人工智慧導論

Foundations of Artificial Intelligence

Linear Models for Supervised Machine Learning

監督式機器學習之線性模型

fai-ta@csie.ntu.edu.tw

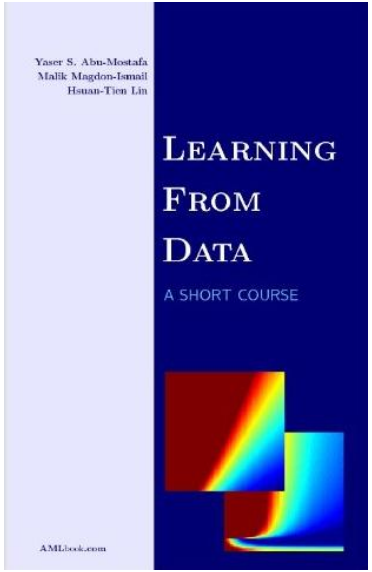
Lectured by Shang-Tse Chen & Yun-Nung (Vivian) Chen; slides credited from Hsuan-Tien Lin

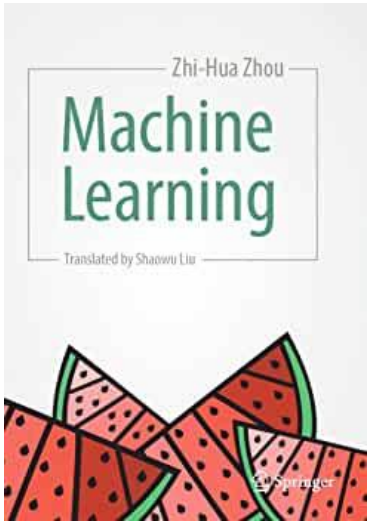
Tentative Schedule

Week	Topic	Assignment
2025/04/16	Linear Models for Supervised Machine Learning	
2025/04/23	Nonlinear Models for Supervised Machine Learning	A3 Release
2025/04/30	Deep Learning for Supervised Machine Learning (1)	
2025/05/07	Deep Learning for Supervised Machine Learning (2)	
2025/05/14	Generative AI & Reasoning Models	A4 Release
2025/05/21	Reinforcement Learning & Advanced Topics	
2025/05/28	Unsupervised Machine Learning (Virtual)	
2025/06/04	Final Project Due	

Textbooks

- Y. Abu-Mostafa, M. Magdon-Ismail, H.-T. Lin, **Learning from Data: A Short Course**
  - Not required, but good to read
- Z.-H. Zhou, **Machine Learning**
  - Coverage closer to this course





LIBSVM Demo

- https://www.csie.ntu.edu.tw/~cjlin/libsvm/

TITLE	CITED BY	YEAR
LIBSVM: A library for support vector machines CC Chang, CJ Lin ACM Transactions on Intelligent Systems and Technology (TIST) 2 (3), 27	55706	2011

What is Machine Learning?

甚麼是機器學習？

From Learning to Machine Learning

- Learning: acquiring *skill* with experience accumulated from *observations*

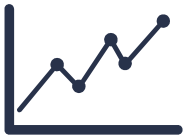
- Machine learning: acquitting *skill* with experience computed from *data*

  improving *performance measure* from *data*

skill ↔ improve performance score

Application #1

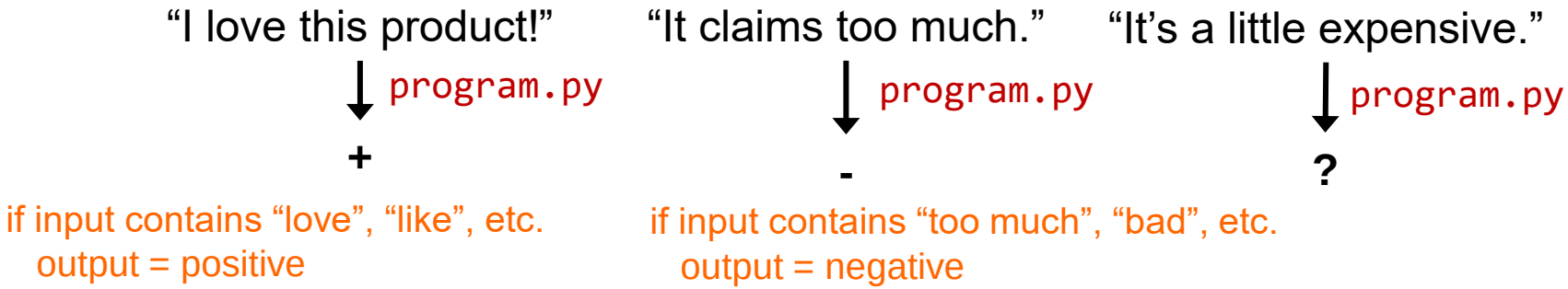
Computational Finance



stock data → **ML** → investment gain

Why Machine Learning?

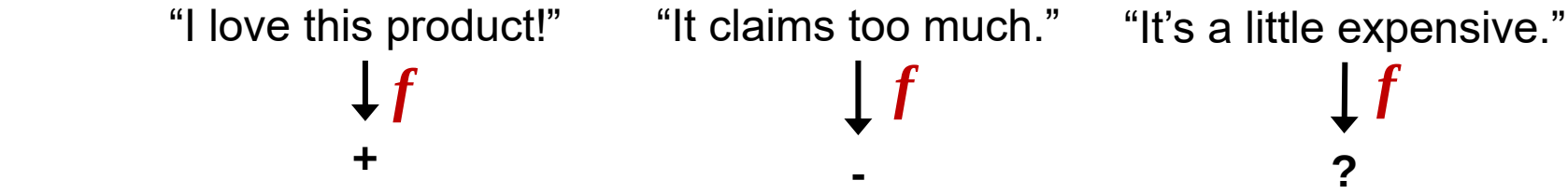
- Task: predicting positive or negative given a product review



Some tasks are complex, and we don't know how to write a program to solve them.

Learning ≈ Looking for a Function

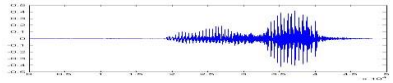
- Task: predicting positive or negative given a product review



Given a large amount of data, the machine learns what the function ***f*** should be.

Learning ≈ Looking for a Function

- Speech Recognition

$f($  $)=$ “你好”

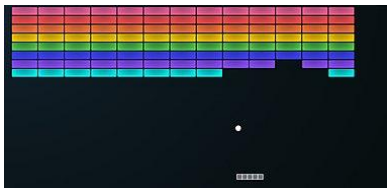
- Handwritten Recognition

$f($  $)=$ “2”

- Weather forecast

$f($  Thursday $)=$ “  Saturday”

- Play video games

$f($  $)=$ “move left”

ML: alternative/popular route to build a complicated system

Machine Learning vs. Artificial Intelligence

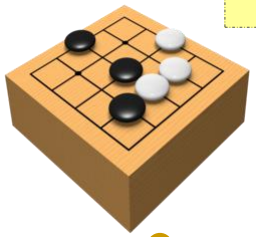
- **Artificial intelligence (AI)** is intelligence—perceiving, synthesizing, and inferring information—demonstrated by machines.

  → Compute something that shows intelligent behavior

- **Machine learning (ML)** is methods that leverage data to improve performance on some set of tasks.

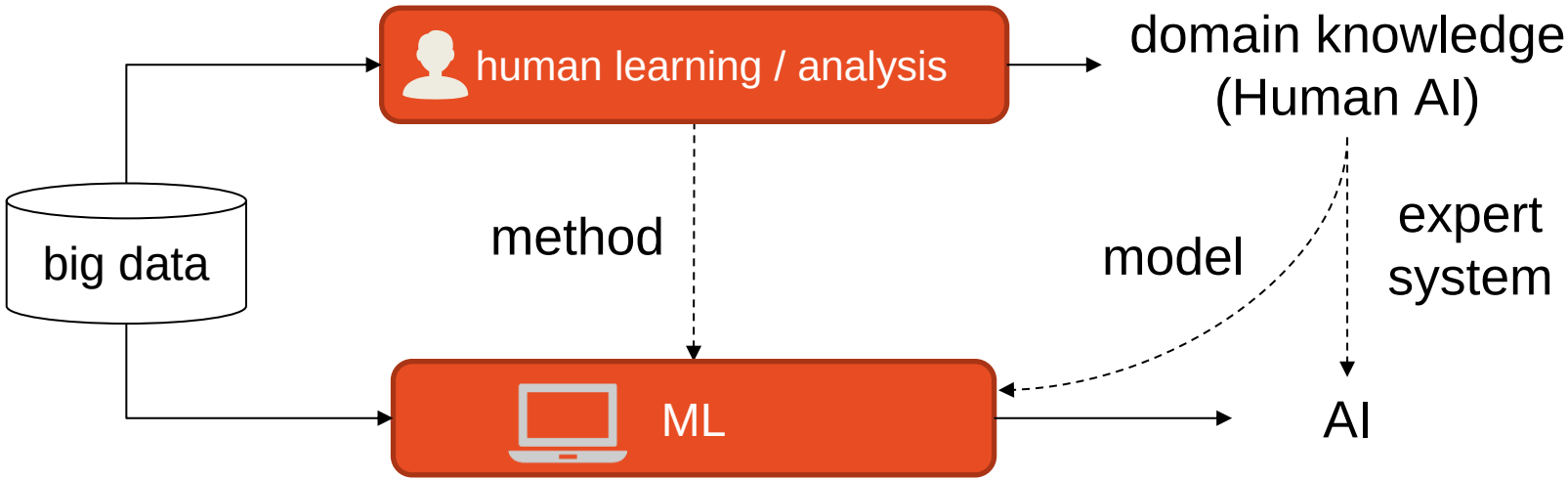
  → Use data to compute something that improves performance

ML is one possible route to realize AI



- Traditional AI: game tree
- ML for AI: learning from board data

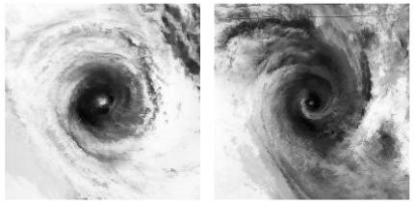
ML for Modern AI



Leveraging human learning is important especially in the beginning

Application #2

Tropical Cyclone Intensity Estimation



human learning / analysis

domain knowledge
(Human AI)

CNN

polar rotation
invariance

current
weather
system

ML

intensity
estimation

more accurate and can estimate in real-time

(Chen et al., Weather & Forecasting'19; Chen et al., AAAI'21)

Machine Learning Recipe

機器學習詳細步驟

Application #3

Job Application

- User Information

Attribute	Value
age	20 years
gender	female
OS	Windows
internship	2 times
GPA	3.92
project	NLP

What to learn (for improving performance)?
“Is this applicant suitable for company?”

Problem Formulation

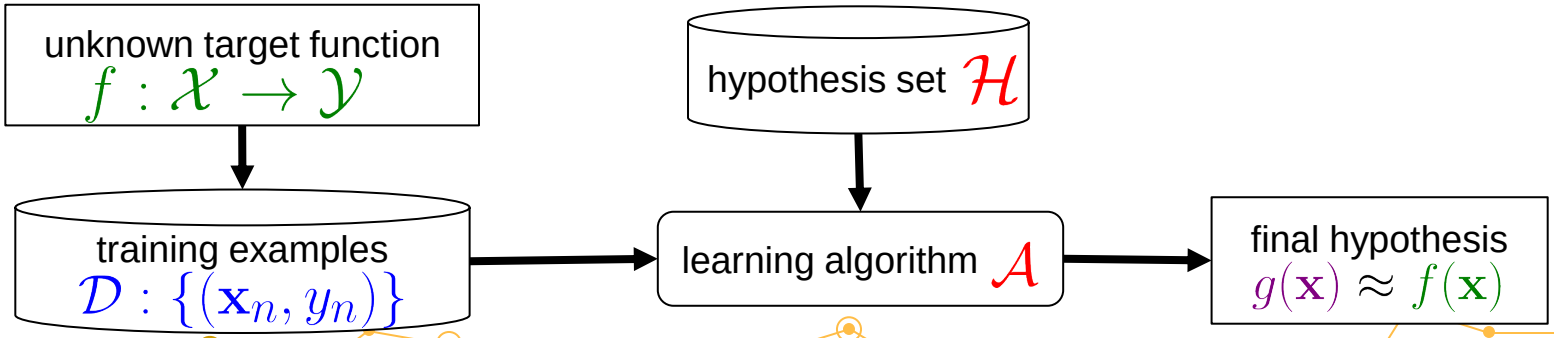
- Input: $\mathbf{x} \in \mathcal{X}$ (user application)
- Output: $y \in \mathcal{Y}$ (good/bad after approving this user)
- **Unknown** underlying pattern to be learned ⇔ target function:
  $f : \mathcal{X} \to \mathcal{Y}$ (ideal approval formula)
- Data ⇔ training examples: $\mathcal{D} = \{(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), \cdots, (\mathbf{x}_N, y_N)\}$ (history record)
- Hypothesis ⇔ skill with hopefully good performance:
  $g : \mathcal{X} \to \mathcal{Y}$ ("learned" formula), i.e. approve if
  - h_1: GPA > 4.0
  - h_2: internship > 1
  - h_3: project includes NLP

  – all candidate formula being considered: hypothesis set $\mathcal{H}$
  – procedure to learn best formula: algorithm $\mathcal{A}$

$\{(\mathbf{x}_n, y_n)\}$ from $f \longrightarrow \text{ML}(\mathcal{A}, \mathcal{H}) \longrightarrow g$

Machine Learning Recipe

0) Review key essence (is ML suitable?)
1) Collect/clean/process data $\mathcal{D}$
2) Decide hypothesis set $\mathcal{H}$
3) Choose error measure: how $g(\mathbf{x}) \approx f(\mathbf{x})$
4) Optimize error (and others) on $\mathcal{D}$ as $\mathcal{A}$
5) Pray for generalization: whether $g(\mathbf{x}) \approx f(\mathbf{x})$ for unseen $\mathbf{x}$



Step 0: Key Essence of ML

- Machine learning: use data to compute hypothesis

$$\{(\mathbf{x}_n, y_n)\} \longrightarrow \text{ML}(\mathcal{A}, \mathcal{H}) \longrightarrow g \approx f$$

improved performance measure

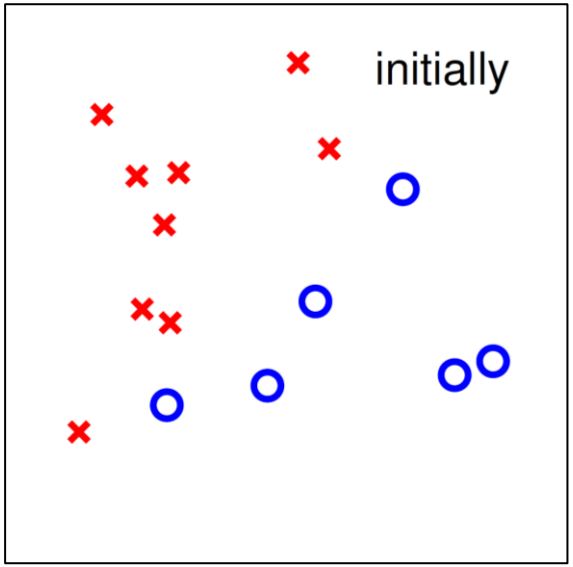
1. Exists some "underlying pattern" to be learned
   → *"Performance measure"* can be improved
2. No programmable (easy) definition
   → *"ML"* is needed
3. There is data about the pattern
   → ML has *"input"* to learn from

Key essence: decide whether to use ML

Step 1: Collect Data

- Features $\mathbf{x}$: points on the hyperplane (in $\mathbb{R}^d$)
- Labels y: ○ (+1) ✖ (-1)

  binary classification



Supervised learning: collect $\{(\mathbf{x}_n, y_n)\}$ pairs before learning

Application #4

Education

- KDDCup 2010 (NTU won world-champion)

quiz data → **ML** → whether a student can answer a given question correctly

9 million records
((student, quiz), correct?)
from 3000 students

binary classification

Step 2: Choose Hypothesis Set

- For user features $\mathbf{x} = (x_1, x_2, \cdots, x_d)$, compute a weighted "score" and

  approve if $\sum_{i=1}^{d} w_i x_i >$ threshold

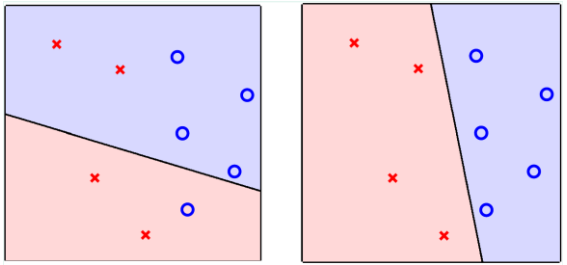
  deny if $\sum_{i=1}^{d} w_i x_i <$ threshold

- $\mathcal{Y}$: +1 (good), -1 (bad), 0 (ignored)

  linear formula $h \in \mathcal{H}$ are

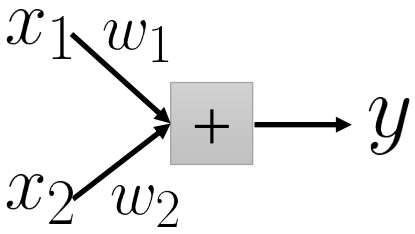
$$h(\mathbf{x}) = \text{sign}\left(\mathbf{w}^\top \mathbf{x} - \text{threshold}\right)$$

Attribute	Value
age	20 years
gender	female
OS	Windows
internship	2 times
GPA	3.92
project	NLP



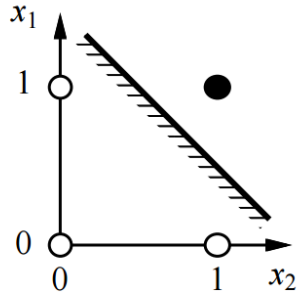
Linear (binary) classifier ("perceptron" historically)

Expression of Perceptron

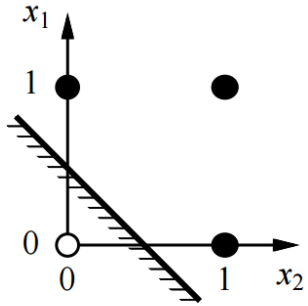


$$\begin{cases} \text{yes} & y \geq \theta \text{ (threshold)} \\ \text{no} & y < \theta \end{cases}$$

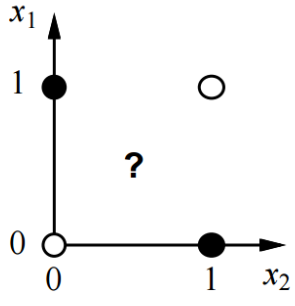
$$y = w_1 x_1 + w_2 x_2$$



(a) x_1 **and** x_2



(b) x_1 **or** x_2

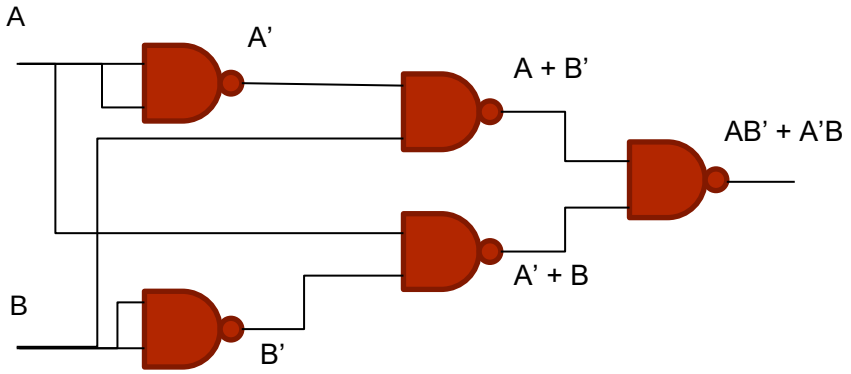


(c) x_1 **xor** x_2

A perceptron can represent AND, OR, NOT, etc., but not XOR → linear separator

How to Implement XOR?

Input		Output
A	B	
0	0	0
0	1	1
1	0	1
1	1	0



A xor B = AB' + A'B

Multiple operations can produce more complicate output

Step 3: Choose Error Measure

- $g \approx f$ is often evaluated by average $\mathrm{err}(g(\mathbf{x}), f(\mathbf{x}))$

pointwise error measure

- In-sample (within data)

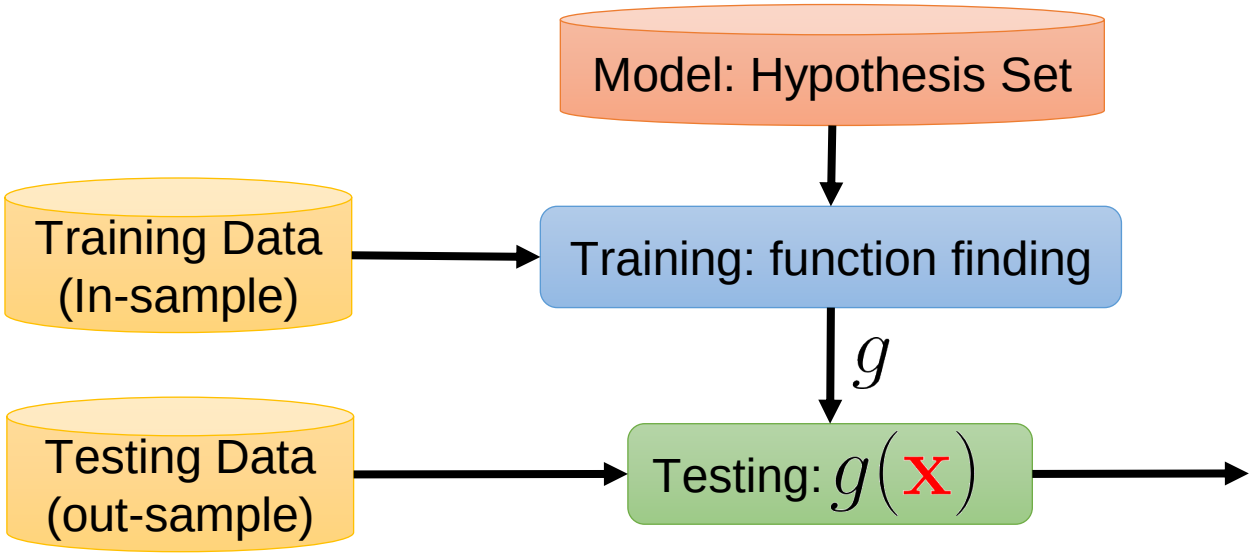
$$E_{\text{in}}(g) = \frac{1}{N}\sum_{n=1}^{N} \mathrm{err}(g(\mathbf{x}_n), \underbrace{f(\mathbf{x}_n)}_{=\,y_n})$$

- Out-of-sample (future data)

$$E_{\text{out}}(g) = \mathop{\mathbb{E}}_{\text{future } \mathbf{x}} \mathrm{err}(g(\mathbf{x}), f(\mathbf{x}))$$

Start from 0/1 error $\mathrm{err}(\tilde{y}, y) = [\tilde{y} \neq y]$ for **classification**

Machine Learning Framework



Training is to pick the best function given the observed data
Testing is to predict the label using the learned function

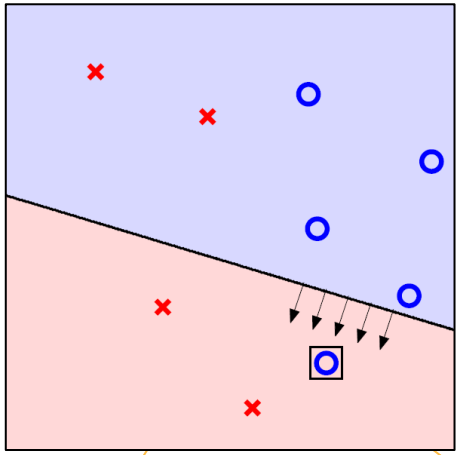
Execution of Machine Learning

機器學習過程

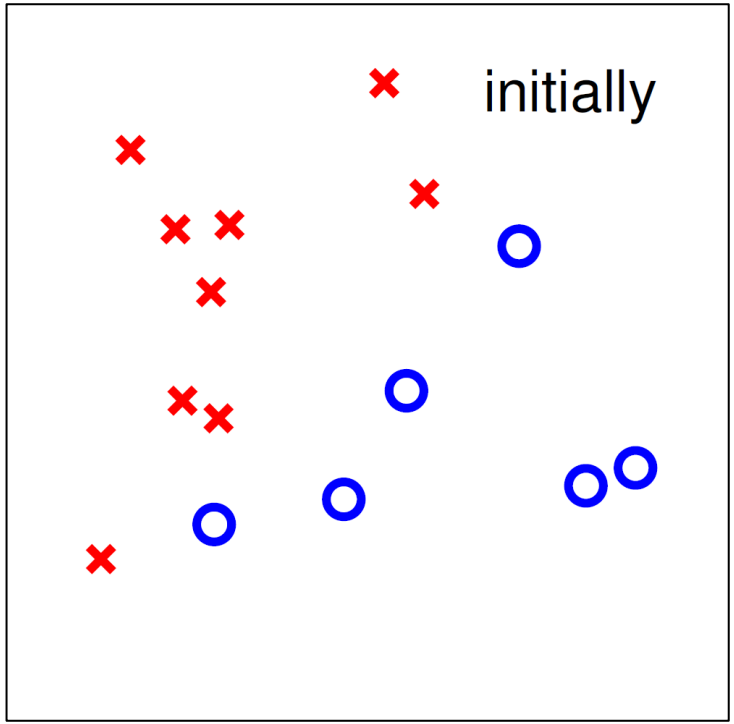
Step 4: Optimize Error on Data

- $\mathcal{H}$ = all possible perceptrons, $g = ?$

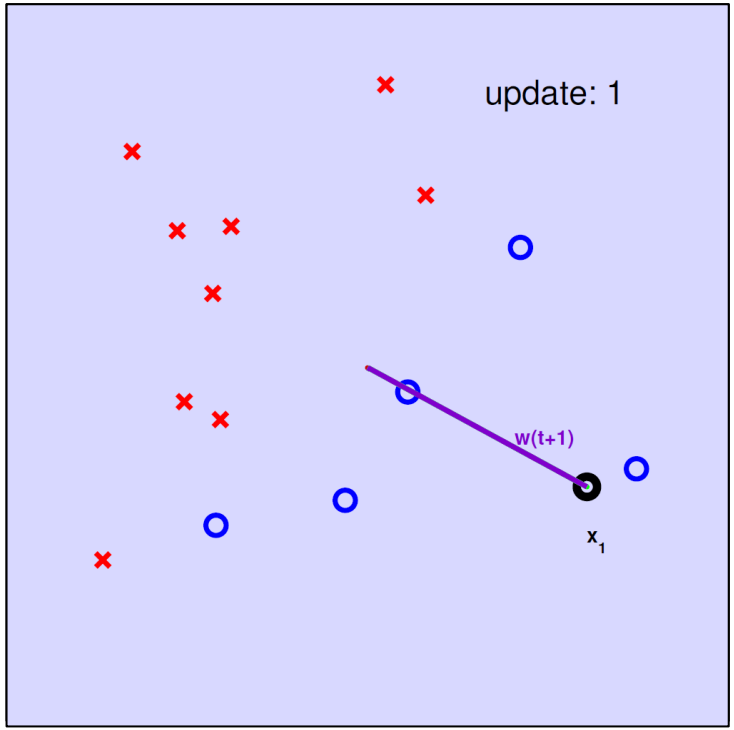
- Goal: $g \approx f$ (unknown)
  - Ideally $g(\mathbf{x}_n) = f(\mathbf{x}_n) = y_n$ ($g \approx f$ on $\mathcal{D}$)
  - Difficulty: $\mathcal{H}$ is of *infinite* size
- Idea: start from some g_0 , and "correct" its mistakes on $\mathcal{D}$



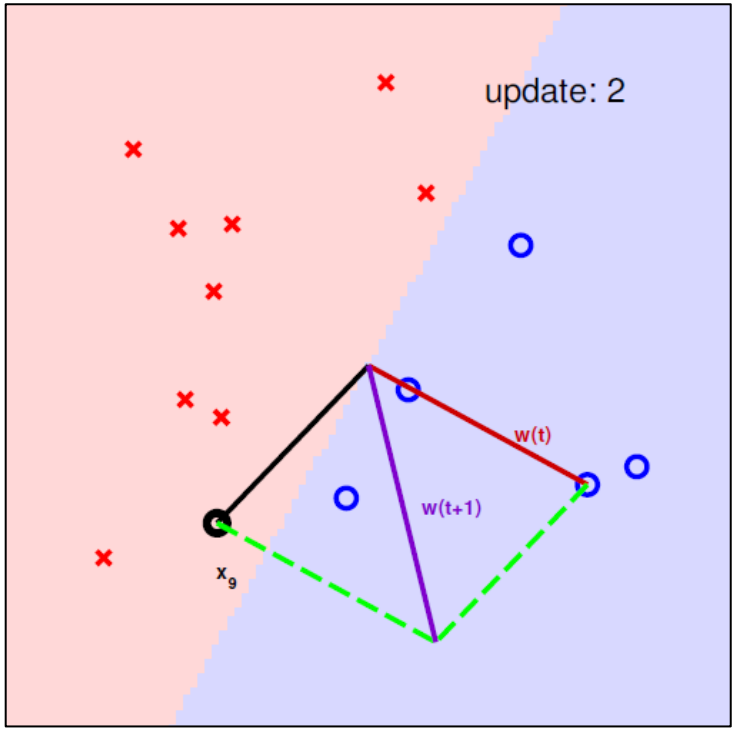
Procedure Visualization



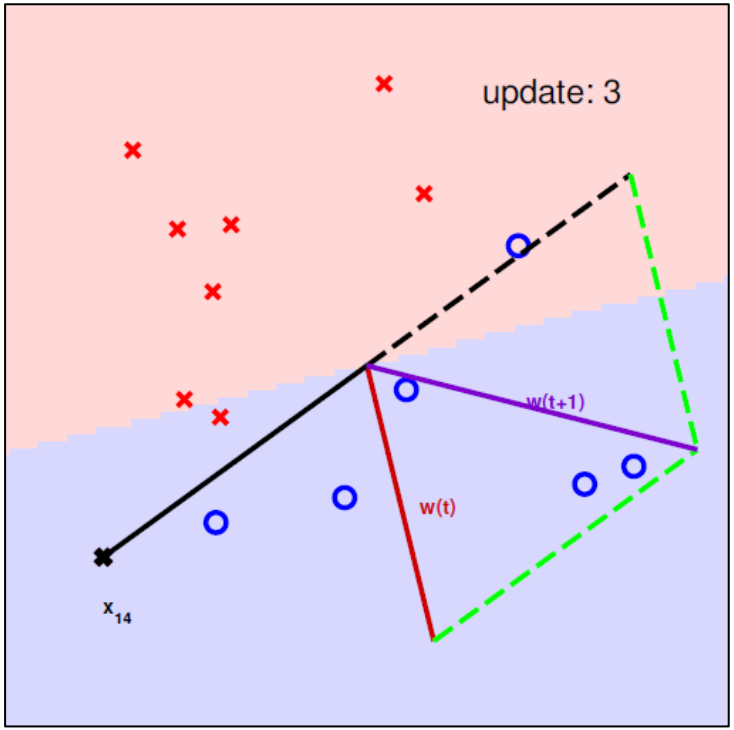
Procedure Visualization



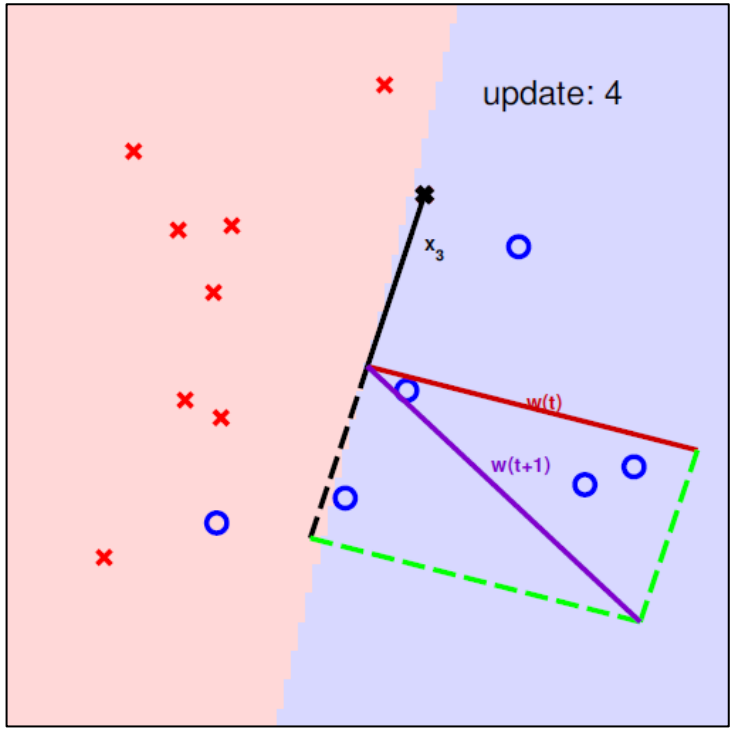
Procedure Visualization



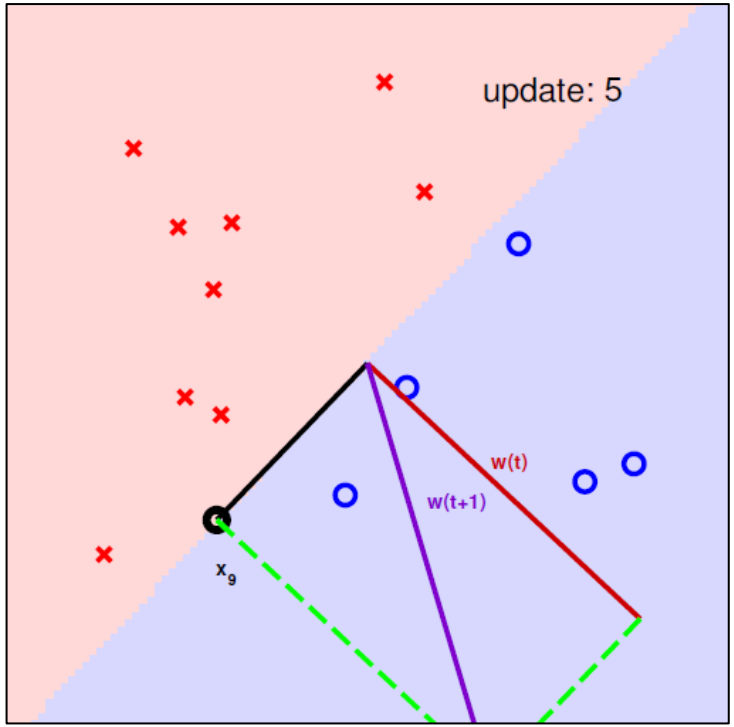
Procedure Visualization



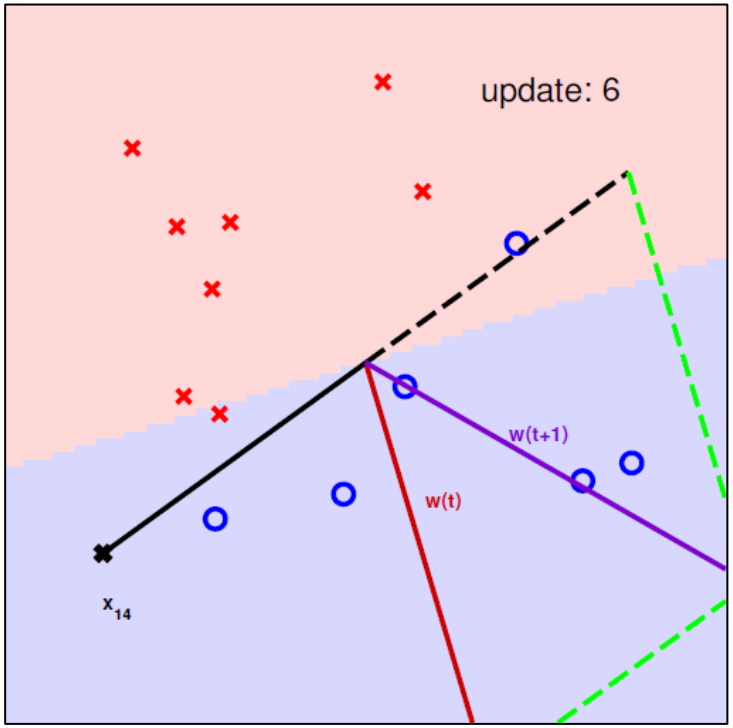
Procedure Visualization



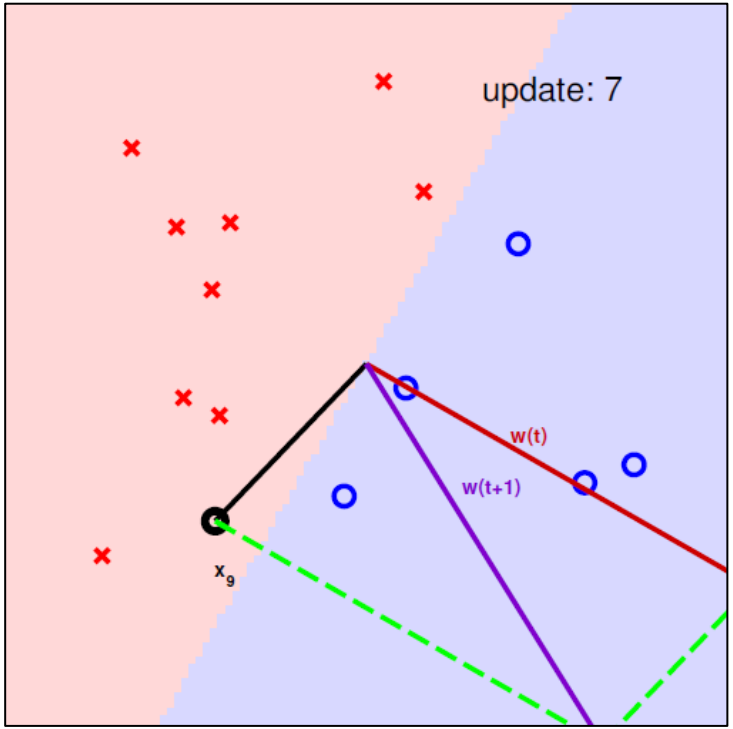
Procedure Visualization



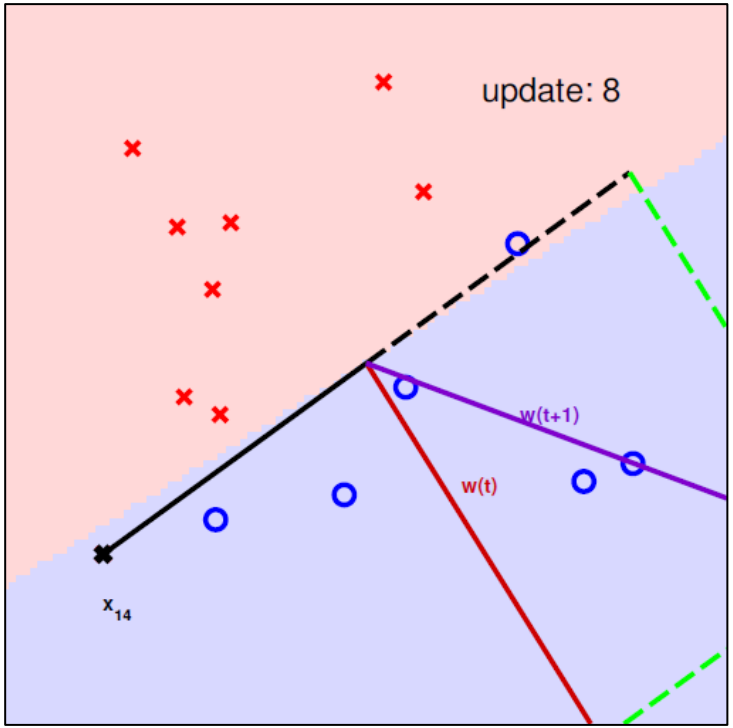
Procedure Visualization



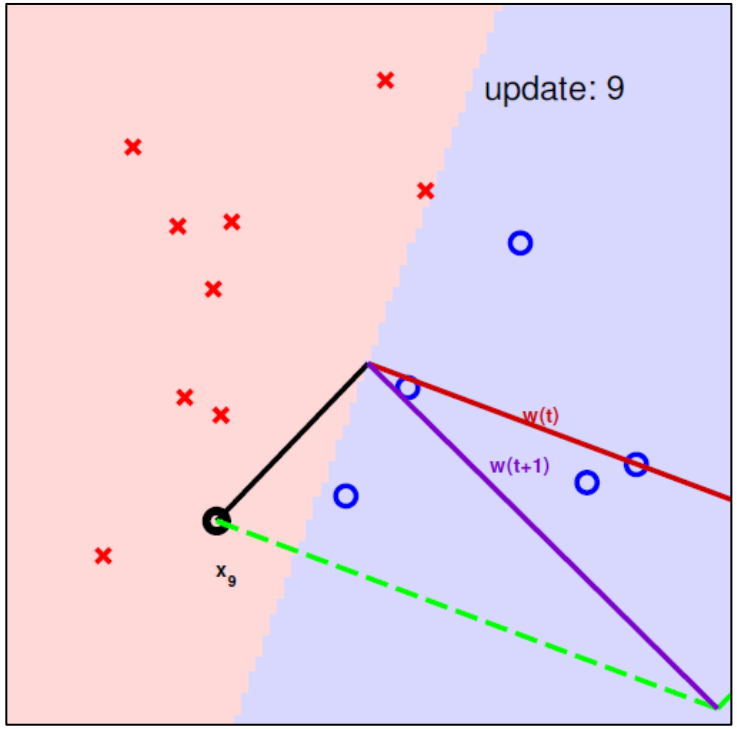
Procedure Visualization



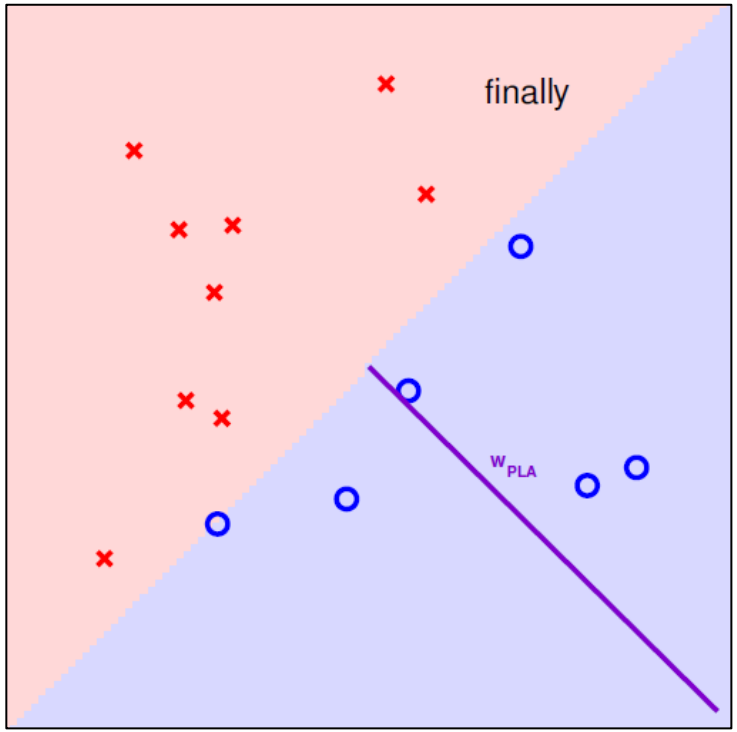
Procedure Visualization



Procedure Visualization

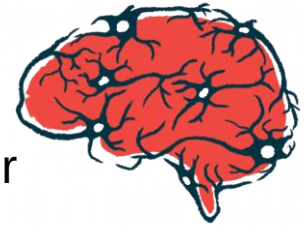


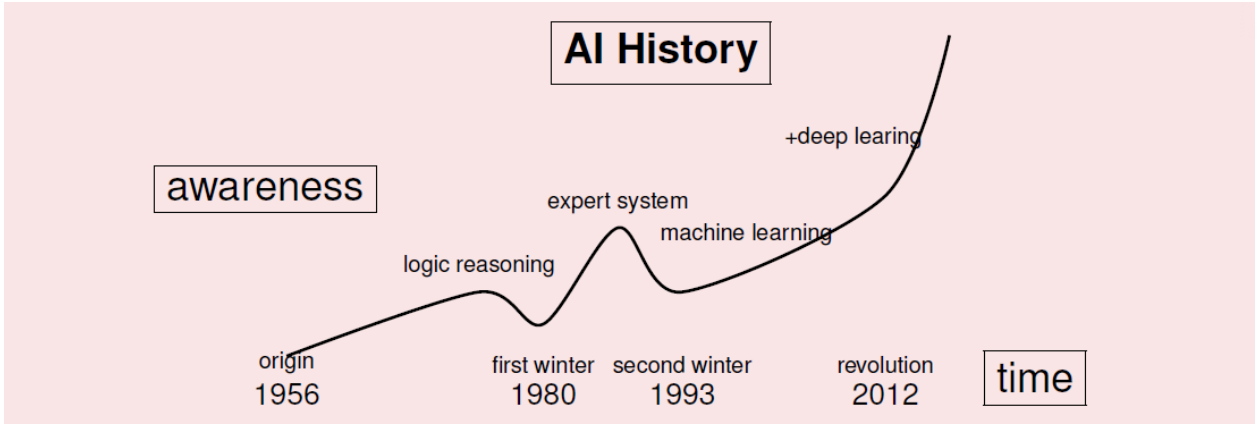
Procedure Visualization

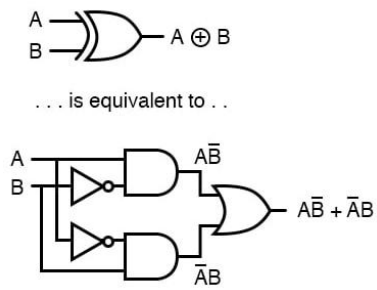


History: Perceptron Learning Algorithm

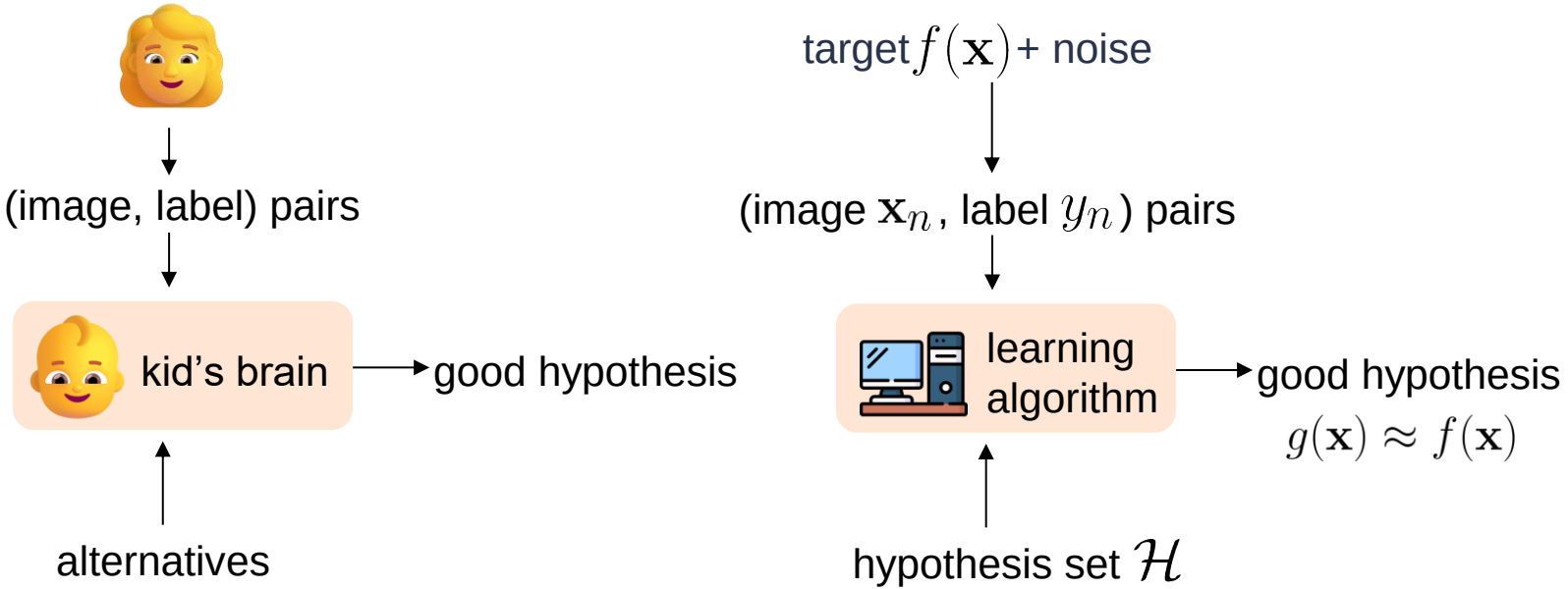
- Perceptron: designed as a soft/probabilistic logic reasoning model simulating a neuron
- Perceptron learning algorithm (PLA)
  - Limitations (Minsky & Papert, 1969) partially caused first winter







Step 5: Pray for Generalization



Challenge: see only $\{(\mathbf{x}_n, y_n)\}$ without knowing f nor noise
$\Rightarrow$ generalize to unseen $(\mathbf{x}, y)$ w.r.t. $f(\mathbf{x})$

Generalization

- memorize ≠ generalize
- perfect from kid's view ≠ good for parent
- perfect during training ≠ good when testing

If $\mathcal{H}$ is **simple** (like lines), generalization is usually possible

Linear Regression

線性迴歸

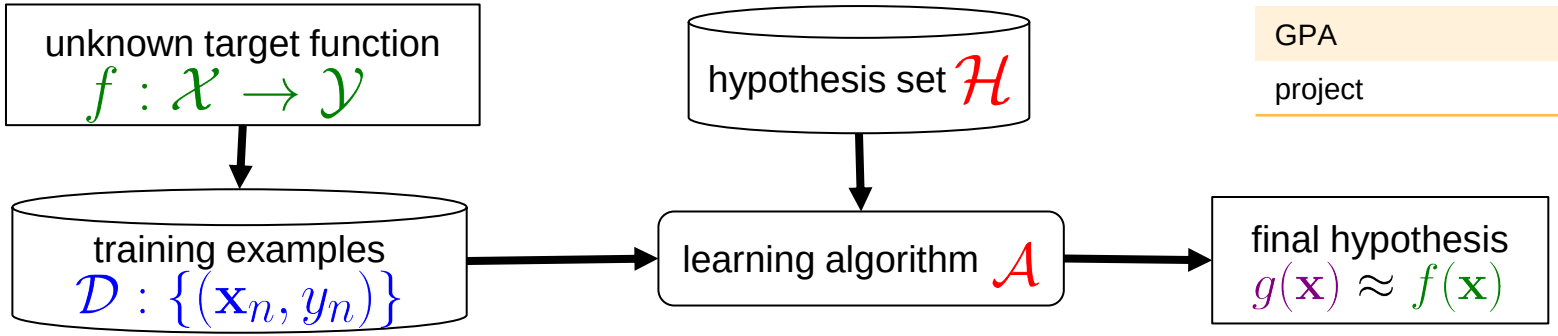
Problem Formulation of Linear Regression

- Output is a **value** instead of yes/no

Application #5

approval order?

Attribute	Value
age	20 years
gender	female
OS	Windows
internship	2 times
GPA	3.92
project	NLP



$\mathcal{Y} = \mathbb{R}$: **regression**

Application #6

Recommendation System

- Netflix prize in 2006: 10% improvement = 1 million dollar prize

rated data → **ML** → predict how a user rates a movie

100M ratings that 480K users gave to 17.7K movies

1-5 (stars)
regression

- Yahoo! Competition (movie → songs) in KDDCup 2011 (NTU won world-champion again)

Linear Regression Hypothesis

- For user features $\mathbf{x} = (x_0, x_1, x_2, \cdots, x_d)$, approximate the desired value with a weighted sum:

$$y \approx \sum_{i=0}^{d} w_i x_i$$

Attribute	Value
age	20 years
gender	female
browser	Microsoft Edge
account age	2 years
payment history	1800

- $\mathcal{Y} = \mathbb{R}$ linear regression hypothesis:

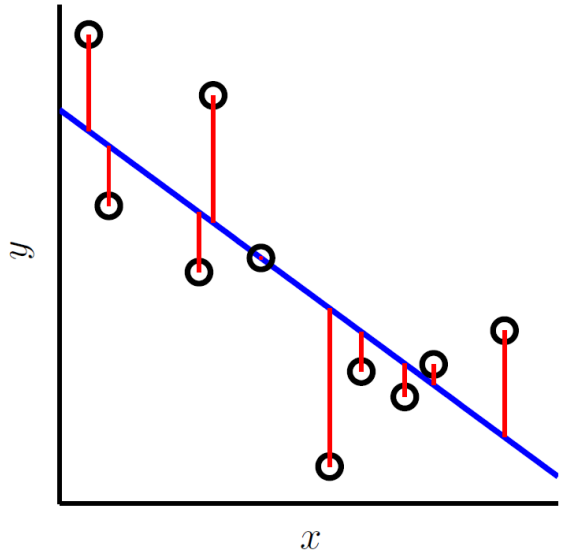
$$h(\mathbf{x}) = \mathbf{w}^\top \mathbf{x}$$

$$h(\mathbf{x}) = \text{sign}\left(\mathbf{w}^\top \mathbf{x} - \text{threshold}\right)$$

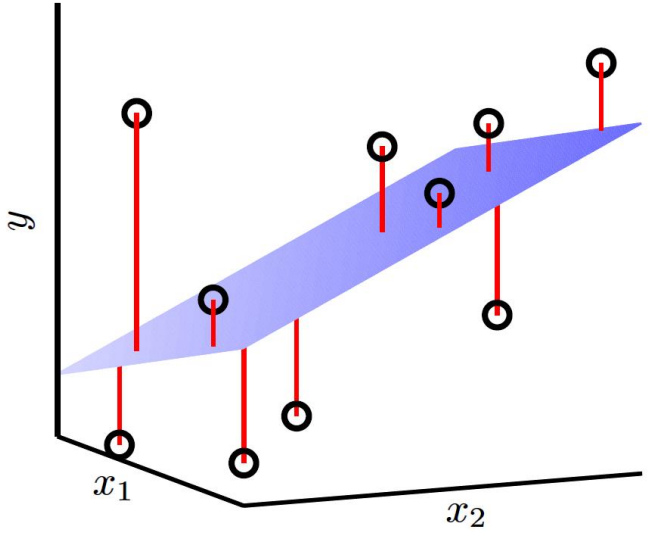
$h(\mathbf{x})$: like perceptron, but without the sign

Illustration of Linear Regression

$\mathbf{x} = (x) \in \mathbb{R}$



$\mathbf{x} = (x_1, x_2) \in \mathbb{R}^2$



Linear regression: find lines/hyperplanes with small residuals

Choose Error Measure

- $g \approx f$ is often evaluated by average $\mathrm{err}(g(\mathbf{x}), f(\mathbf{x}))$
- Popular error measure is squared error: $\mathrm{err}(\hat{y}, y) = (\hat{y} - y)^2$

- In-sample (within data)

$$E_{\text{in}}(\mathbf{w}) = \frac{1}{N} \sum_{n=1}^{N} (\underbrace{h(\mathbf{x}_n)}_{= \mathbf{w}^\top \mathbf{x}_n} - y_n)^2$$

- Out-of-sample (future data)

$$E_{\text{out}}(\mathbf{w}) = \mathop{\mathbb{E}}_{(\mathbf{x},y) \sim P} (\mathbf{w}^\top \mathbf{x} - y)^2$$

Next: how to minimize $E_{\text{in}}(\mathbf{w})$?

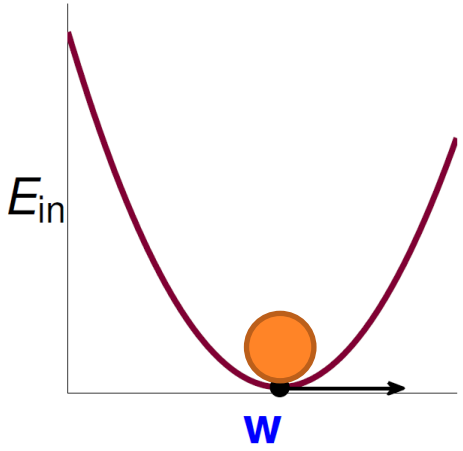
Optimize Error on Data

- Goal: minimize the error on data $\mathcal{D}$

$$\min_{\mathbf{w}} E_{\text{in}}(\mathbf{w}) = \frac{1}{N}\sum_{n=1}^{N}(\mathbf{w}^\top \mathbf{x}_n - y_n)^2$$

- $E_{\text{in}}(\mathbf{w})$: continuous, differentiable, **convex**
- Necessary condition of "best" $\mathbf{w}$

$$\nabla E_{\text{in}}(\mathbf{w}) = \begin{bmatrix} \frac{\partial E_{\text{in}}}{\partial w_0}(\mathbf{w}) \\ \frac{\partial E_{\text{in}}}{\partial w_1}(\mathbf{w}) \\ \vdots \\ \frac{\partial E_{\text{in}}}{\partial w_d}(\mathbf{w}) \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ \vdots \\ 0 \end{bmatrix}$$



Task: find $\mathbf{w}^*$ s.t. $\nabla E_{\text{in}}(\mathbf{w}^*) = 0$.

Linear Regression Algorithm

1) Construct an input matrix $\mathbf{X}$ and an output vector $\mathbf{y}$ from $\mathcal{D}$

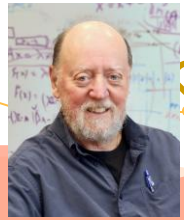
$$\mathbf{X} = \underset{N \times (d+1)}{\begin{bmatrix} -- \mathbf{x}_1^\top -- \\ -- \mathbf{x}_2^\top -- \\ \vdots \\ -- \mathbf{x}_N^\top -- \end{bmatrix}} \qquad \mathbf{y} = \underset{N \times 1}{\begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_N \end{bmatrix}}$$

2) Calculate pseudo-inverse $\underset{(d+1) \times N}{\mathbf{X}^\dagger} = (\mathbf{X}^\top \mathbf{X})^{-1} \mathbf{X}^\top$ (if $\mathbf{X}^\top \mathbf{X}$ invertible)

3) Return $\underset{(d+1) \times 1}{\mathbf{w}^*} = \mathbf{X}^\dagger \mathbf{y}$

Simple and efficient with good library

2021 Turing Award
Jack Dongarra



Statistics vs. Machine Learning

- Linear regression roots from statistics
- **Statistics**: use data to make inference about an unknown process
- **Machine learning**: use data to compute hypothesis $g \approx f$ (target)

- g is an inference outcome; f is something unknown
- Traditional statistics also focus on *provable results with math assumptions* (less about computation) $\Rightarrow$ many useful tools for ML

Logistic Regression

邏輯迴歸

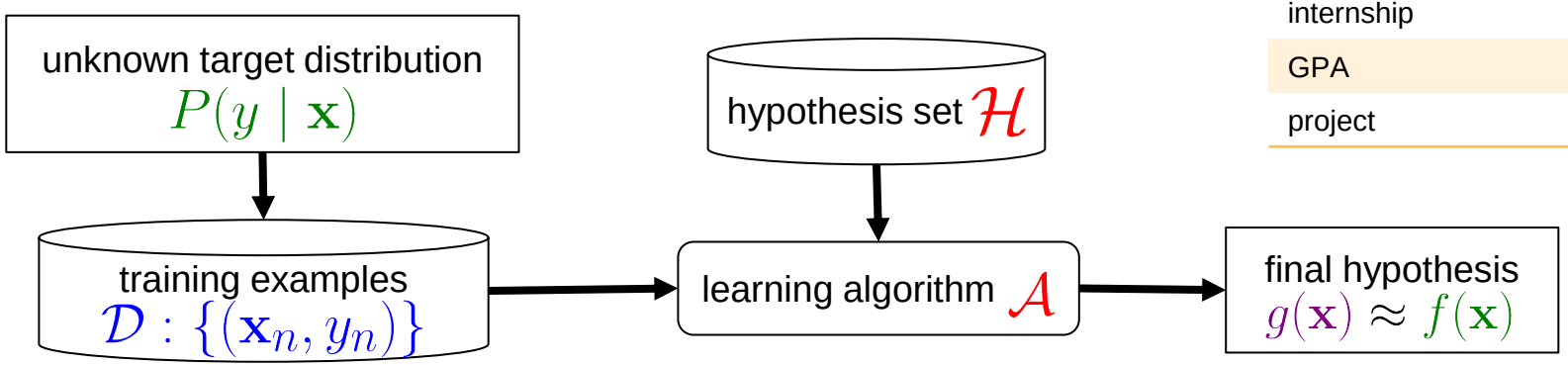
Problem Formulation of Logistic Regression

- Output is a **probability**

Application #7

Probability of accepting offer?

Attribute	Value
age	20 years
gender	female
OS	Windows
internship	2 times
GPA	3.92
project	NLP

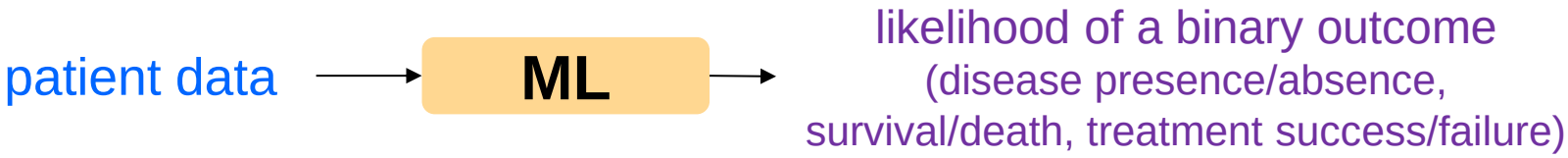


"soft" binary classification:

$$f(\mathbf{x}) = P(+1 \mid \mathbf{x}) \in [0, 1]$$

Application #8

Medical Research



- Determining the risk factors for a certain disease by examining various risk factors
- Investigating the association between a treatment or medication and the risk of developing a certain outcome (e.g., side effects).
- Identifying the predictors of mortality in critically ill patients in the ICU, such as vital signs.

Soft Binary Classification

- Target function $f(\mathbf{x}) = P(+1 \mid \mathbf{x}) \in [0, 1]$

- Ideal "regression" data

$$(\mathbf{x}_1, y'_1 = 0.9 = P(+1 \mid \mathbf{x}_1))$$
$$(\mathbf{x}_2, y'_2 = 0.2 = P(+1 \mid \mathbf{x}_2))$$
$$\vdots$$
$$(\mathbf{x}_N, y'_N = 0.6 = P(+1 \mid \mathbf{x}_N))$$

- Actual data

$$(\mathbf{x}_1, y_1 = \circ \sim P(y \mid \mathbf{x}_1))$$
$$(\mathbf{x}_2, y_2 = \times \sim P(y \mid \mathbf{x}_2))$$
$$\vdots$$
$$(\mathbf{x}_N, y_N = \times \sim P(y \mid \mathbf{x}_N))$$

soft binary classification ≈
[0,1]-regression with classification data

Soft Binary Classification

- Target function $f(\mathbf{x}) = P(+1 \mid \mathbf{x}) \in [0, 1]$

- Ideal "regression" data

$$(\mathbf{x}_1, y'_1 = 0.9 = P(+1 \mid \mathbf{x}_1))$$
$$(\mathbf{x}_2, y'_2 = 0.2 = P(+1 \mid \mathbf{x}_2))$$
$$\vdots$$
$$(\mathbf{x}_N, y'_N = 0.6 = P(+1 \mid \mathbf{x}_N))$$

- Actual data

$$(\mathbf{x}_1, y'_1 = 1 = [\circ \sim P(y \mid \mathbf{x}_1)])$$
$$(\mathbf{x}_2, y'_2 = 0 = [\circ \sim P(y \mid \mathbf{x}_2)])$$
$$\vdots$$
$$(\mathbf{x}_N, y'_N = 0 = [\circ \sim P(y \mid \mathbf{x}_N)])$$

soft binary classification ≈
[0,1]-regression with classification data

Logistic Hypothesis

Attribute	Value
age	20 years
gender	female
OS	Windows
internship	2 times
GPA	3.92
project	NLP

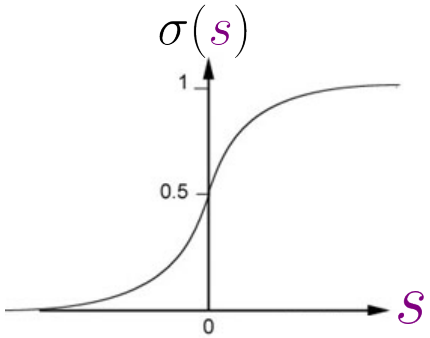
- For user features $\mathbf{x} = (x_0, x_1, x_2, \cdots, x_d)$, calculate a weighted "potential score":

$$s \approx \sum_{i=0}^{d} w_i x_i$$

convert the score to **estimated probability** by logistic function $\sigma(s)$

- Logistic hypothesis:

$$h(\mathbf{x}) = \sigma(\mathbf{w}^\top \mathbf{x}) = \frac{1}{1 + \exp(-\mathbf{w}^\top \mathbf{x})}$$



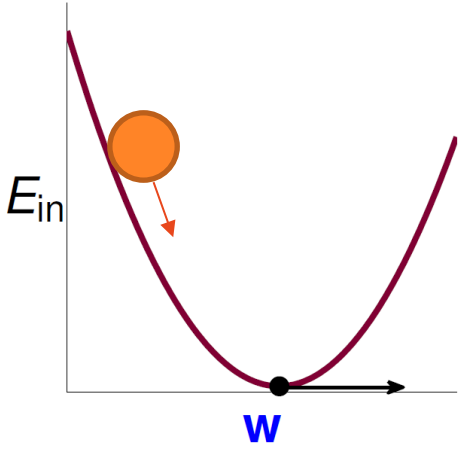
Choose Error and Optimize on Data

- Popular error measure $-\ln \sigma(y_n \mathbf{w}^\top \mathbf{x}_n)$ is cross-entropy derived from maximum likelihood
- Goal: minimize the error on data $\mathcal{D}$

$$\min_{\mathbf{w}} E_{\text{in}}(\mathbf{w}) = \frac{1}{N} \sum_{n=1}^{N} -\ln \sigma(y_n \mathbf{w}^\top \mathbf{x}_n)$$

- $E_{\text{in}}(\mathbf{w})$: continuous, differentiable, twice-differentiable, **convex**
- Locate **valley** $\nabla E_{\text{in}}(\mathbf{w}) = 0$

Intuition: larger $y_n \mathbf{w}^\top \mathbf{x}_n$



Basic algorithm: **gradient descent**

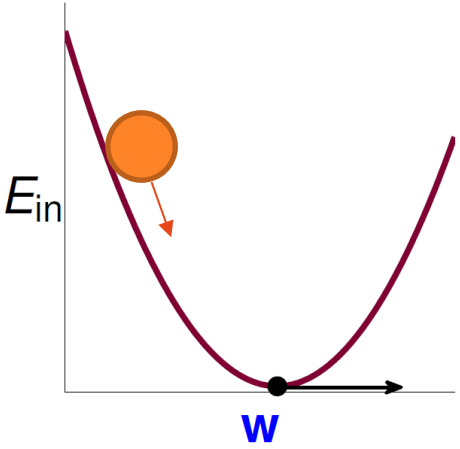
Gradient Descent

- For t = 0, 1, …

$$\mathbf{w}_{t+1} \leftarrow \mathbf{w}_t + \eta \mathbf{v}$$

  when stop, return last $\mathbf{w}$ as g

- PLA: $\mathbf{v}$ comes from mistake correction
- Smooth $E_{\text{in}}(\mathbf{w}_t)$ for logistic regression: rolling a ball
  - Direction $\mathbf{v}$: (assumed) of unit length
  - Step size η: (assumed) positive



gradient descent: $\mathbf{v} \propto \nabla E_{\text{in}}(\mathbf{w}_t)$.

Logistic Regression Algorithm

1) Initialize $\mathbf{w}_0$
2) For t = 0, 1, …
   - Compute $\nabla E_{\text{in}}(\mathbf{w}_t) = \frac{1}{N}\sum_{n=1}^{N} \sigma(-y_n \mathbf{w}^\top \mathbf{x}_n)(-y_n \mathbf{x}_n)$
   - Update by $\mathbf{w}_{t+1} \leftarrow \mathbf{w}_t + \eta \nabla E_{\text{in}}(\mathbf{w}_t)$
3) Until $\nabla E_{\text{in}}(\mathbf{w}_{t+1}) \approx 0$ or enough iterations
4) Return last $\mathbf{w}$ as g

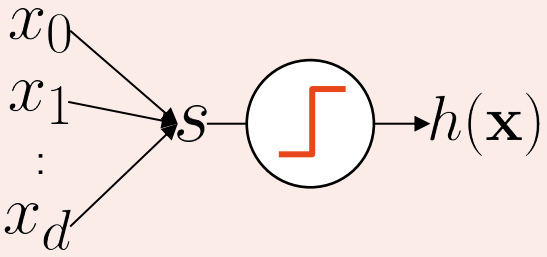
Speed up with more sophisticated tools

Linear Models

- Linear scoring function: $s = \mathbf{w}^\top \mathbf{x}$

Linear Classification

$$h(\mathbf{x}) = \text{sign}(s)$$

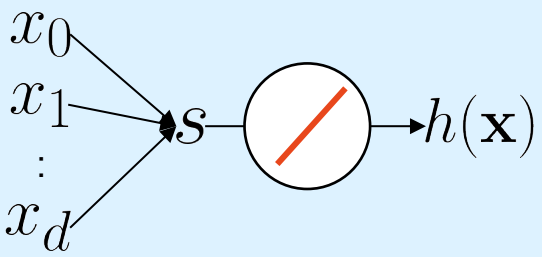


plausible err = 0/1

discrete $E_{\text{in}}(\mathbf{w})$:
→ solvable in special case

Linear Regression

$$h(\mathbf{x}) = s$$

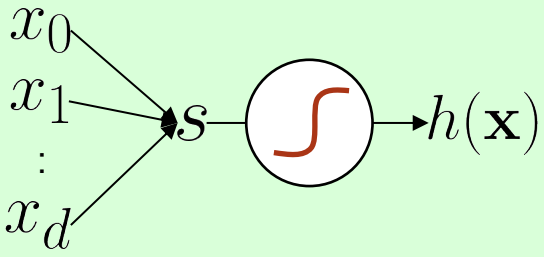


plausible err = squared

quadratic convex $E_{\text{in}}(\mathbf{w})$:
→ closed-form solution

Logistic Regression

$$h(\mathbf{x}) = \sigma(s)$$



plausible err = cross-entropy

Smooth convex $E_{\text{in}}(\mathbf{w})$:
→ gradient descent

Consider linear models first!

Lecture 1 Summary: Linear Models

- What is Machine Learning
  - Use data to improve performance (& achieve AI)
- Machine Learning Recipe
  - Essence, data, error, hypotheses
- Execution of Machine Learning
  - Optimize $\Rightarrow$ generalize
- Linear Regression
  - Analytic solution by pseudo inverse
- Logistic Regression
  - Minimize cross-entropy with gradient descent