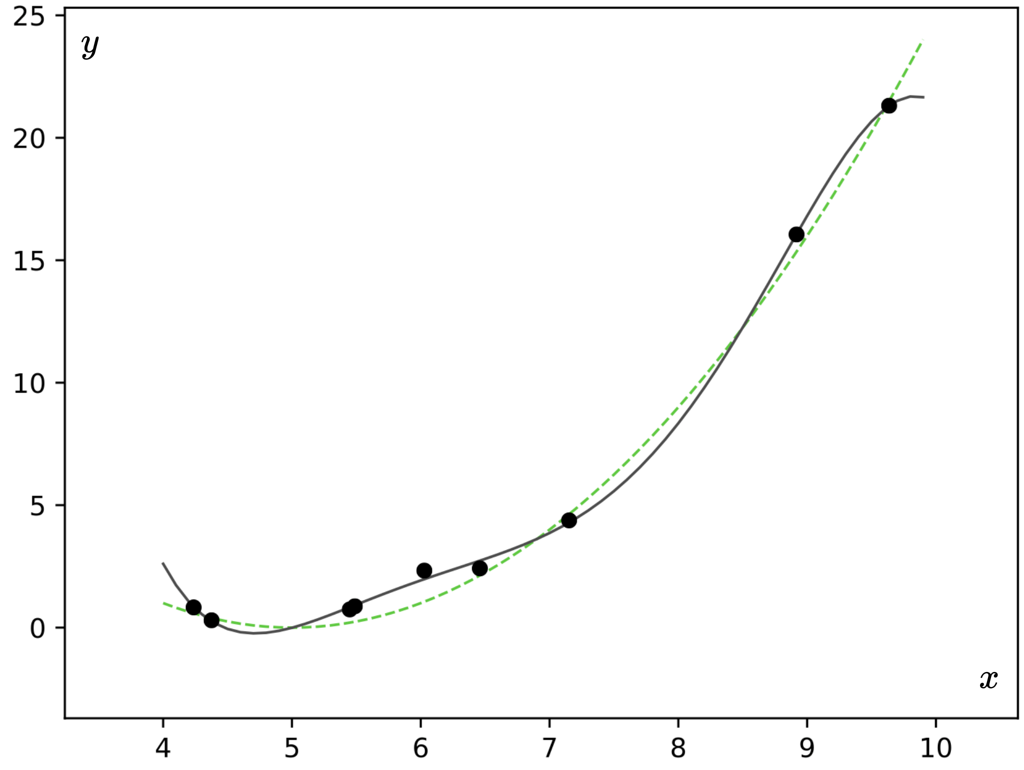
generated from green dashed line



- $k = 1$
- $h(x; \theta) = \theta_0 + \theta_1 x$
- Learn 2 parameters for linear function

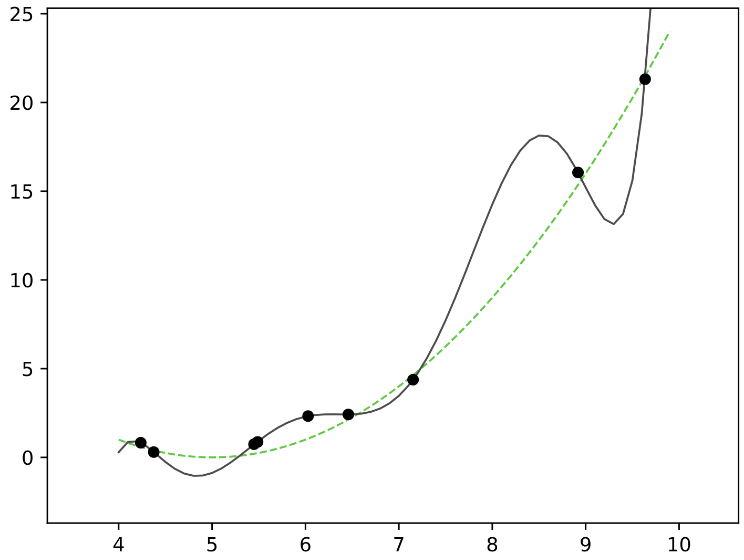


- Choose $k = 2$
- New features $\phi = [1; x; x^2]$
- $h(x; \theta) = \theta_0 + \theta_1 x + \theta_2 x^2$
- Learn 3 parameters for quadratic function



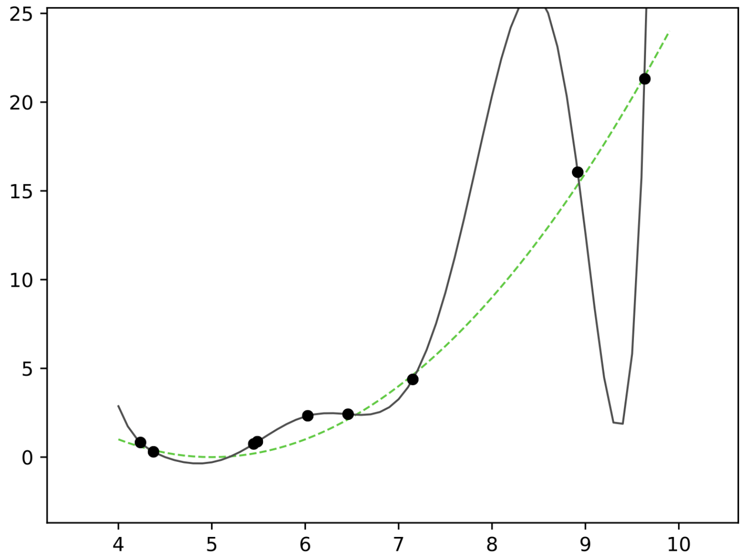
- Choose $k = 5$
- New features $\phi = [1; x; x^2; x^3; x^4; x^5]$
- $h(x; \theta) = \theta_0 + \theta_1 x + \theta_2 x^2 + \theta_3 x^3 + \theta_4 x^4 + \theta_5 x^5$
- Learn 6 parameters for degree-5 polynomial function

$k = 7$



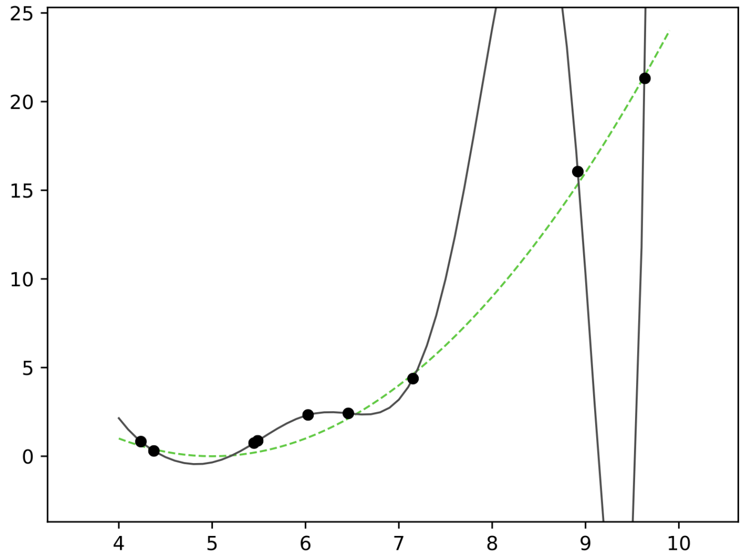
25
20
15
10
5
0
4
5
6
7
8
9
10

$k = 8$



25
20
15
10
5
0
4
5
6
7
8
9
10

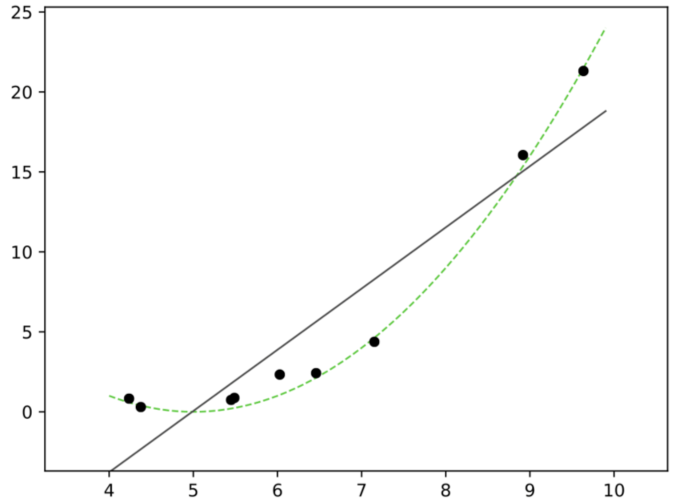
$k = 10$



25
20
15
10
5
0
4
5
6
7
8
9
10

Underfitting

$$k = 1$$

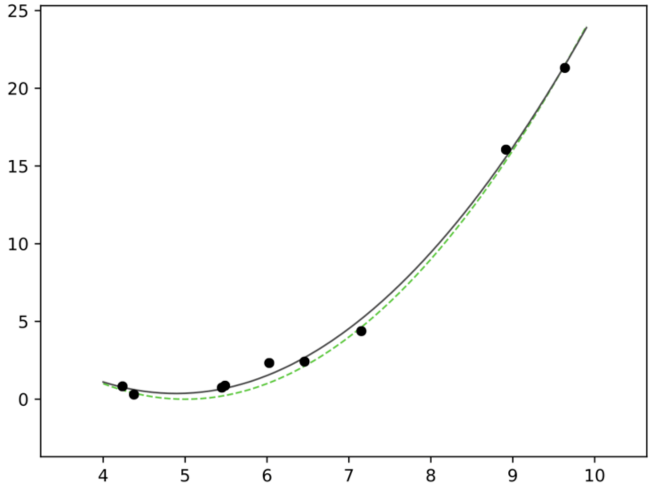


high error on train set

high error on test set

Appropriate model

$$k = 2$$

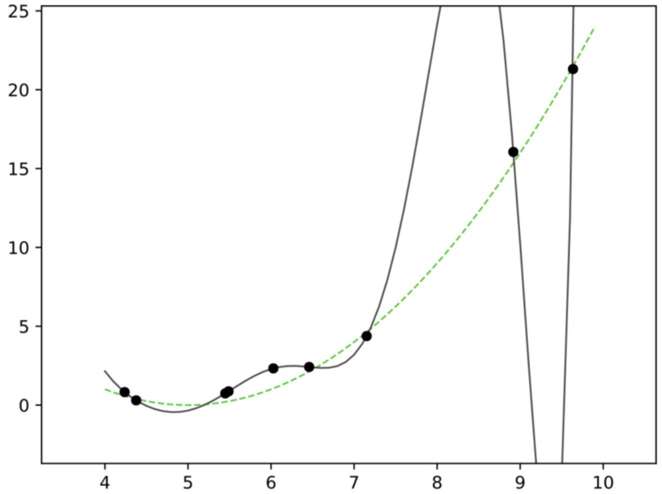


low error on train set

low error on test set

Overfitting

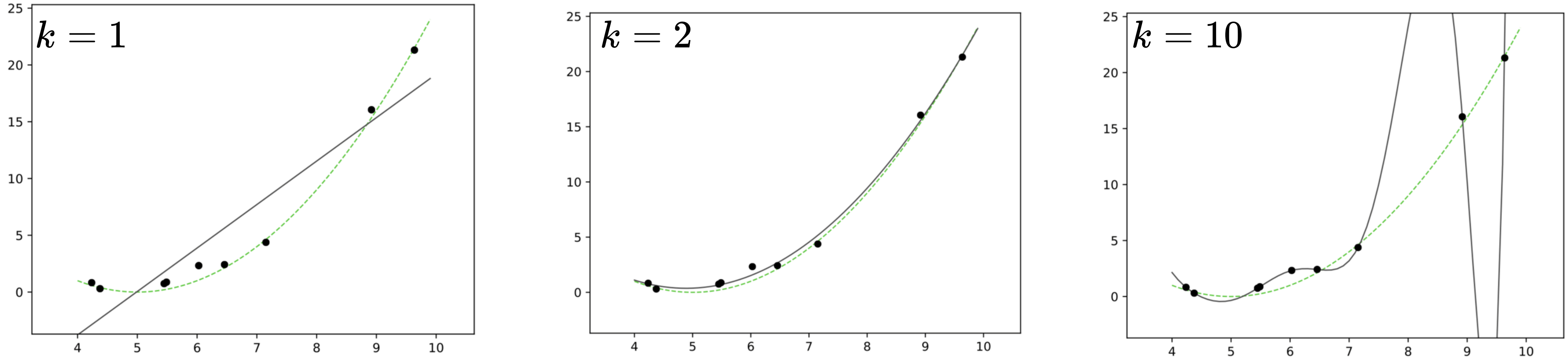
$$k = 10$$



low error on train set

high error on test set

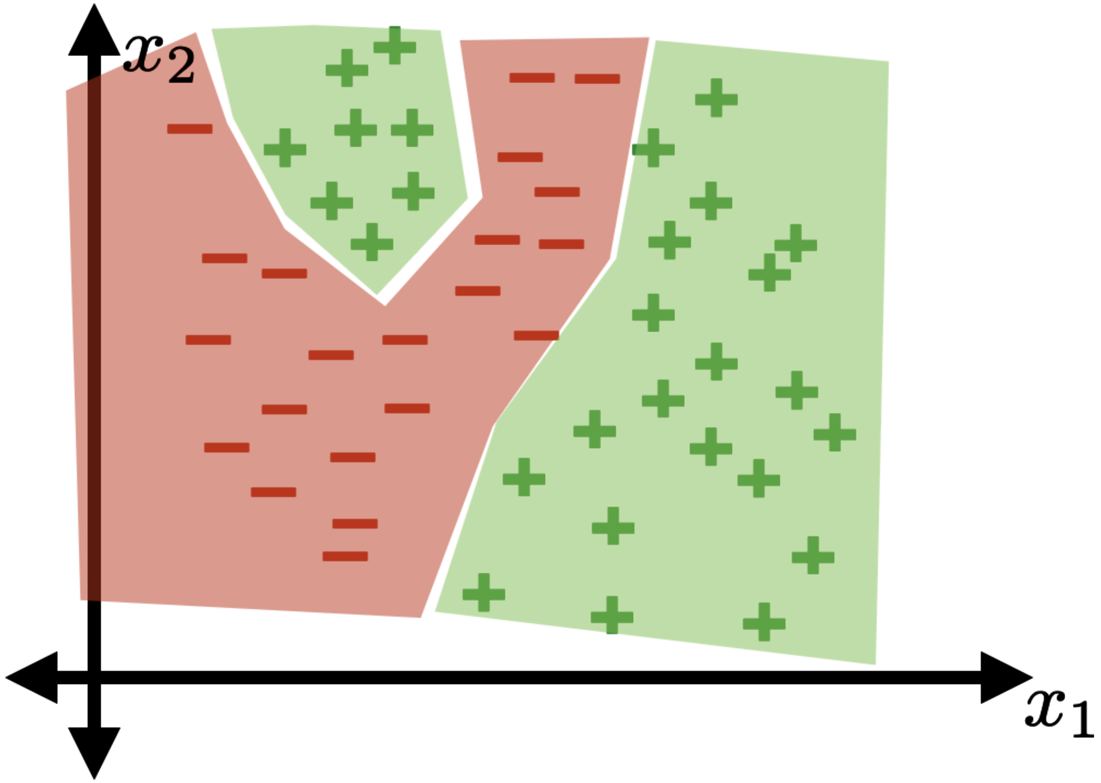
Underfitting | Appropriate model | Overfitting

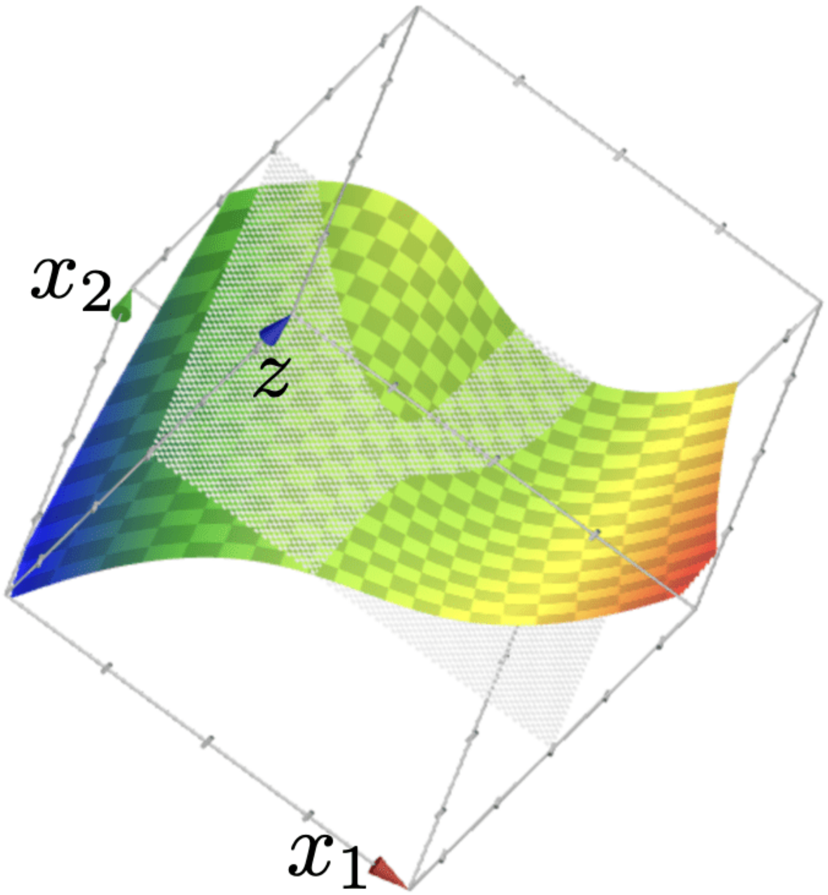


- k : a hyperparameter that determines the capacity (expressiveness) of the hypothesis class.
- Models with many rich features and free parameters tend to have high capacity but also greater risk of overfitting.
- How to choose k? Validation/cross-validation.

Similar overfitting can happen in classification

Using polynomial features of order 3





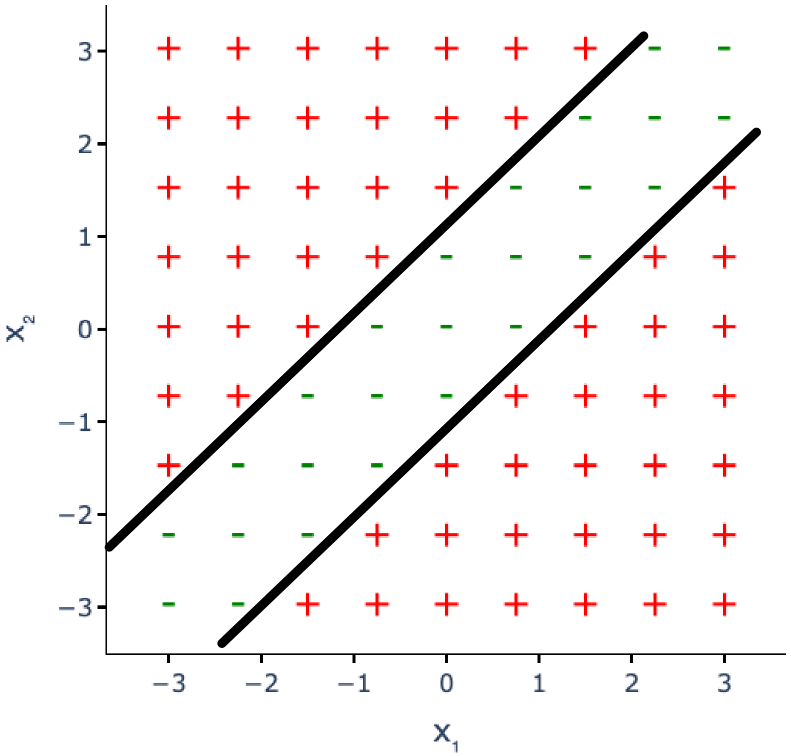
Quick summary:

- Linear models are mathematically and algorithmically convenient but not expressive enough -- by themselves -- for most jobs.
- We can express really rich hypothesis classes by performing a **fixed** non-linear feature transformation first, then applying our linear regression or classification methods.
- Can think of fixed transformation as "adapters", enabling us to use old tools in broader situations.
- Standard feature transformations: polynomials, absolute-value functions.
- For a significant period, the essence of machine learning revolved around **feature engineering**—manually designing transformations to extract useful representations.

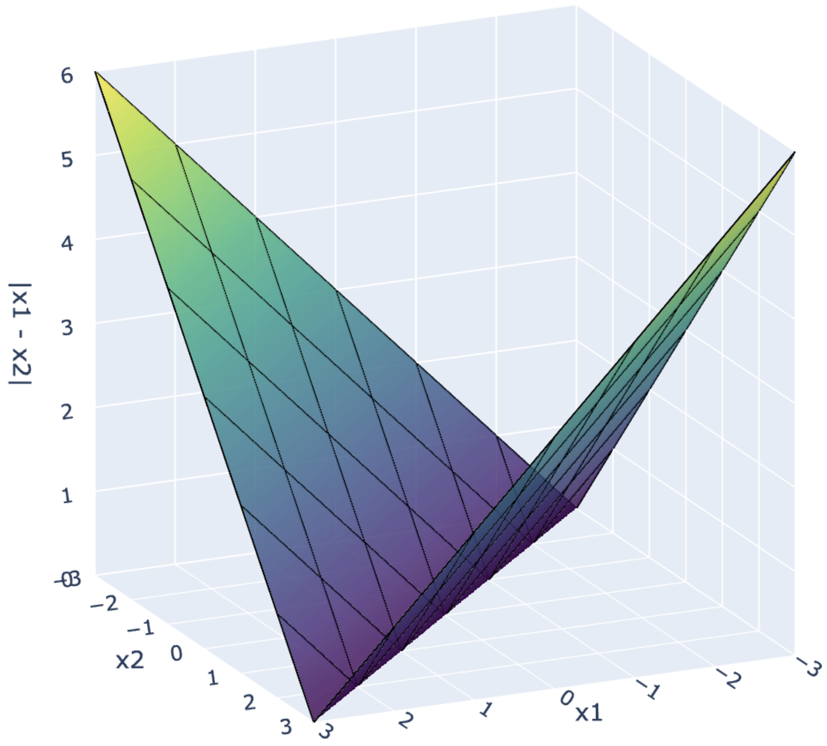
Outline

- Systematic feature transformations
  - Engineered features
  - Polynomial features
  - Expressive power
- Neural networks
  - Terminologies
    - neuron, activation function, layer, feedforward network
  - Design choices

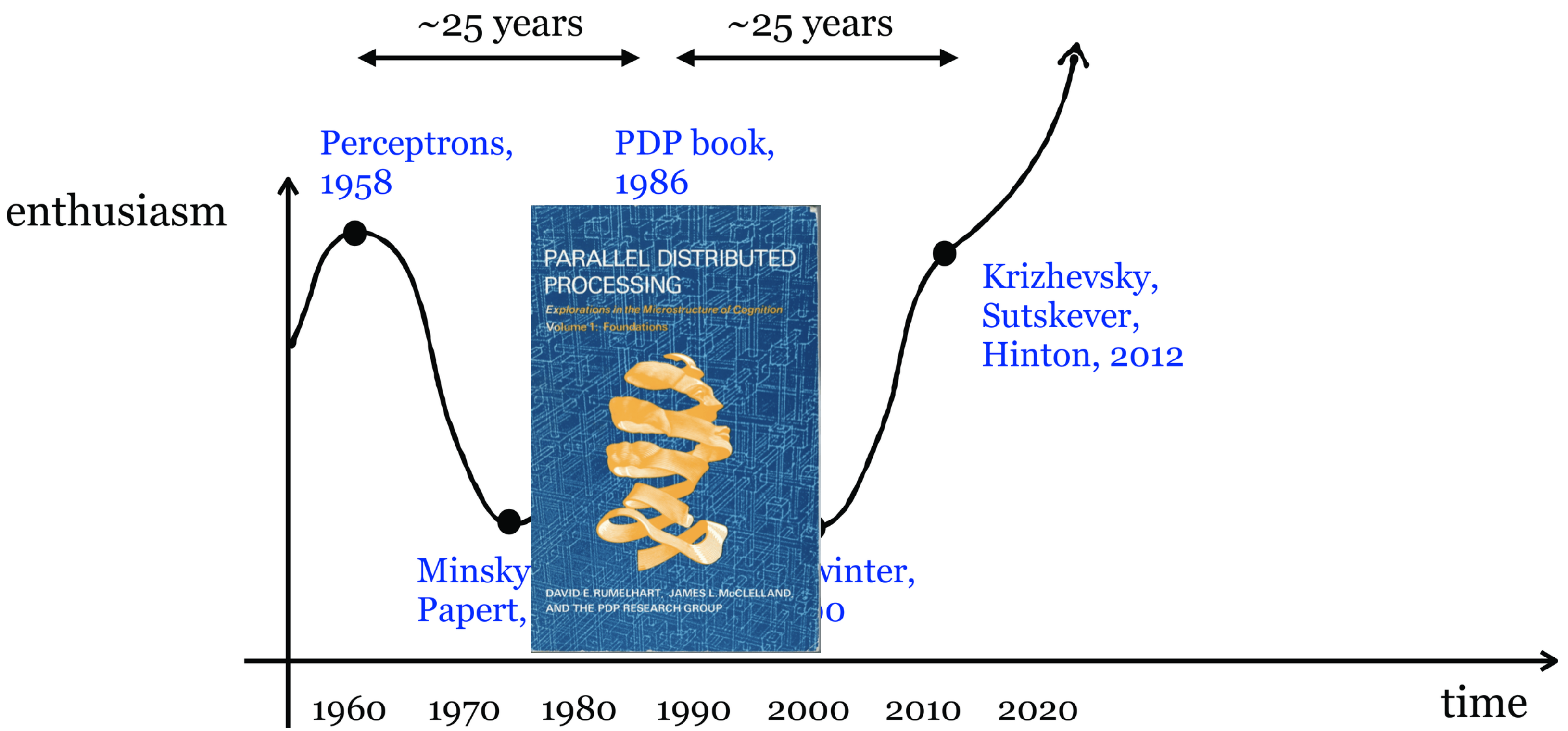
leveraging nonlinear transformations



transform via $\phi([x_1; x_2]) = [1; |x_1 - x_2|]$



importantly, linear in ϕ, non-linear in x



Outlined the fundamental concepts of neural networks:

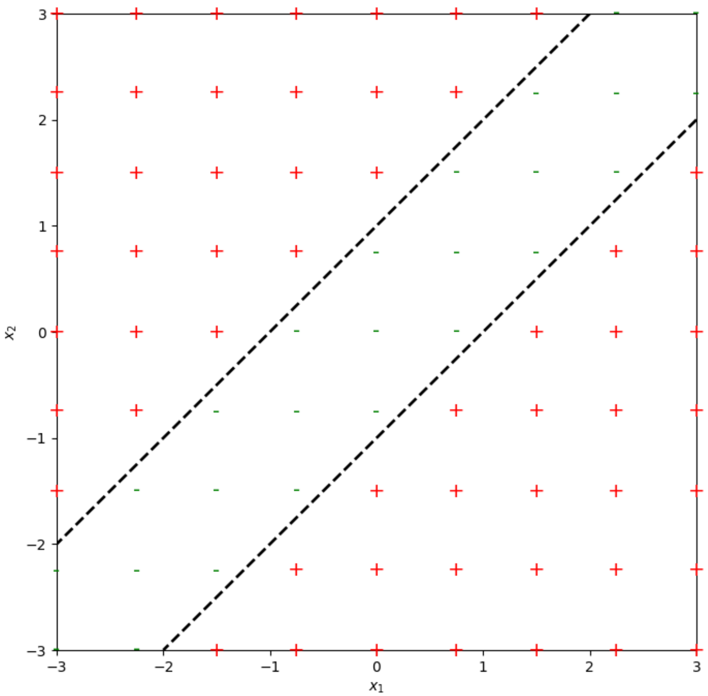
- Nonlinear feature transformation
- "Composing" simple transformations

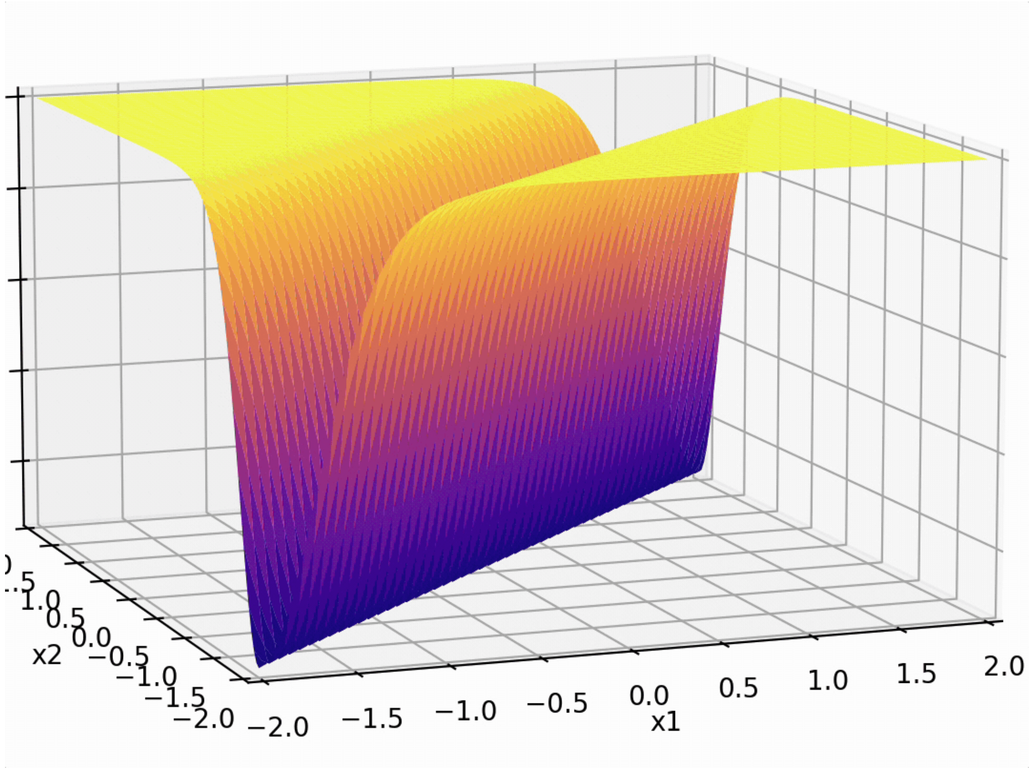
} expressiveness

- Backpropagation

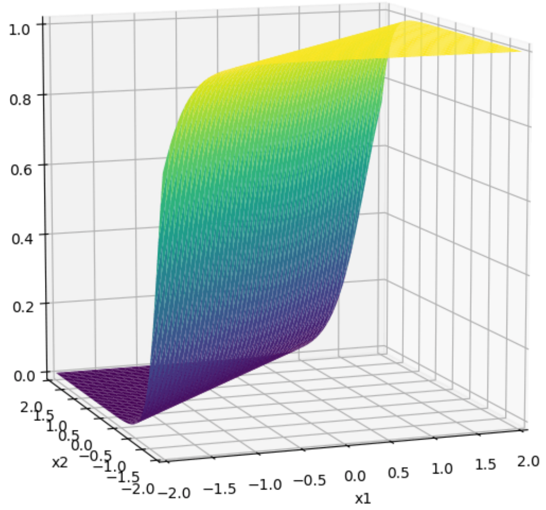
efficient learning

- "Composing" simple transformations

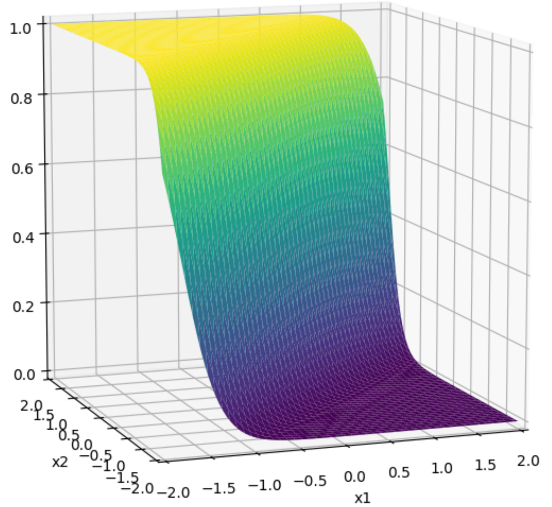




$$\sigma_1 = \sigma(5x_1 - 5x_2 + 1)$$



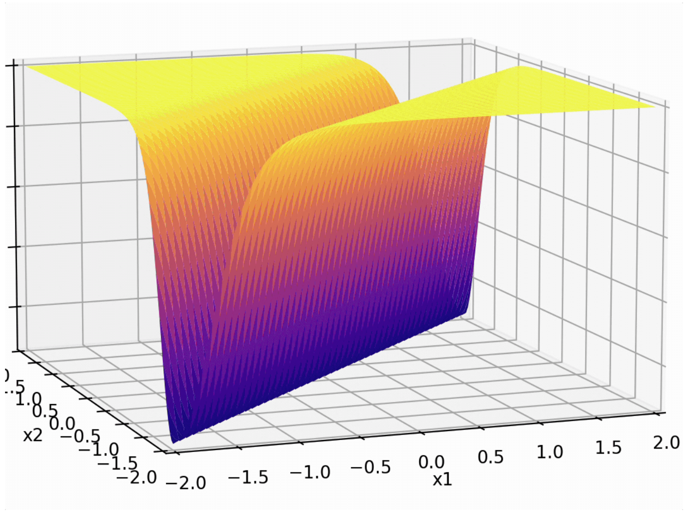
$$\sigma_2 = \sigma(-5x_1 + 5x_2 + 1)$$

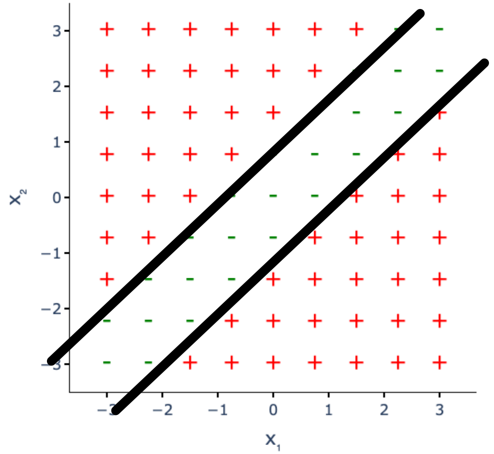


some appropriately weighted sum

Two epiphanies:

- nonlinear transformation empowers linear tools
- "composing" simple nonlinearities *amplifies* such effect





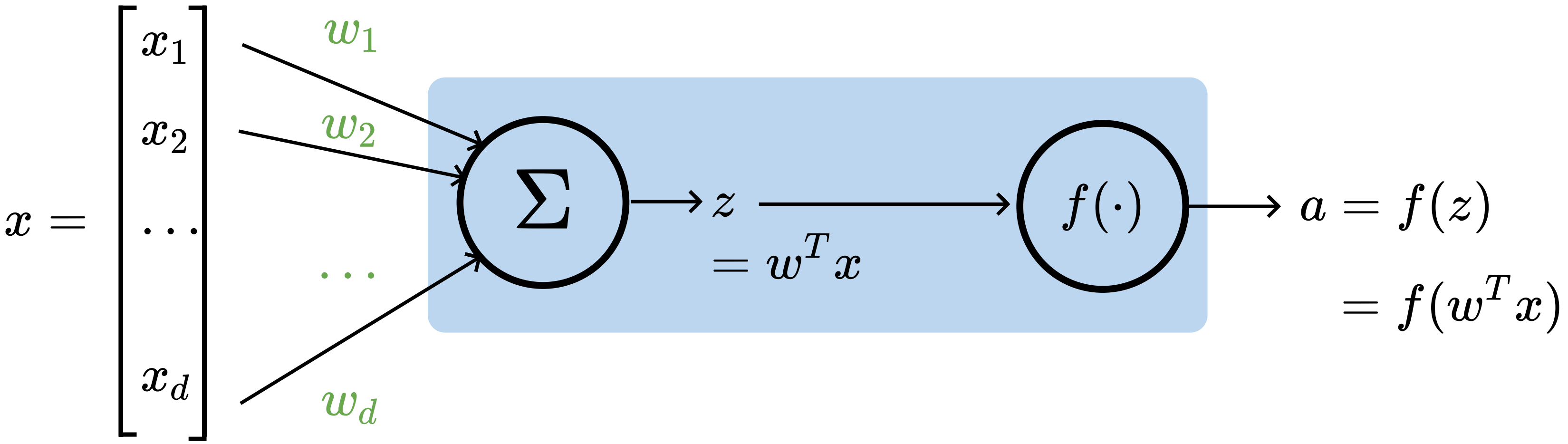
Outline

- Systematic feature transformations
  - Engineered features
  - Polynomial features
  - Expressive power
- Neural networks
  - Terminologies
    - neuron, activation function, layer, feedforward network
  - Design choices

👋 heads-up:

all neural network diagrams focus on a single data point

A neuron:



- x: d-dimensional input
- w: weights (i.e. parameters)
- z: pre-activation output
- f: activation function
- a: post-activation output

w: what the algorithm learns

z: scalar

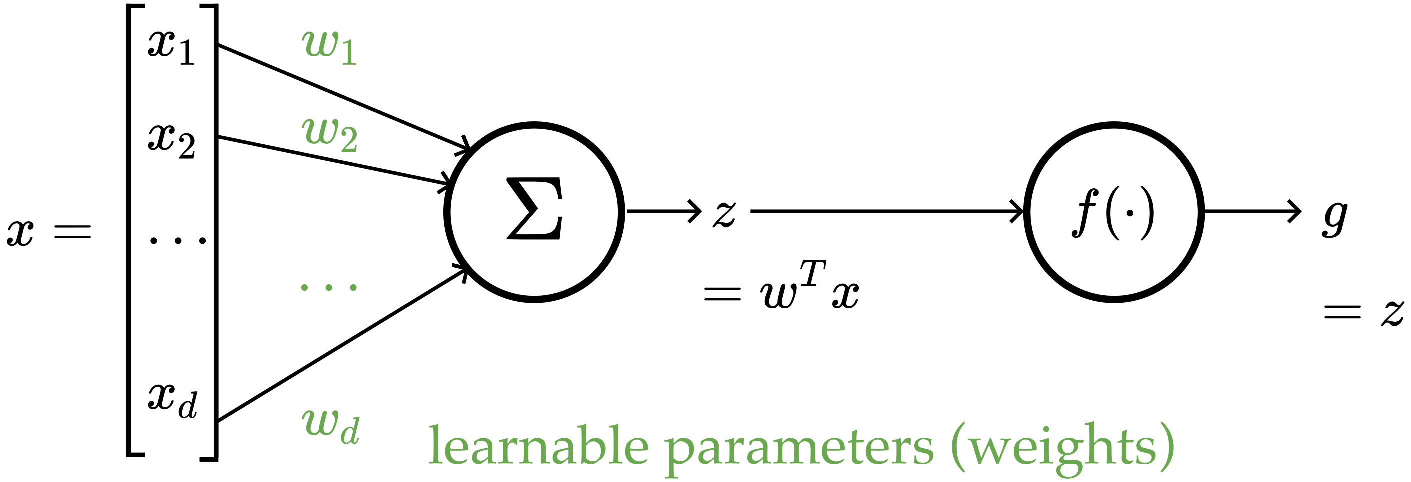
↓

f: what we engineers choose

↓

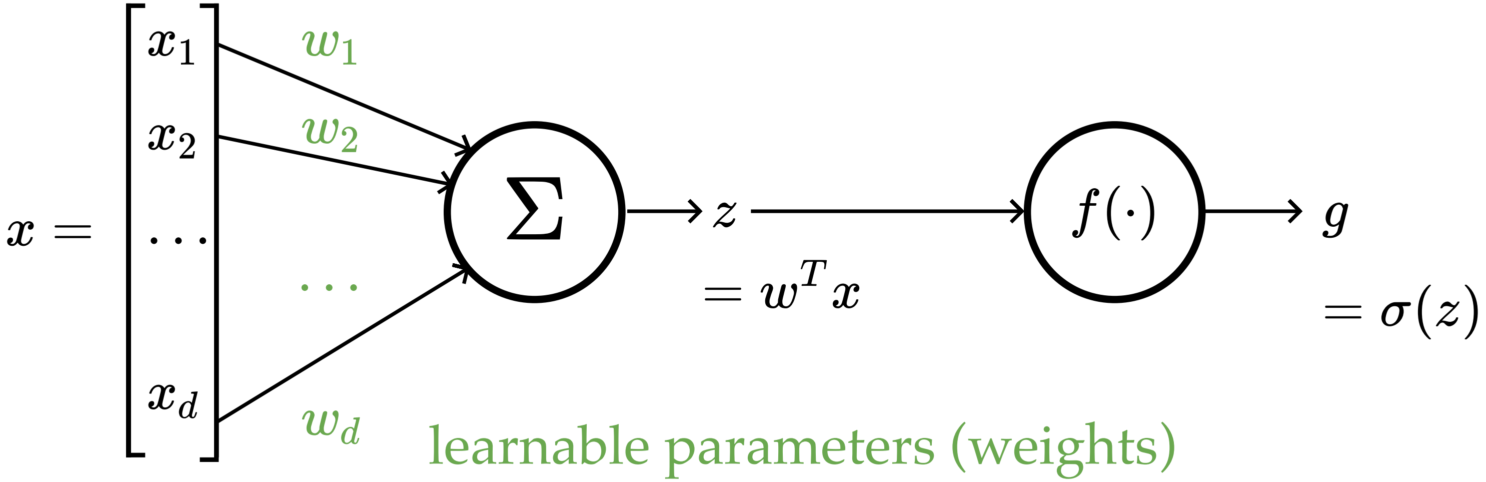
a: scalar

e.g. linear regressor represented as a computation graph



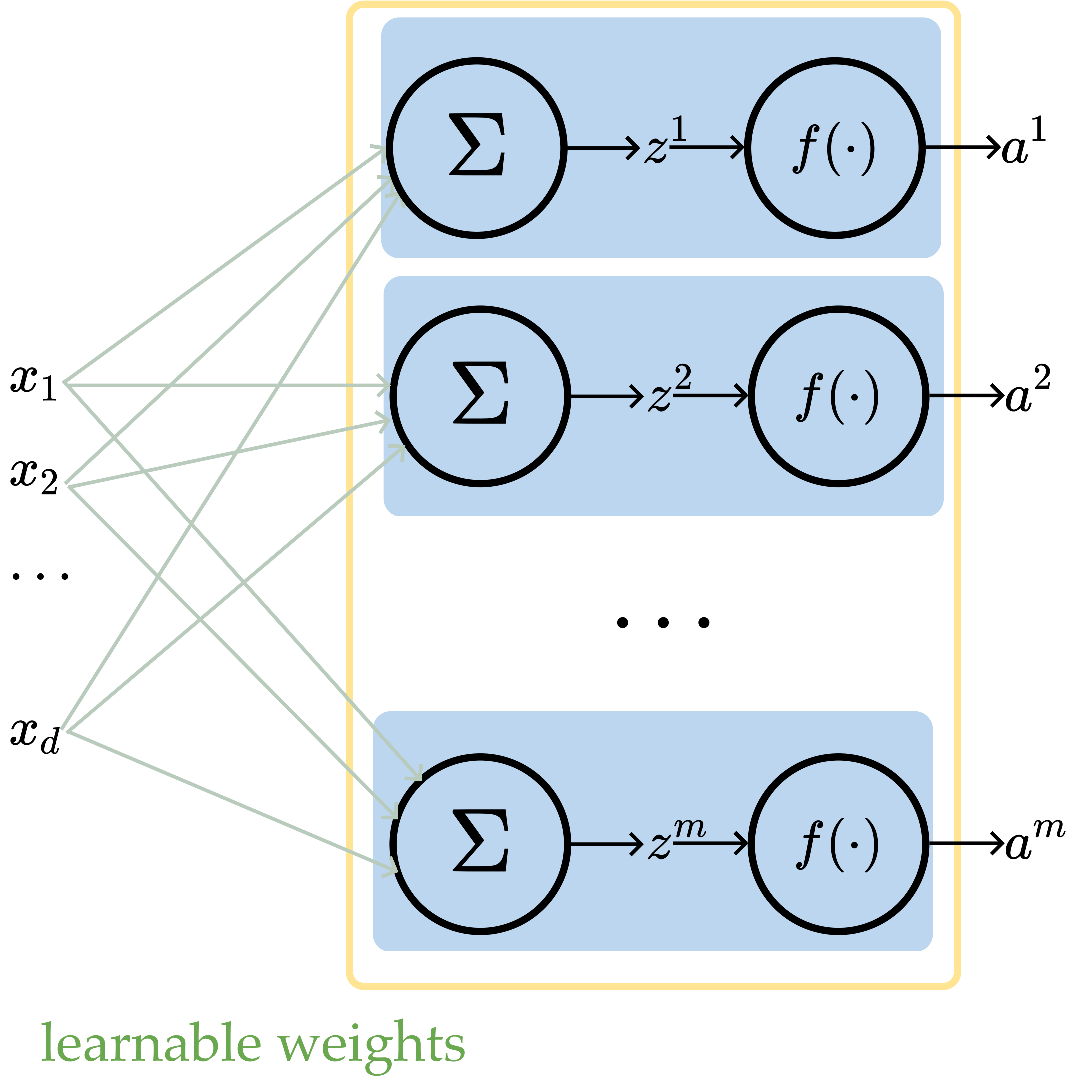
Choose activation $f(z) = z$

e.g. linear logistic classifier represented as a computation graph



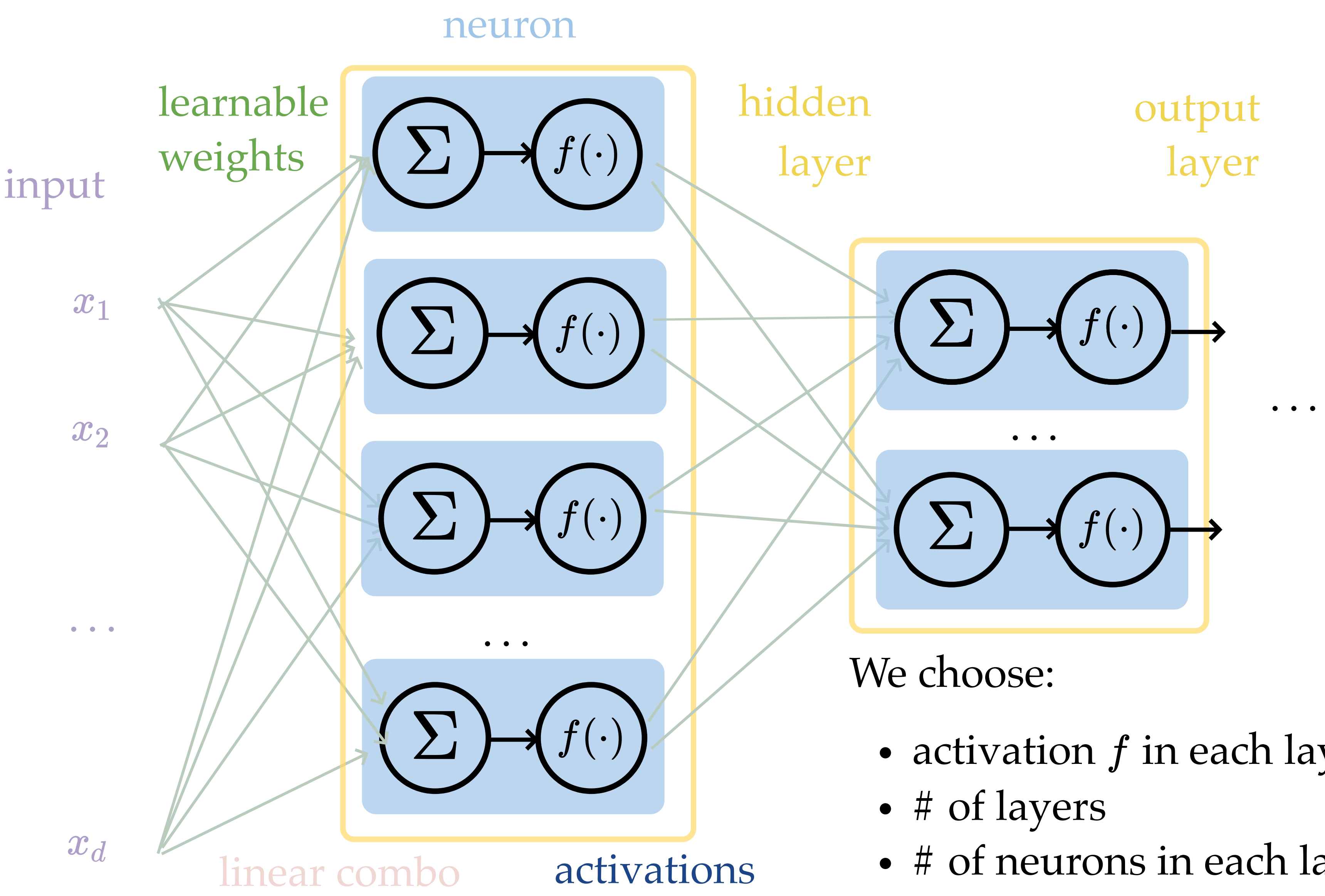
Choose activation $f(z) = \sigma(z)$

A layer:



- (# of neurons) = (layer's output dimension).
- typically, all neurons in one layer use the same activation f (if not, uglier algebra).
- typically fully connected, where all x_i are connected to all z^j, meaning each x_i influences every a^j eventually.
- typically, no "cross-wiring", meaning e.g. z^1 won't affect a^2. (the output layer may be an exception if softmax is used.)

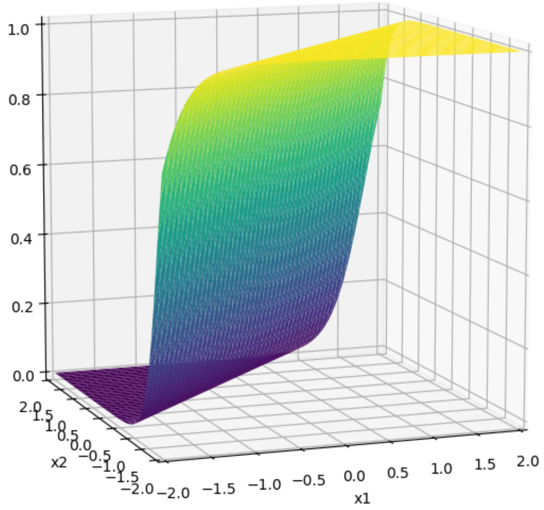
A (fully-connected, feed-forward) neural network:



We choose:

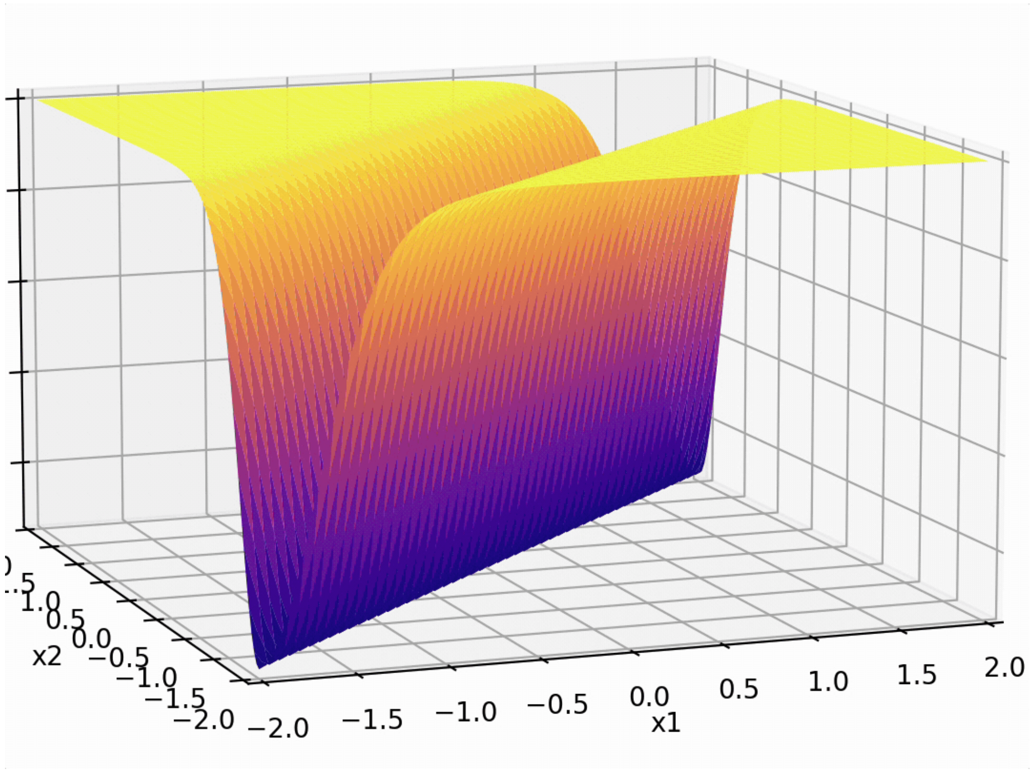
- activation f in each layer
- # of layers
- # of neurons in each layer

$$\sigma_1 = \sigma(5x_1 - 5x_2 + 1)$$

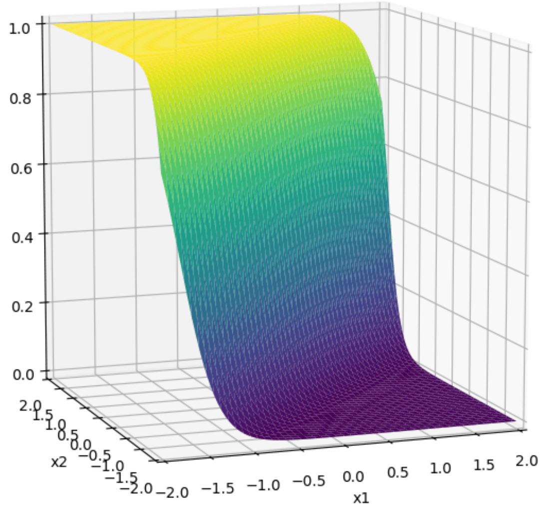


recall this example

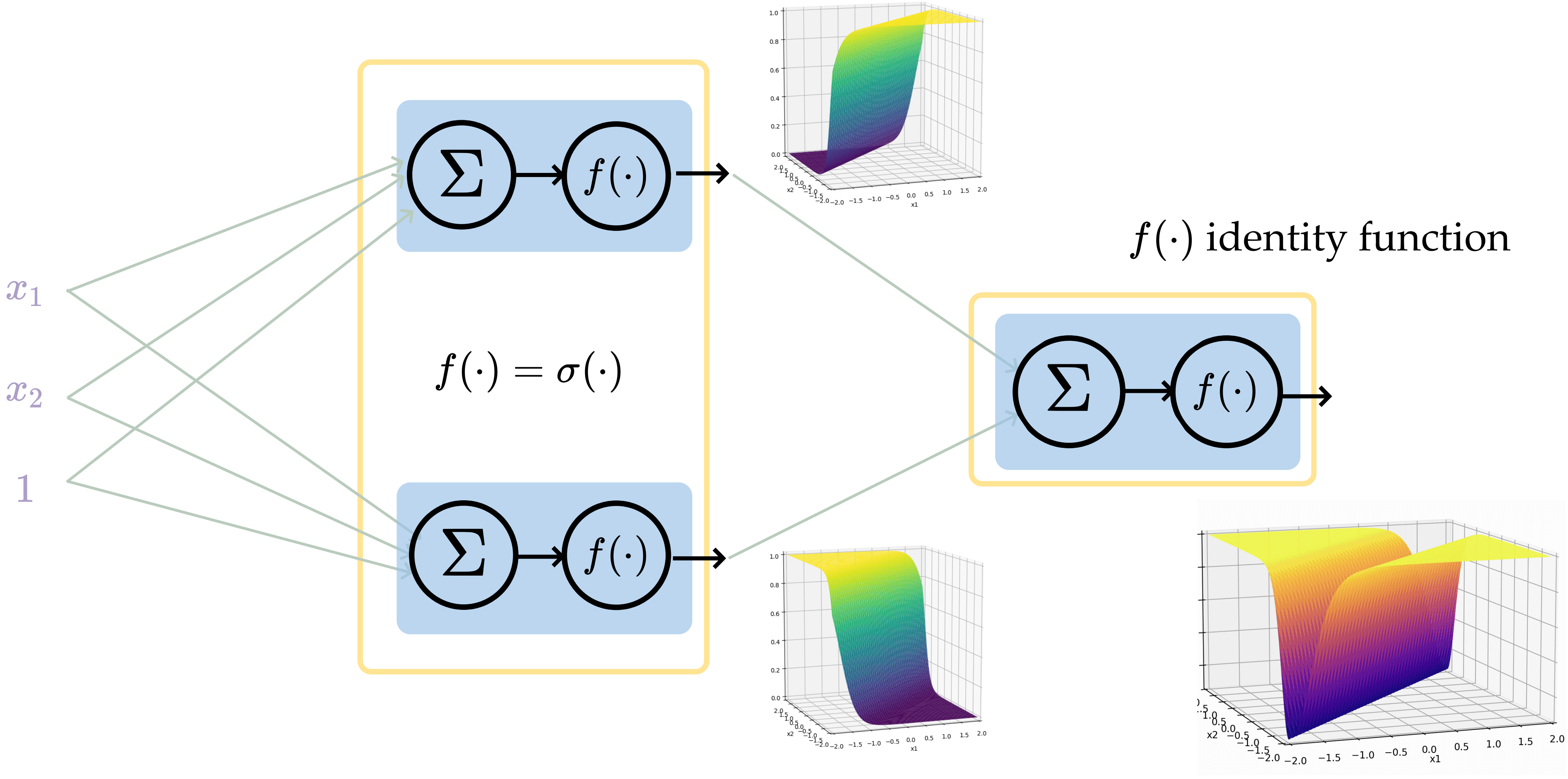
some appropriate weighted sum



$$\sigma_2 = \sigma(-5x_1 + 5x_2 + 1)$$



it can be represented as



Outline

- Systematic feature transformations
  - Engineered features
  - Polynomial features
  - Expressive power
- Neural networks
  - Terminologies
    - neuron, activation function, layer, feedforward network
  - Design choices

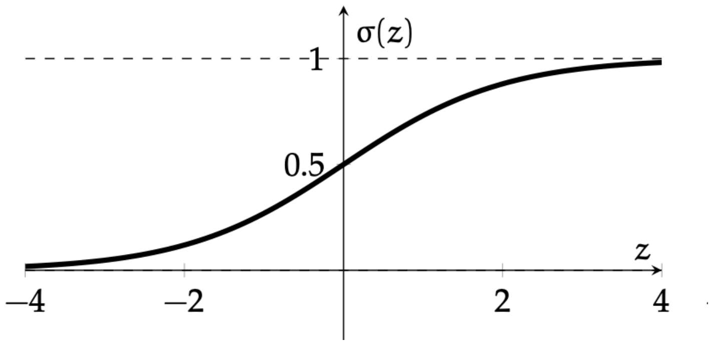
👋 heads-up:

all neural network diagrams focus on a single data point

Hidden layer activation function f choices

σ used to be the most popular

- firing rate of a neuron
- elegant gradient $\sigma'(z) = \sigma(z) \cdot (1 - \sigma(z))$

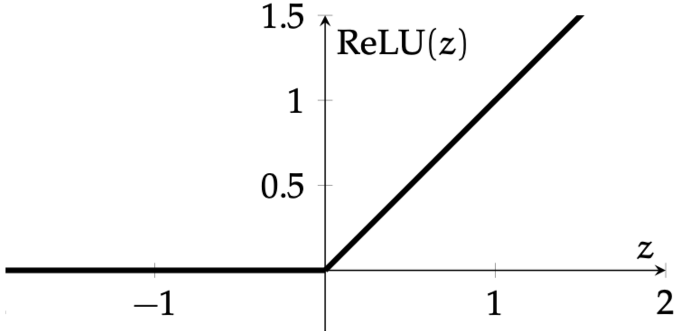


nowadays, default choice:

$$\mathrm{ReLU}(z) = \begin{cases} 0 & \text{if } z < 0 \\ z & \text{otherwise} \end{cases}$$

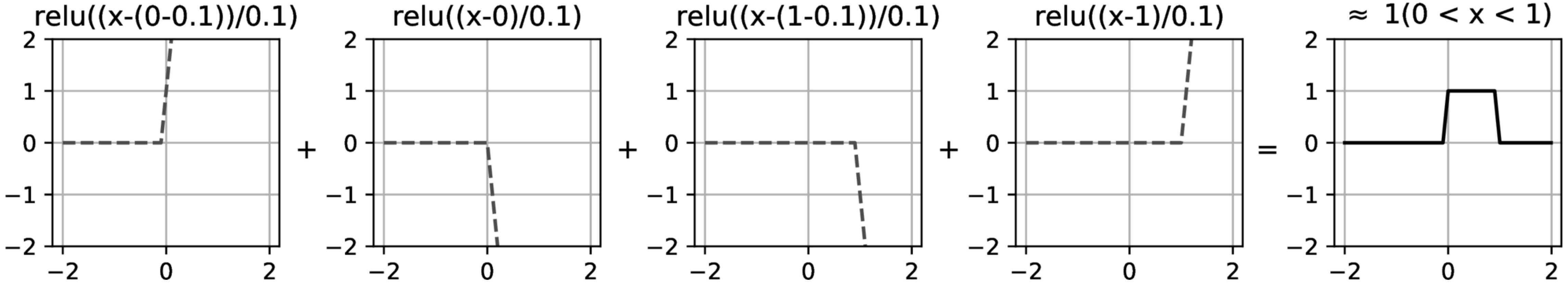
$$= \max(0, z)$$

$$= \max(0, w^T x)$$



very simple function form (so is the gradient).

compositions of ReLU(s) can be quite expressive



in fact, asymptotically, can approximate any function!

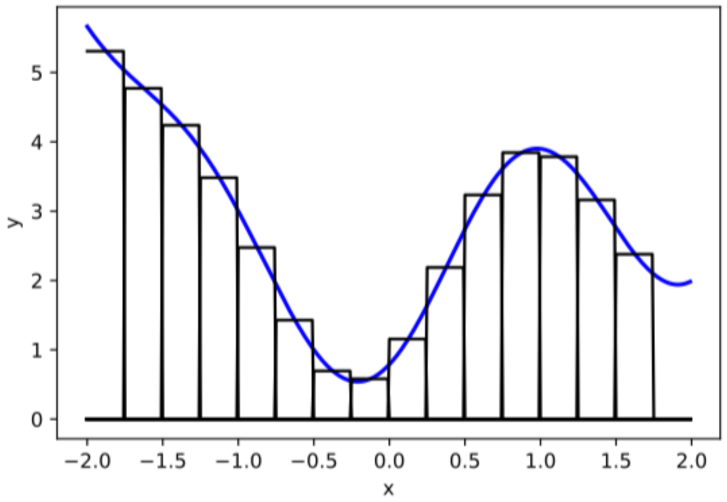
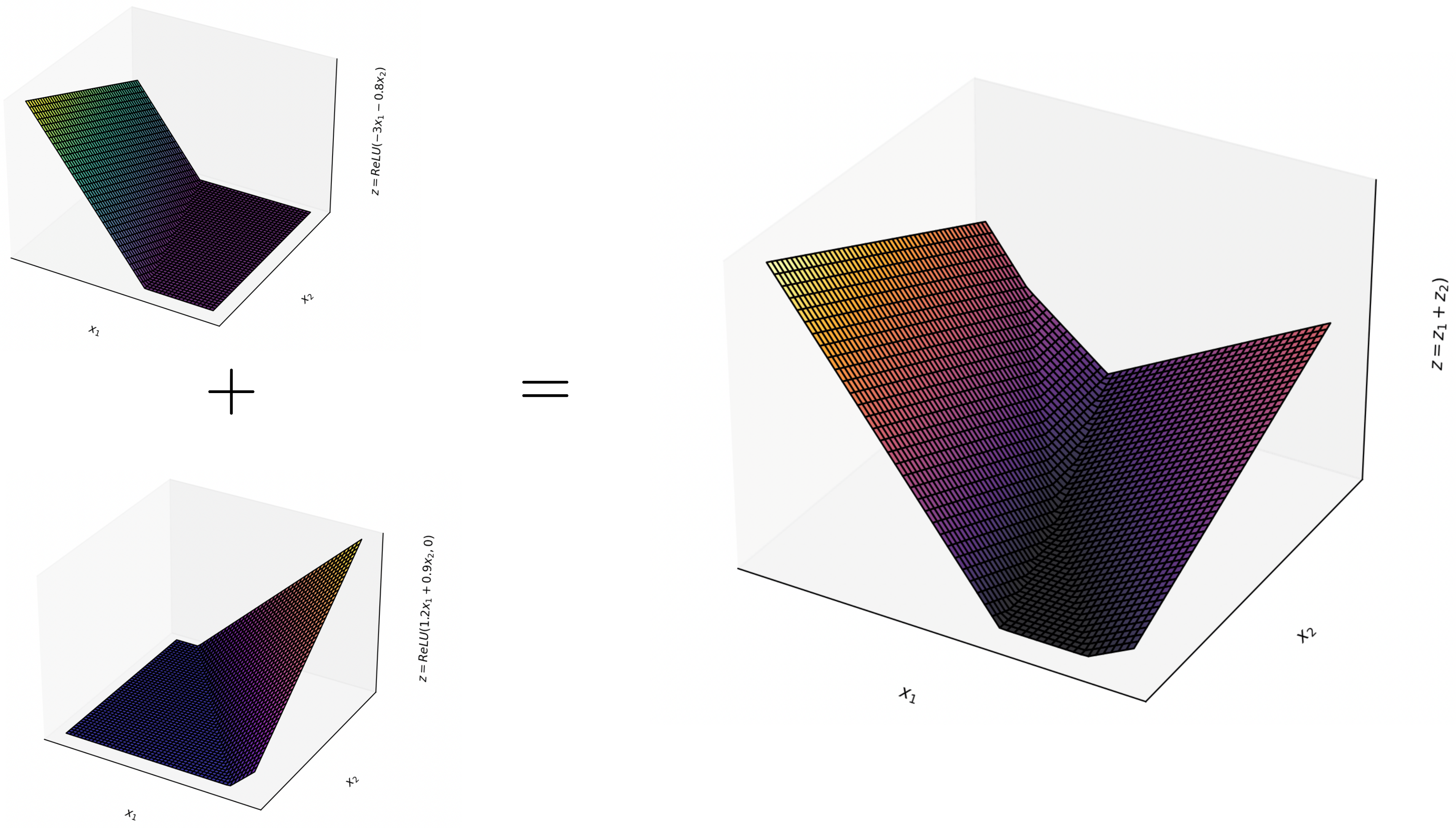


image credit: Phillip Isola

or give arbitrary decision boundaries!

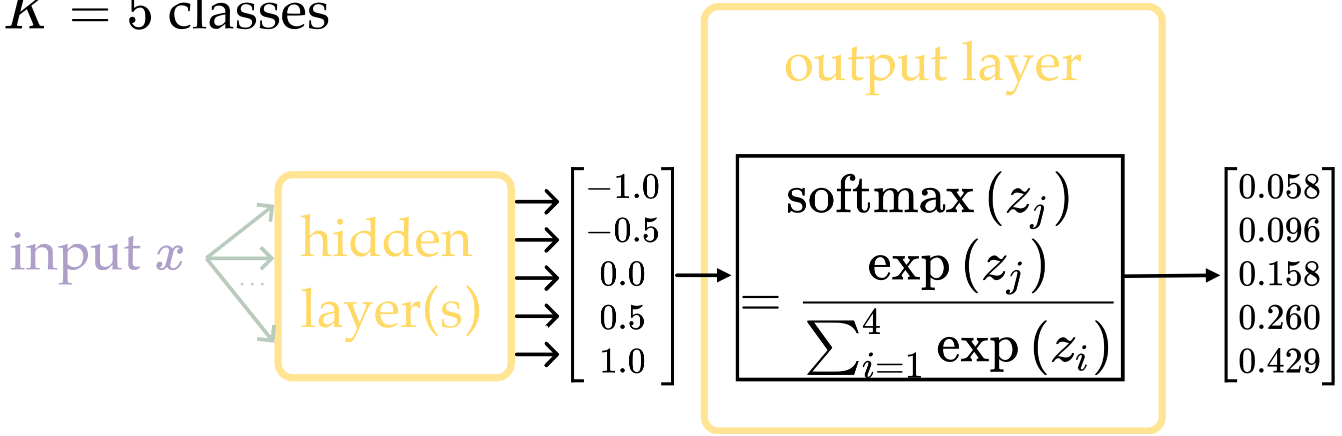


https://shenshen.mit.edu/demos/2layers.html output layer design choices

- # neurons, activation, and loss depend on the high-level goal.
- typically straightforward.
- Multi-class setup: if predict *one and only one* class out of K possibilities, then

  last layer: K neurons, softmax activation, cross-entropy loss

  e.g., say $K = 5$ classes



- other multi-class settings, see lab.

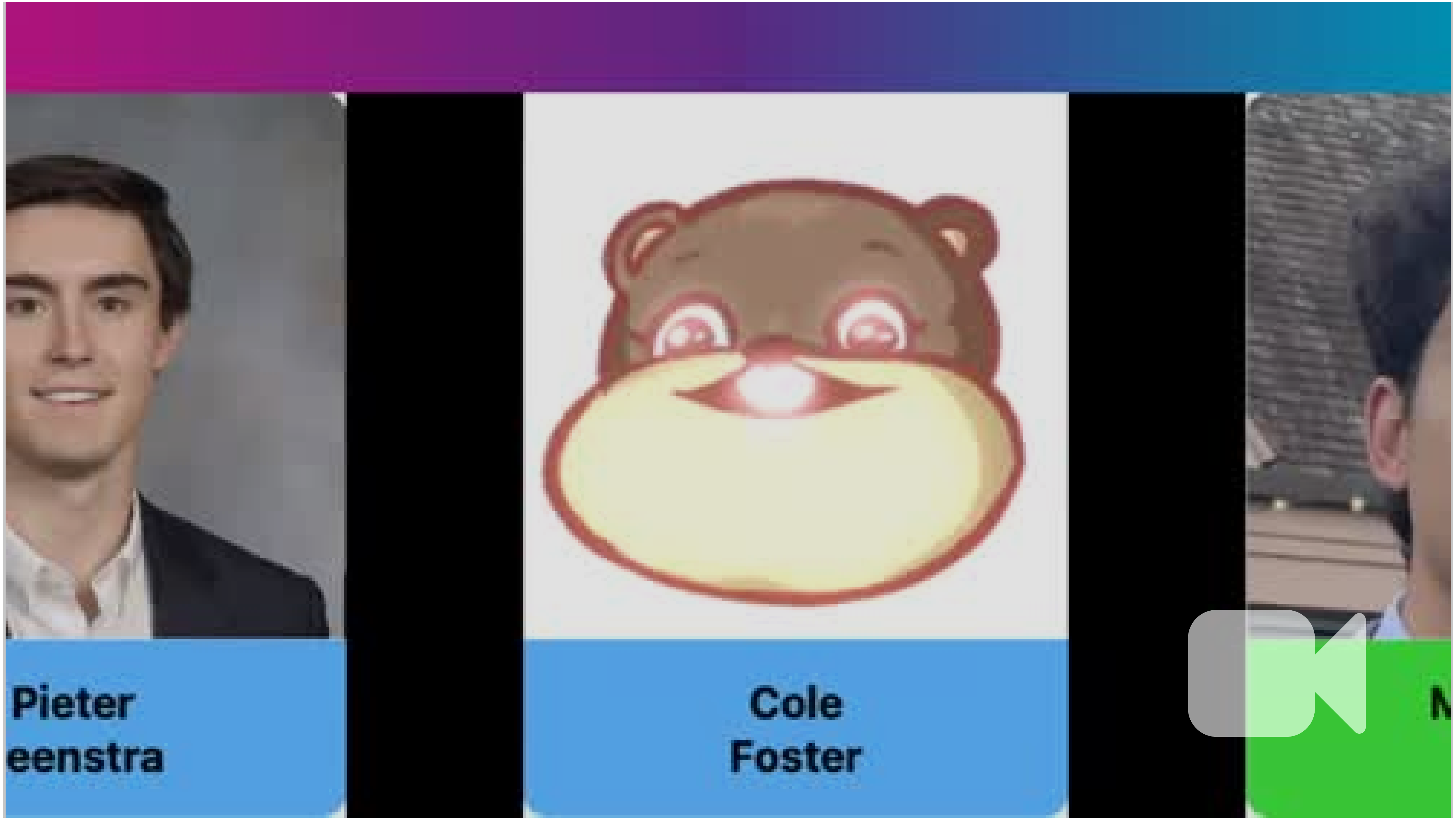
- Width: # of neurons in layers
- Depth: # of layers

- Typically, increasing either the width or depth (with non-linear activation) makes the model more expressive, but it also increases the risk of overfitting.

However, in the realm of neural networks, the precise nature of this relationship remains an active area of research—for example, phenomena like the double-descent curve and scaling laws

(The demo won't embed in PDF. But the direct link below works.)

https://playground.tensorflow.org/


Pieter
eenstra
Cole
Foster

Summary

- Linear models are mathematically and algorithmically convenient but not expressive enough -- by themselves -- for most jobs.
- We can express really rich hypothesis classes by performing a **fixed** non-linear feature transformation first, then applying our linear methods. But this can get tedious.
- Neural networks are a way to *automatically* find good transformations for us!
- Standard NNs have layers that alternate between parameterized linear transformations and fixed non-linear transforms (but many other designs are possible.)
- Typical non-linearities include sigmoid, tanh, relu, but mostly people use relu.
- Typical output transformations for classification are as we've seen: sigmoid, or softmax.

https://forms.gle/HnNqPizhMoNNSyMF7

We'd love to hear your thoughts.

Thanks!