Machine Learning

2025 Fall

Jhih-Ciang Wu

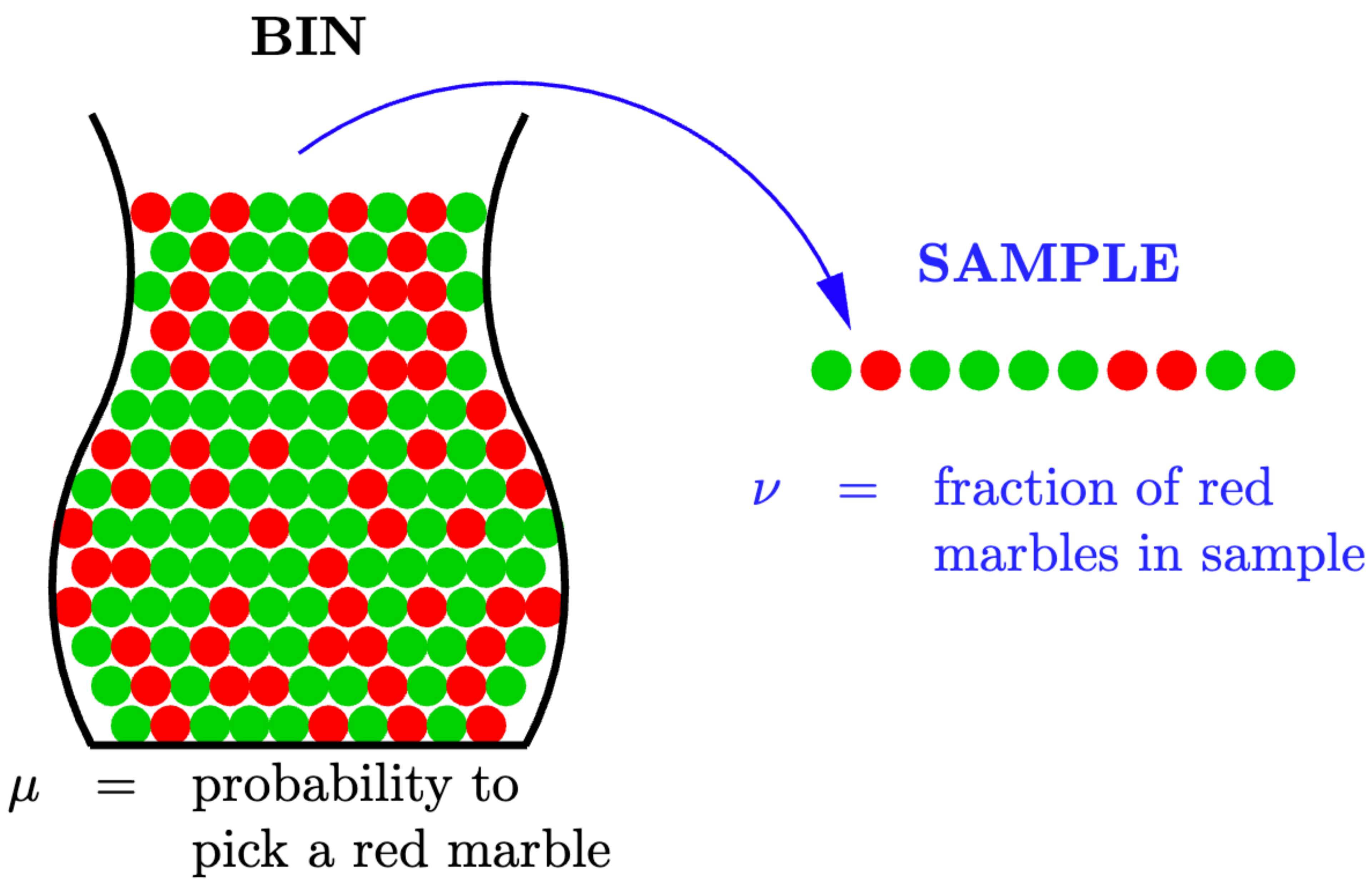
2025-09-17

國立臺灣師範大學
NATIONAL TAIWAN NORMAL UNIVERSITY

Hoeffding's Inequality

The BIN Model

- Bin with red and green marbles.
- Pick a sample of N marbles independently.
- μ: probability to pick a red marble.
- ν: fraction of red marbles in the sample.

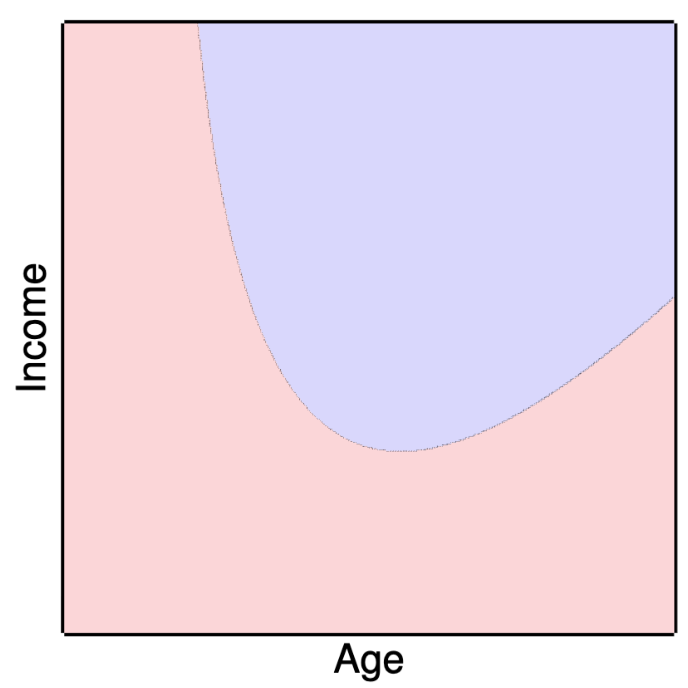


$$\mathbb{P}\left[\,|\nu - \mu| > \epsilon\right] \leq 2\exp\left(-2\epsilon^2 N\right)$$

Connection to Learning

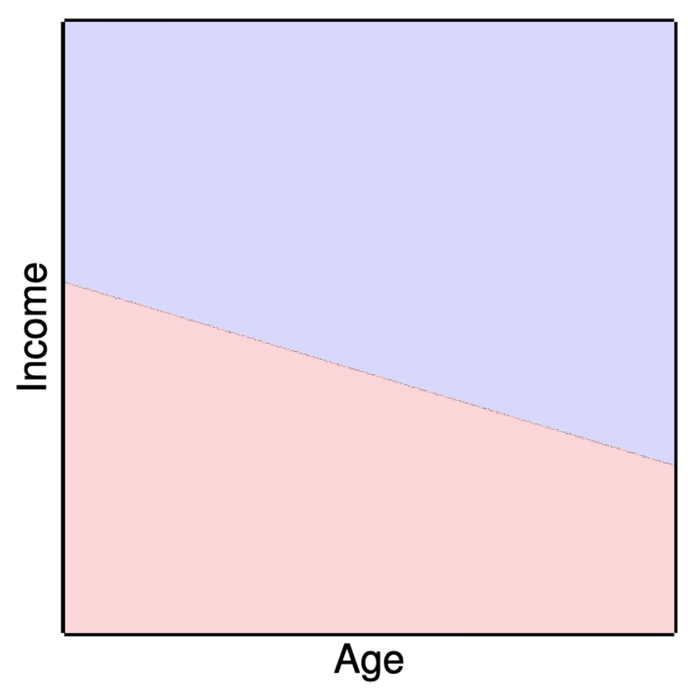
- For a fix hypothesis $h(x)$,
  - red marble $\Rightarrow h$ is wrong $\Rightarrow h(x) \neq f(x)$
  - green marble $\Rightarrow h$ is right $\Rightarrow h(x) = f(x)$

Target Function f



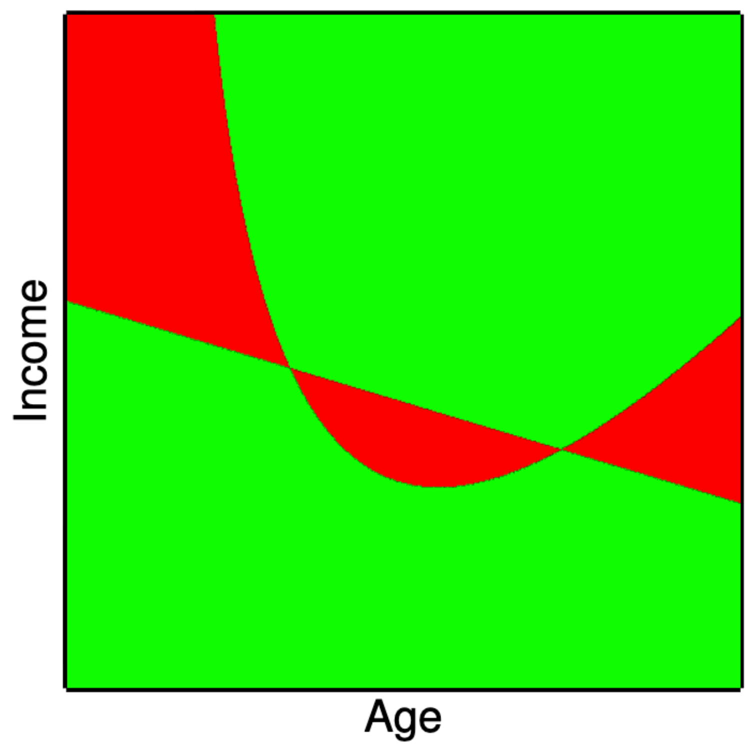
UNKNOWN

Fixed hypothesis h



KNOWN

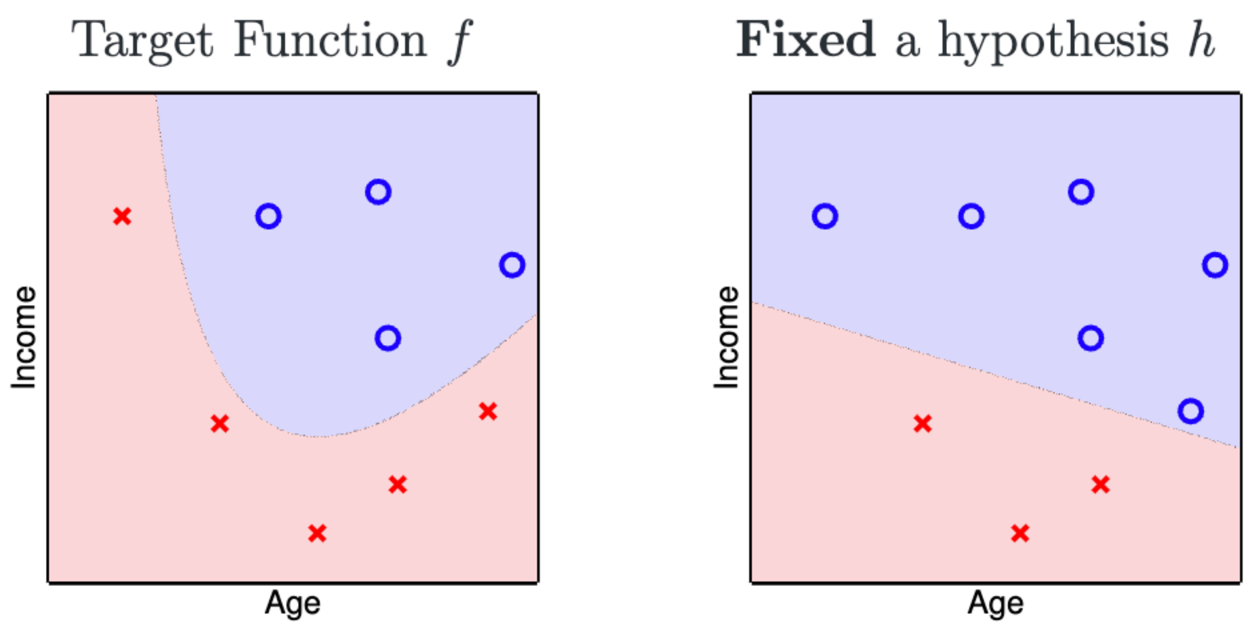
$$\mu \Rightarrow E_{\text{out}}(h) = \mathbb{P}[h(\mathbf{x}) \neq f(\mathbf{x})]$$

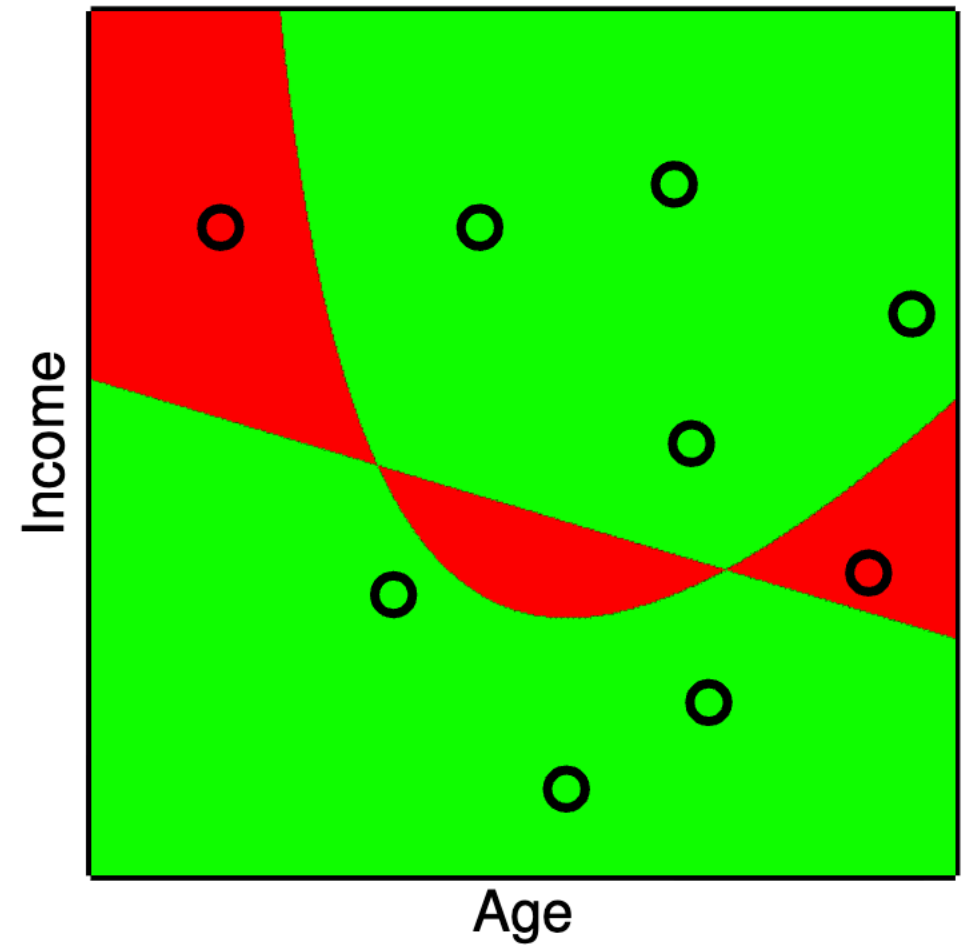


Connection to Learning

- For a fix hypothesis $h(x)$,
  - red marble $\Rightarrow h$ is wrong $\Rightarrow h(x) \neq f(x)$
  - green marble $\Rightarrow h$ is right $\Rightarrow h(x) = f(x)$

$$\nu \Rightarrow E_{\text{in}}(h) = \frac{1}{N}\sum_{n=1}^{N} [[h(\mathbf{x}_n) \neq f(\mathbf{x}_n)]]$$





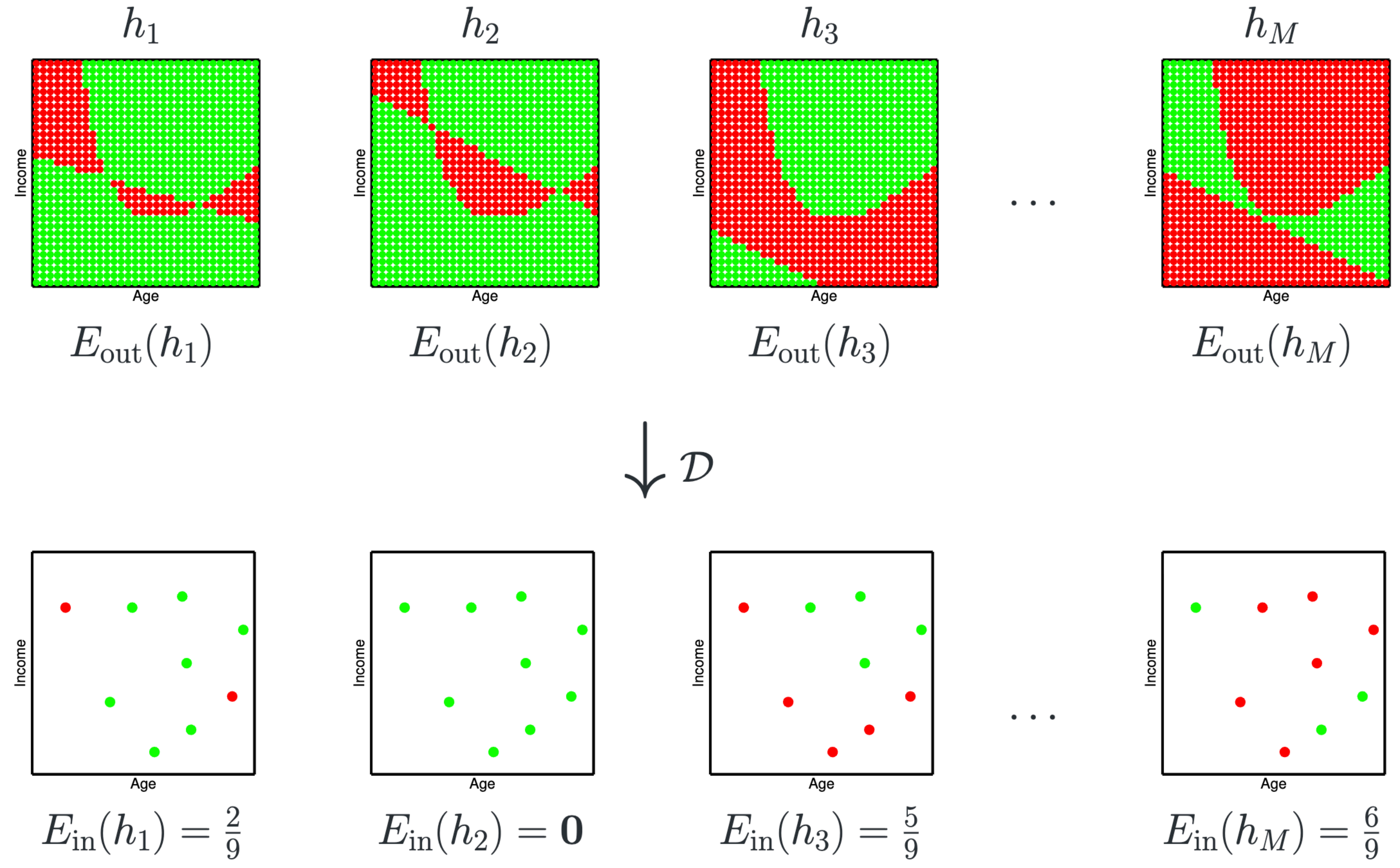
Revisit Hoeffding's Inequality

$$\mathbb{P}\left[\,|\nu - \mu| > \epsilon\right] \leq 2\exp\left(-2\epsilon^2 N\right)$$

$$\mathbb{P}\left[\,|E_{\text{in}}(h) - E_{\text{out}}(h)| > \epsilon\right] \leq 2\exp\left(-2\epsilon^2 N\right)$$

- Can we claim 'good learning'?
- This is verification, not real learning.

Real learning



Real learning

- We would like to claim that $\mathbb{P}\left[|E_{\text{in}}(g) - E_{\text{out}}(g)| > \epsilon \right]$ is small for the final hypothesis g.

$$
\begin{aligned}
|E_{\text{in}}(g) - E_{\text{out}}(g)| > \epsilon \Rightarrow\ & |E_{\text{in}}(h_1) - E_{\text{out}}(h_1)| > \epsilon \\
\text{or }\ & |E_{\text{in}}(h_2) - E_{\text{out}}(h_2)| > \epsilon \\
& \dots \\
\text{or }\ & |E_{\text{in}}(h_M) - E_{\text{out}}(h_M)| > \epsilon
\end{aligned}
$$

Real learning

- We would like to claim that $\mathbb{P}\left[\,|E_{\text{in}}(g) - E_{\text{out}}(g)| > \epsilon\right]$ is small for the final hypothesis g.

$$\mathbb{P}\left[\,|E_{\text{in}}(g) - E_{\text{out}}(g)|\,\right] > \epsilon \leq \mathbb{P}[\ |E_{\text{in}}(h_1) - E_{\text{out}}(h_1)| > \epsilon$$

$$\text{or } |E_{\text{in}}(h_2) - E_{\text{out}}(h_2)| > \epsilon$$

$$\dots$$

$$\text{or } |E_{\text{in}}(h_M) - E_{\text{out}}(h_M)| > \epsilon]$$

$$\leq \sum_{m=1}^{M} \mathbb{P}\left[\,|E_{\text{in}}(h_m) - E_{\text{out}}(h_m)|\,\right] > \epsilon$$

$$\leq 2M \exp\left(-2\epsilon^2 N\right)$$

Real learning

- We enabled this is the Hoeffding Inequality for learning

$$\mathbb{P}\left[\,|E_{\text{in}}(g) - E_{\text{out}}(g)|\,\right] > \epsilon \leq 2M \exp\left(-2\epsilon^2 N\right)$$

$$M = |\mathcal{H}|$$

The Linear Model

Some slides are modified from Prof. Yaser Abu-Mostafa,
Prof. Malik Magdon-Ismail, and Prof. Hsuan-Tien Lin

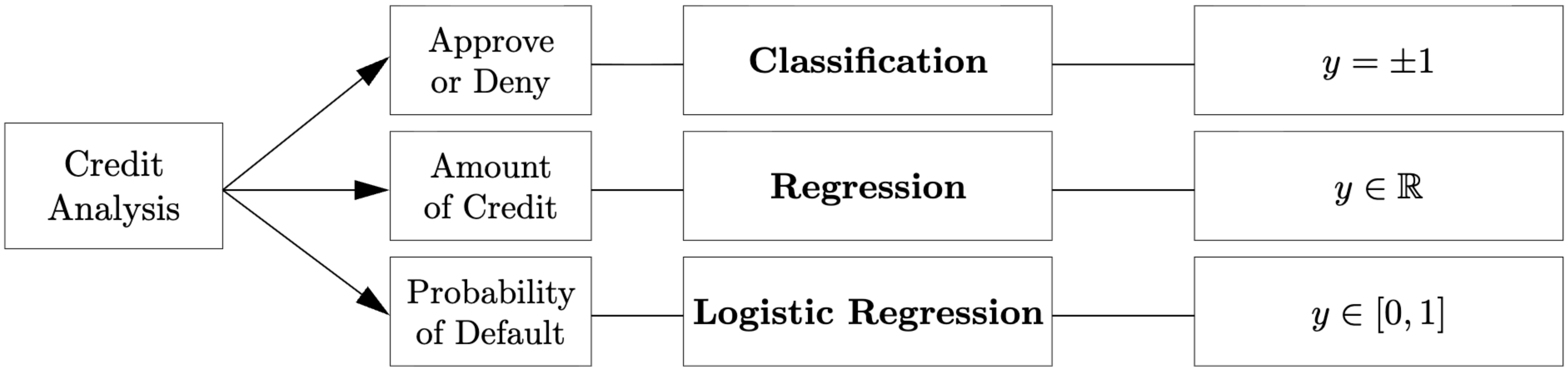
Why a line?

- We often wonder how to draw a line between two categories.
  - right versus wrong
  - personal versus professional life
  - useful email versus spam,

Linear Model

- The aim is to develop the basic linear model into a powerful tool for learning from data.
- Three important problems:
  - Classification
  - Regression
  - Logistic regression

Linear Model



Linear classification

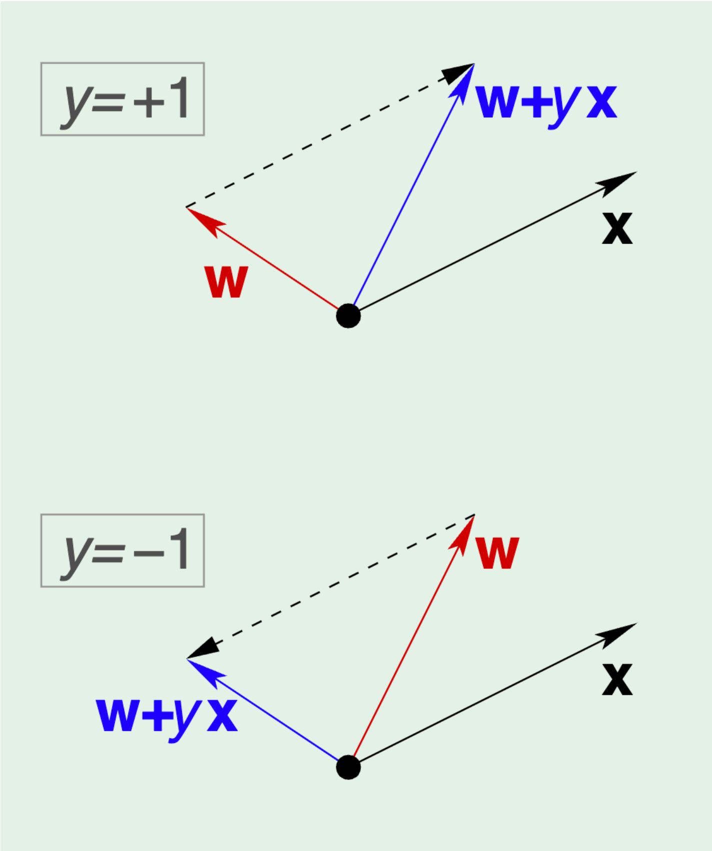
- The linear model for classifying data into two classes uses a hypothesis set of linear classifiers, where each h has the form

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

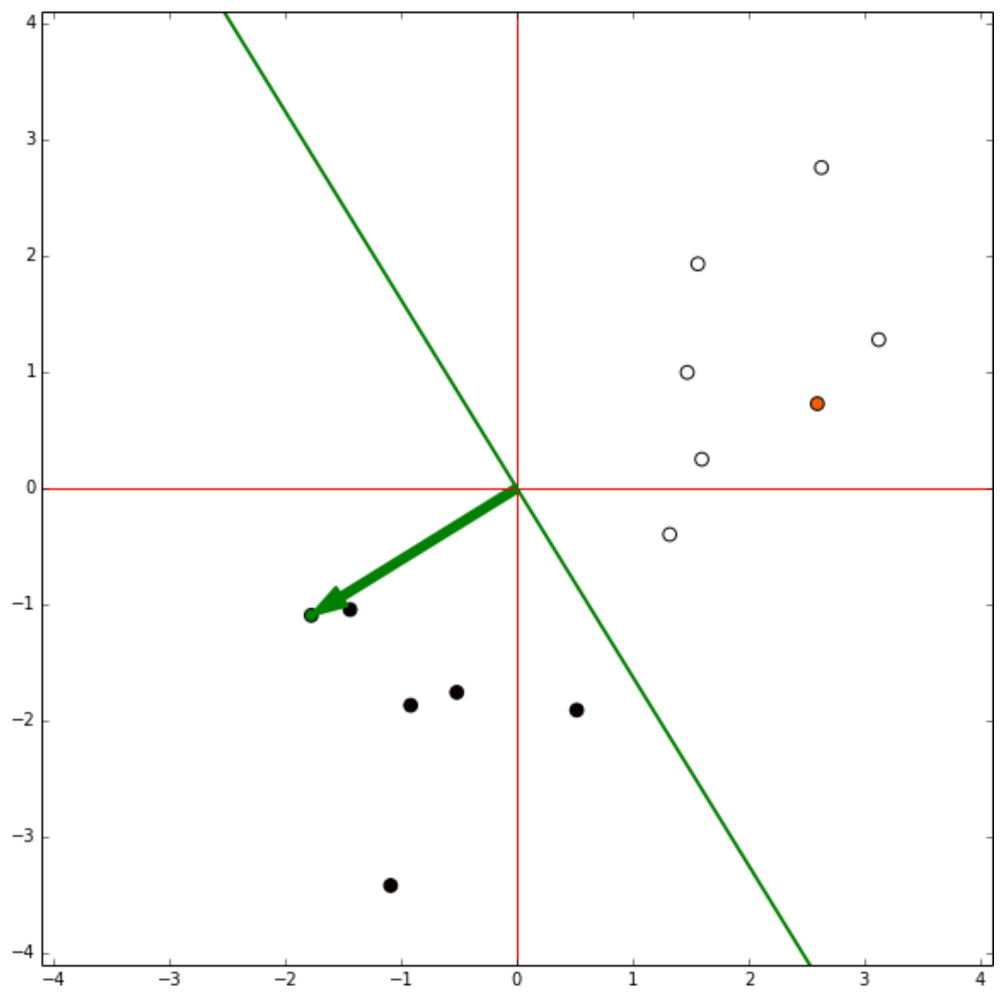
Perceptron learning algorithm

- Start with an arbitrary weight vector $w(0)$
- Then, at every time step$t \geq 0$, select *any* misclassified data point $(x(t), y(t))$, and update $w(t)$ as

$$w(t+1) = w(t) + y(t)x(t)$$

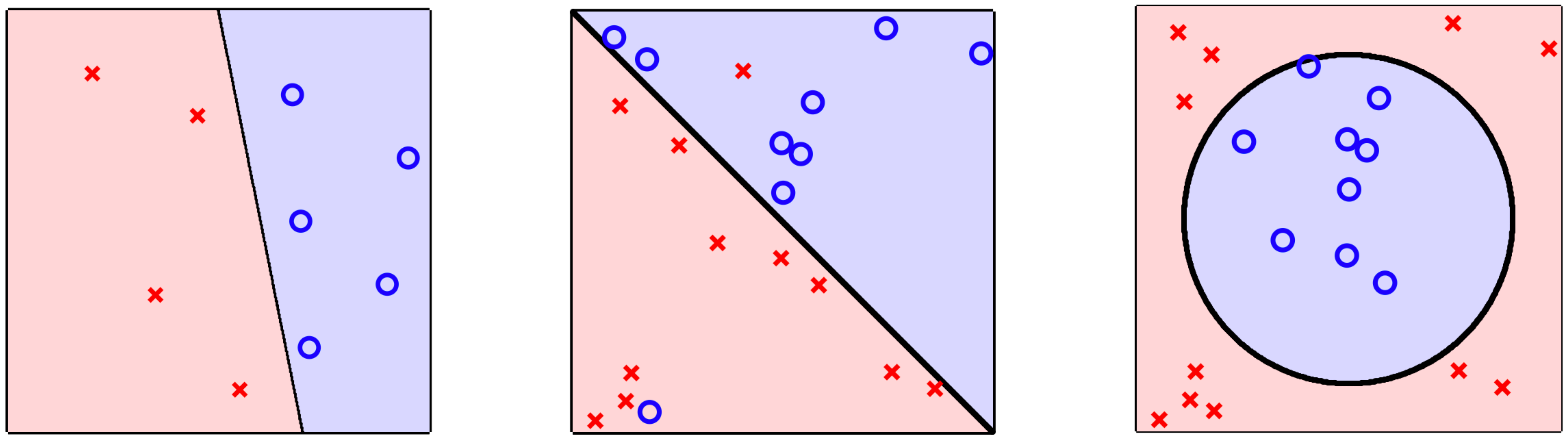


Perceptron learning algorithm



- If *linearly separable data* holds, PLA will
  - eventually stop updating
  - ending at a solution $E_{\text{in}}(w_{PLA}) = 0$
- PLA doesn't test every linear hypothesis to see if it separates the data.
- Using an iterative approach, the PLA manages to search an infinite hypothesis set and output a linear separator in finite time.

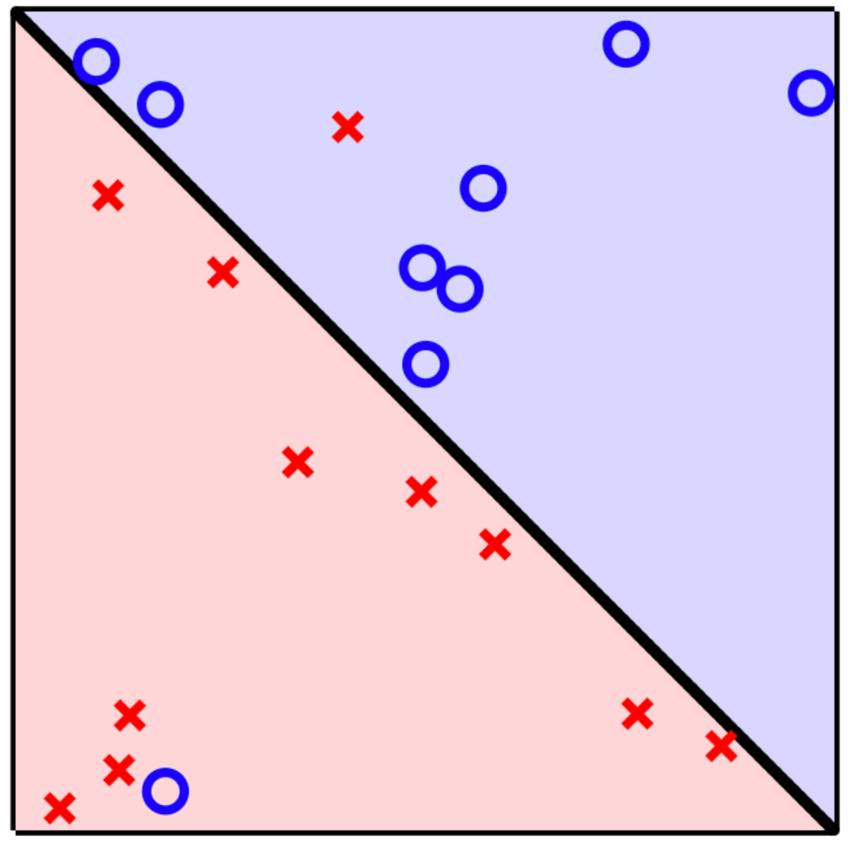
Linear Separability



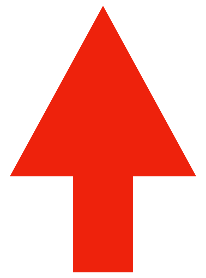
(linear separable) (not linear separable) (not linear separable)

Non-Separable Data

- The data becomes linearly separable after the removal of just two examples, which could be considered noisy examples or outliers.



(not linear separable)

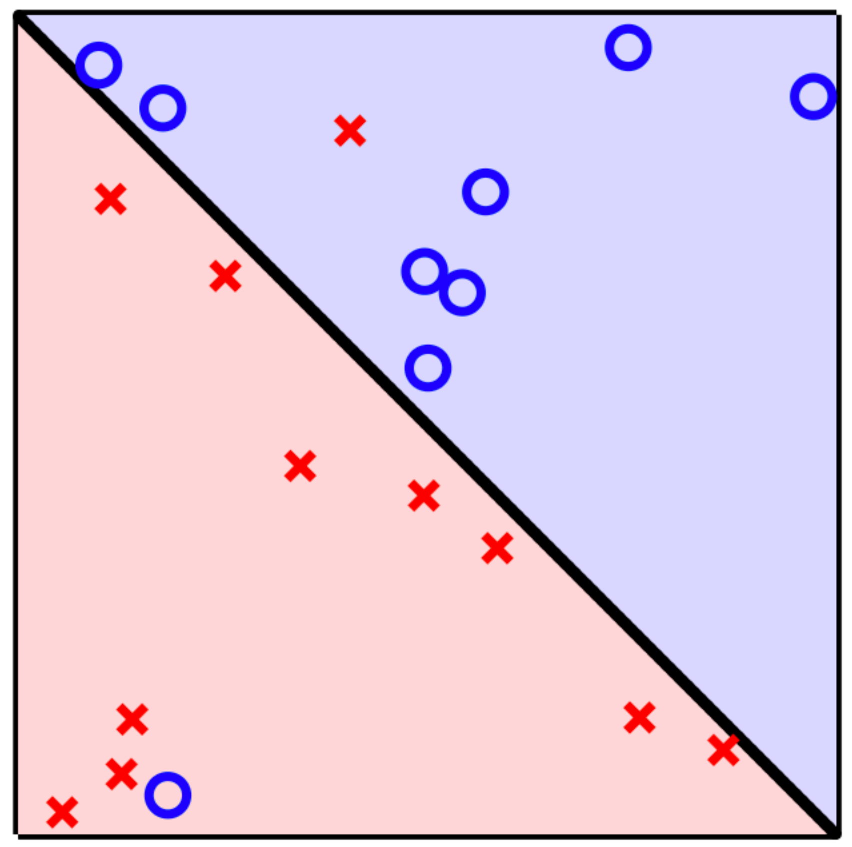


Few noisy data

Non-Separable Data

- There will always be a misclassified training example if we insist on using a linear hypothesis, and hence PLA will never terminate.

- It seems appropriate to stick with a line, but to somehow tolerate noise and output a hypothesis with a small E_{in}, not necessarily $E_{\text{in}} = 0$.



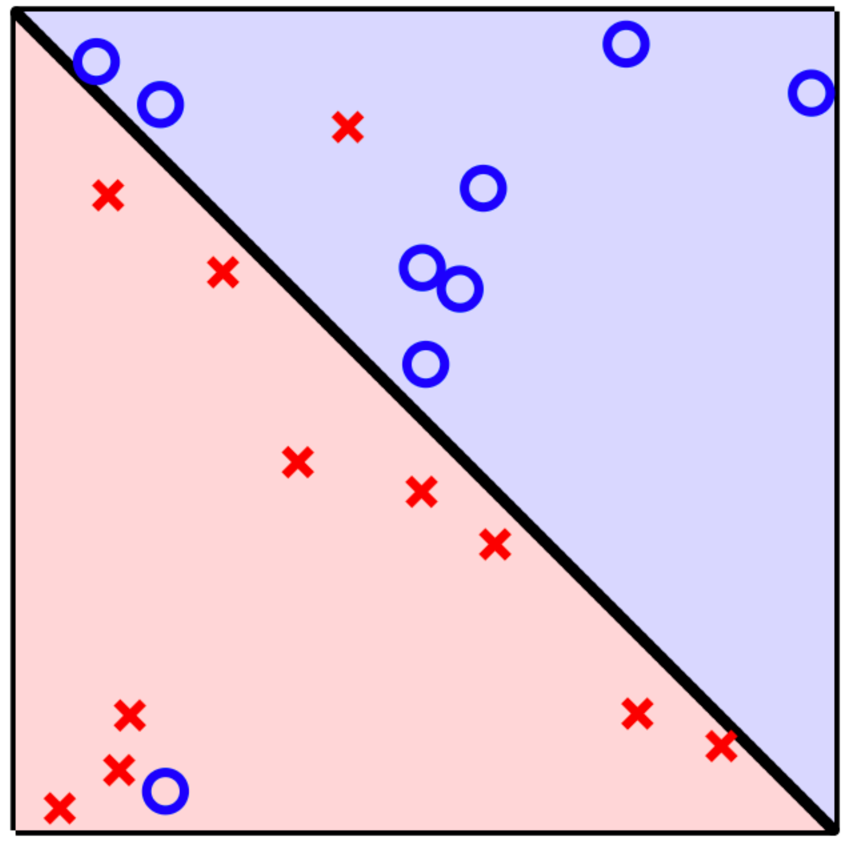
(not linear separable)

Non-Separable Data

- To find a hypothesis with the minimum E_{in}, we need to solve the combinatorial optimization problem:

$$\min_{\mathbf{w} \in \mathbb{R}^{d+1}} \frac{1}{N} \sum_{n=1}^{N} [[\operatorname{sign}(\mathbf{w}^\top \mathbf{x}_n) \neq y_n]]$$

- NP-hard due to the discrete nature.



(not linear separable)

The pocket algorithm

Pocket algorithm

- One approach for getting an approximate solution is to extend PLA through a simple modification.
- Essentially, the pocket algorithm keeps *in its pocket* the best weight vector encountered up to iteration t in PLA.
- At the end, the best weight vector will be reported as the final hypothesis. This simple algorithm is shown below.

The pocket algorithm

The pocket algorithm:

Set the pocket weight vector $\hat{\mathbf{w}}$ to $\mathbf{w}(0)$ of PLA.
for $t = 0, \ldots, T \quad 1$ **do**
    Run PLA for one update to obtain $\mathbf{w}(t+1)$.
    Evaluate $E_{\text{in}}(\mathbf{w}(t+1))$.
    If $\mathbf{w}(t+1)$ is better than $\hat{\mathbf{w}}$ in terms of E_{in}, set $\hat{\mathbf{w}}$ to $\mathbf{w}(t+1)$.
Return $\hat{\mathbf{w}}$.

The pocket algorithm

- The original PLA only checks some of the examples using $w(t)$ to identify $(x(t), y(t))$ in each iteration.

- The pocket algorithm needs an additional step that evaluates all examples using $w(t+1)$ to get $E_{\text{in}}w(w(t+1))$.

- Slower and no guarantee to converge to a good E_{in}.

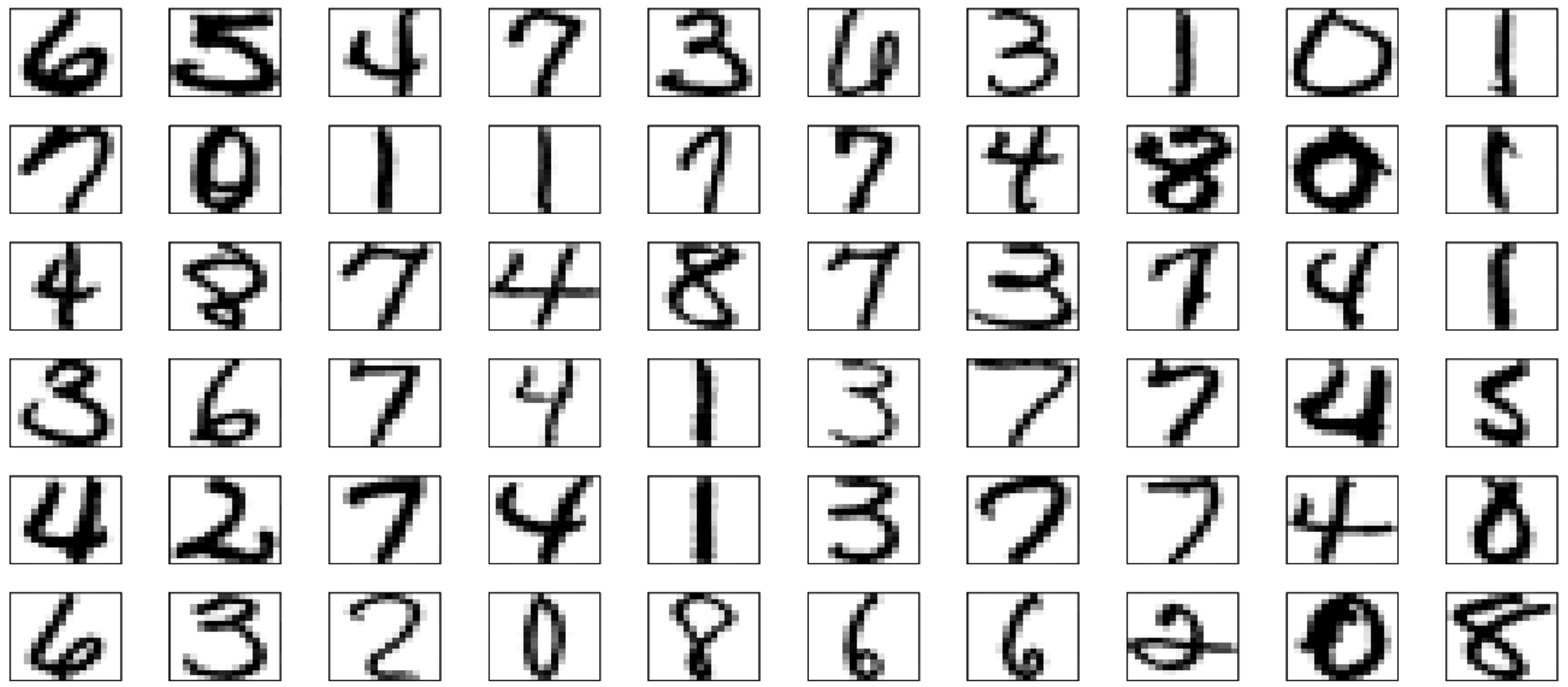
The pocket algorithm:

1: Set the pocket weight vector $\hat{\mathbf{w}}$ to $\mathbf{w}(0)$ of PLA.
2: **for** $t = 0, \ldots, T \quad 1$ **do**
3: Run PLA for one update to obtain $\mathbf{w}(t+1)$.
4: Evaluate $E_{\text{in}}(\mathbf{w}(t+1))$.
5: If $\mathbf{w}(t+1)$ is better than $\hat{\mathbf{w}}$ in terms of E_{in}, set $\hat{\mathbf{w}}$ to $\mathbf{w}(t+1)$.
6: Return $\hat{\mathbf{w}}$.

Example

Handwritten digit recognition

- US Postal Service Zip Code Database
- 16×16 pixel images are preprocessed from the scanned handwritten zip codes.
- The goal is to recognize the digit in each image.



Common confusion occurs between the digits $\{4, 9\}$ and $\{2, 7\}$.

Typical human E_{out} is about 2.5%.

Handwritten digit recognition

- Let's first decompose the big task of separating ten digits into smaller tasks of separating two of the digits, 1 and 5 (multiclass to binary classification).

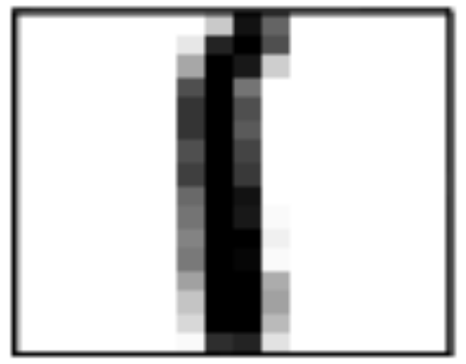
- A human approach to determining the digit corresponding to an image is to look at the shape (or other properties) of the black pixels.

- Thus, rather than carrying all the information in the 256 pixels, it makes sense to summarize the information contained in the image into a few **features**.

Features of digits

- Intensity

  ➡ Digit 5 usually occupies more black pixels than digit 1 , and hence the average pixel intensity of digit 5 is higher.

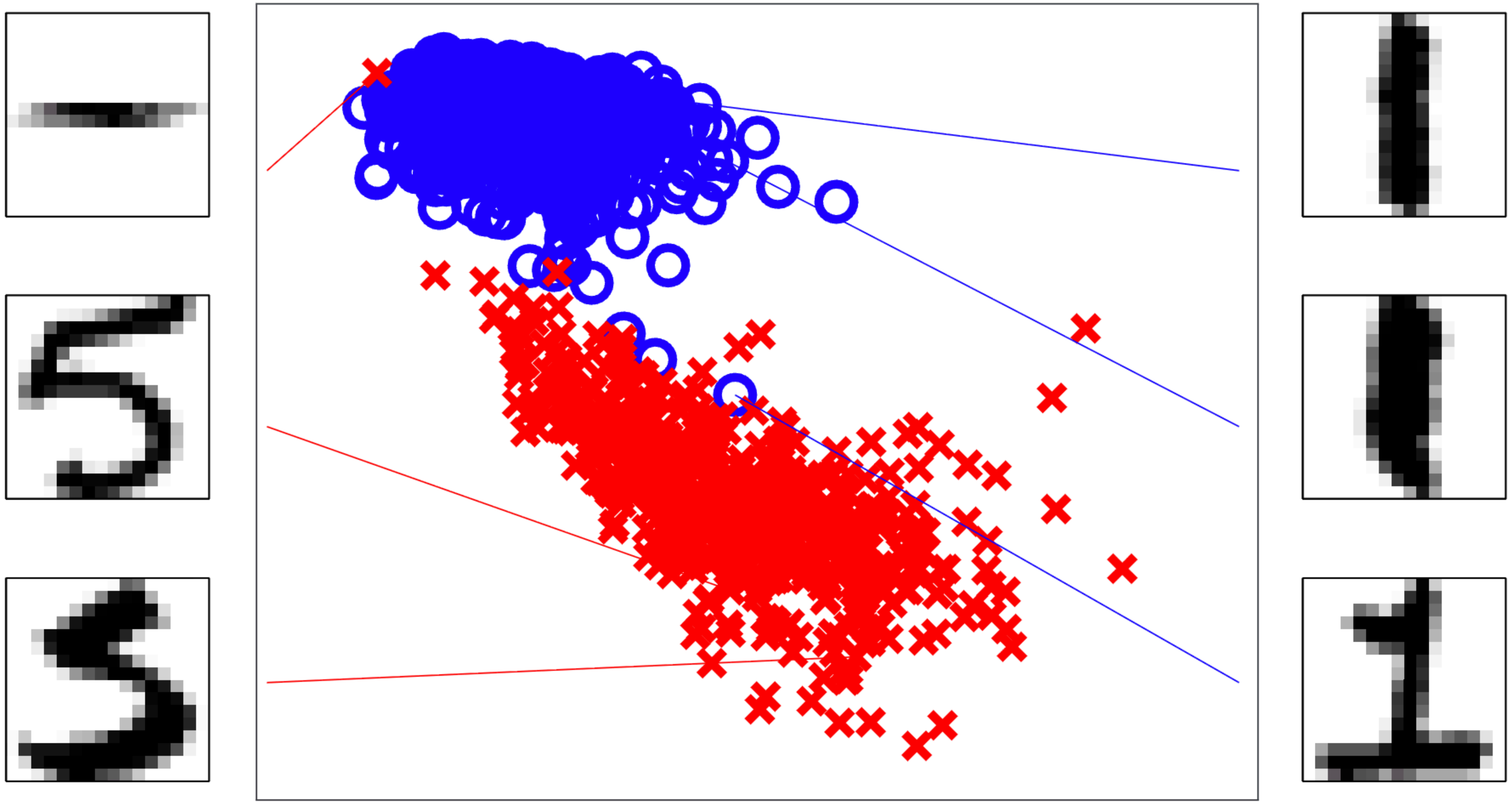


- Symmetric

  ➡ Digit 1 is symmetric while digit 5 is not.

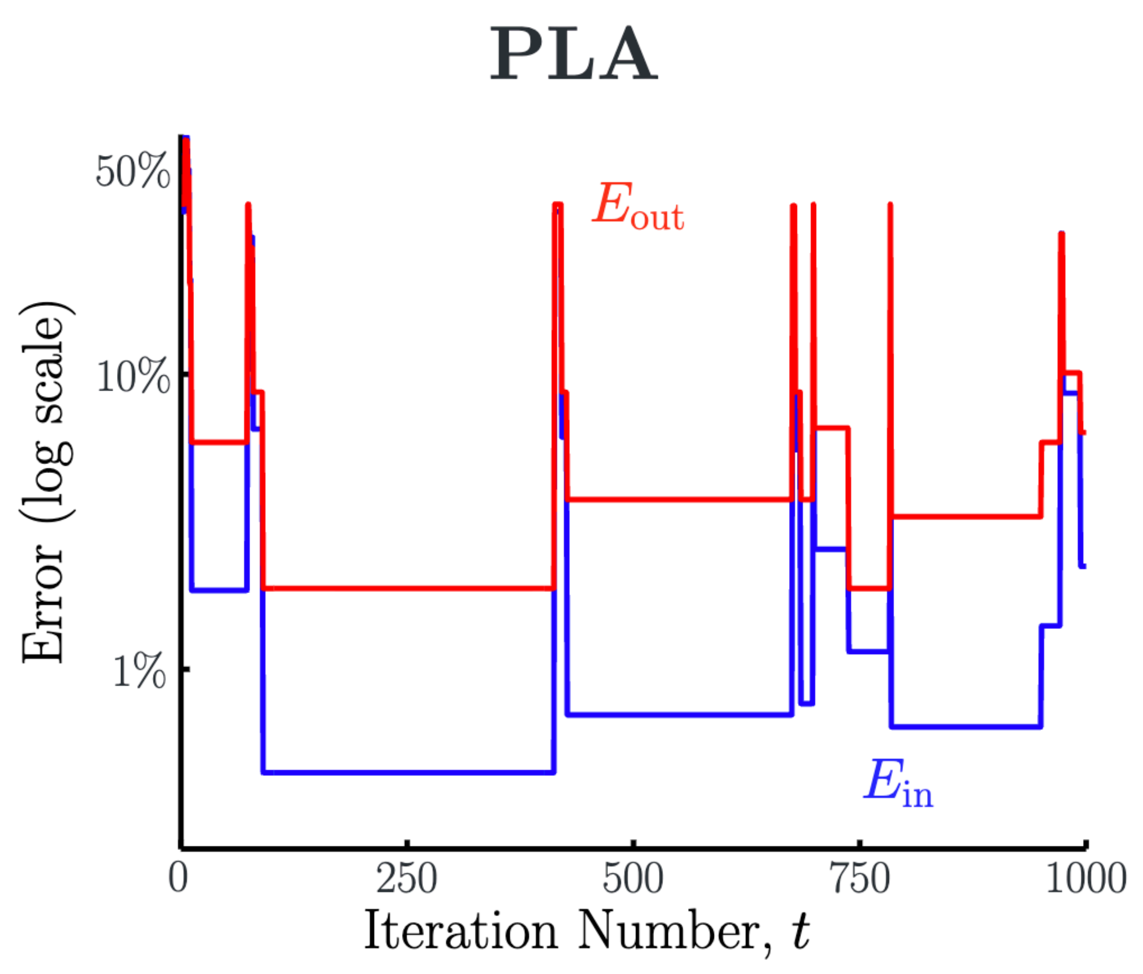
  ➡ If we define asymmetry as the average absolute difference between an image and its flipped versions, digit 1 would result in a higher symmetry value.

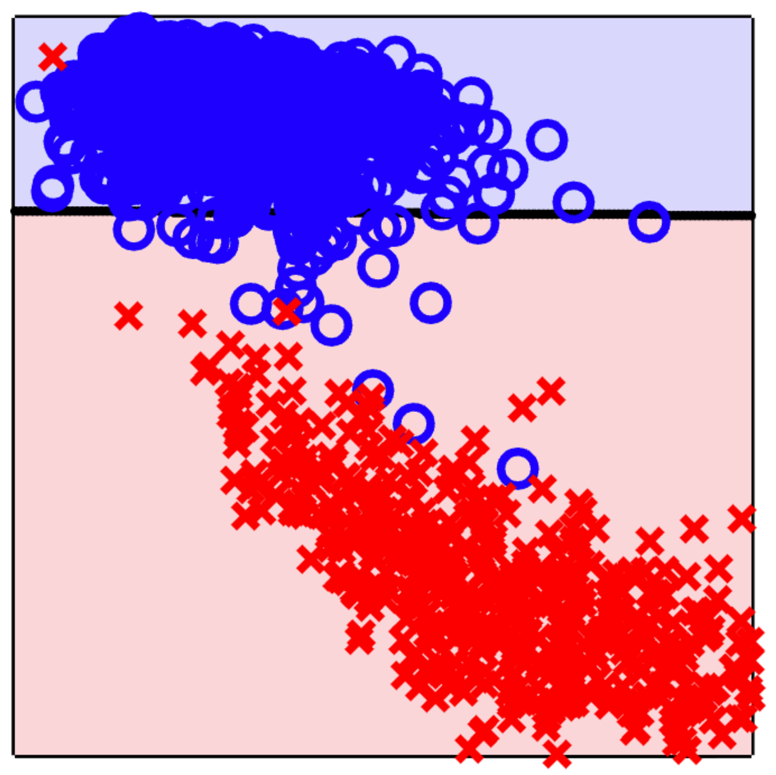
Intensity and Symmetry Features



PLA on Digits Data

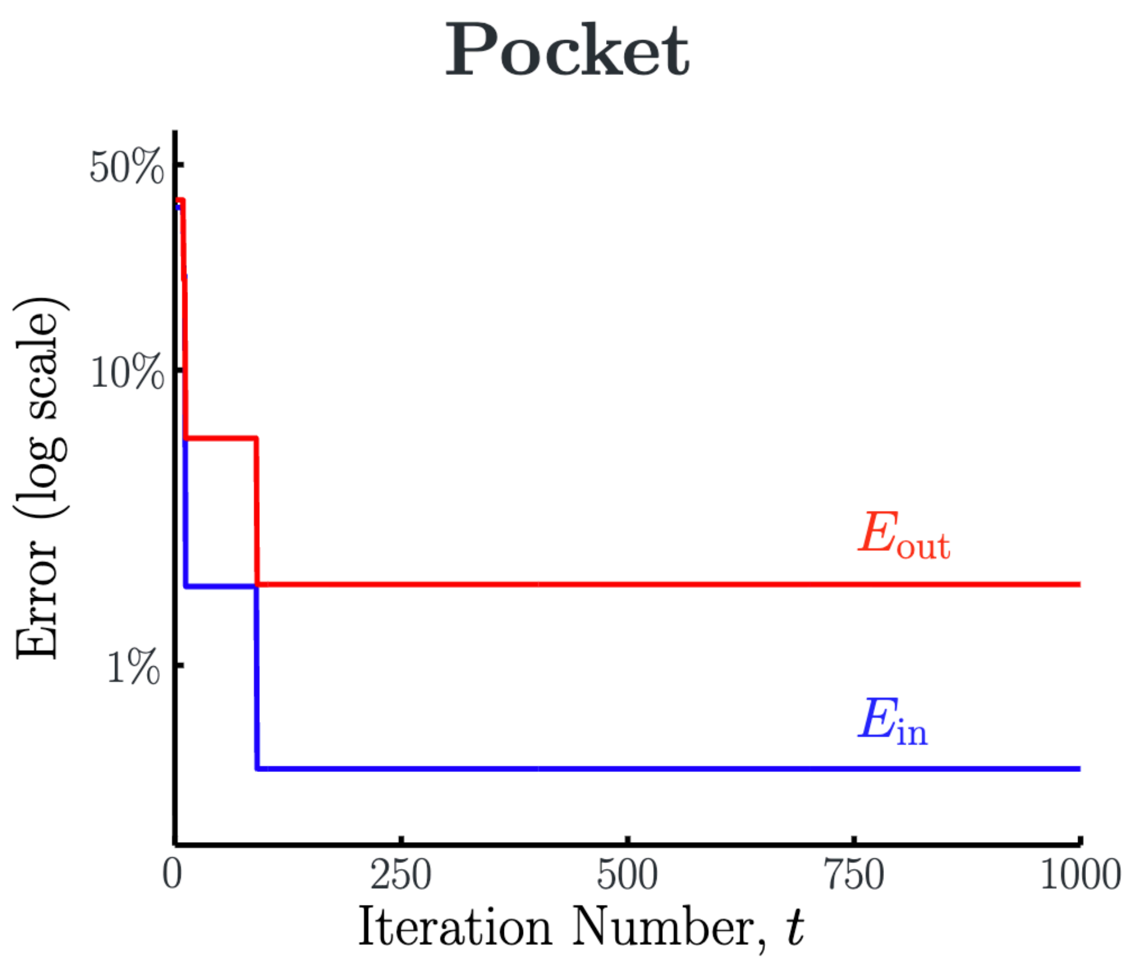
- Since the dataset is not linearly separable, PLA will not stop updating.
- In fact, its behavior can be quite unstable.
- When it is forcibly terminated at iteration 1,000, PLA gives a line that has a poor $E_{\text{in}} = 2.24\,\%$ and $E_{\text{out}} = 6.37\,\%$.

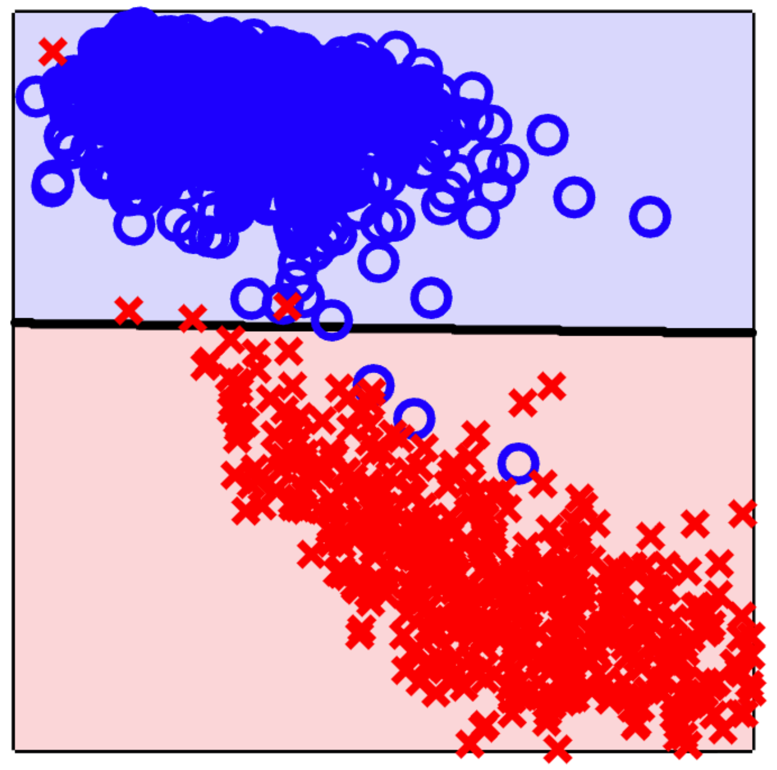




Pocket on Digits Data

- More stable stratege.
- can obtain a line
- Result in a better $E_{\text{in}} = 0.45\,\%$ and a better $E_{\text{out}} = 1.89\,\%$.





Linear Regression

- Let us revisit our application in credit approval, this time considering a regression problem rather than a classification problem.

- approve or not $\Rightarrow$ proper credit limit for each approved customer.

- x_n represents the customer information and $y_n \in \mathbb{R}$ is the credit limit set by one of the human experts in the bank.

- The bank wants to use learning to find a hypothesis g that replicates how human experts determine credit limits.

Linear Regression

- Since there is more than one human expert, and since each expert may not be perfectly consistent, our target will not be a deterministic function $y = f(x)$.

- Instead, it will be a noisy target formalized as a distribution of the random variable y that comes from the different views of different experts as well as the variation within the views of each expert.

- That is, the label y_n comes from some distribution $P(y \mid x)$ instead of a deterministic function $f(x)$.

stochastic = randomly determined

Linear Regression

- The linear regression algorithm is based on minimizing the squared error between $h(x)$ and y.

$$E_{\text{out}}(h) = \mathbb{E}\left[(h(x) - y)^2\right]$$

- Similar to what we did in classification, the in-sample version is

$$E_{\text{in}}(h) = \frac{1}{N}\sum_{n=1}^{N}(h(x_n) - y_n)^2$$

- In linear regression, h takes the form of a linear combination of the components of x.

$$h(x) = \sum_{i=0}^{d} w_i x_i = w^\top x$$

Using Matrices for Linear Regression

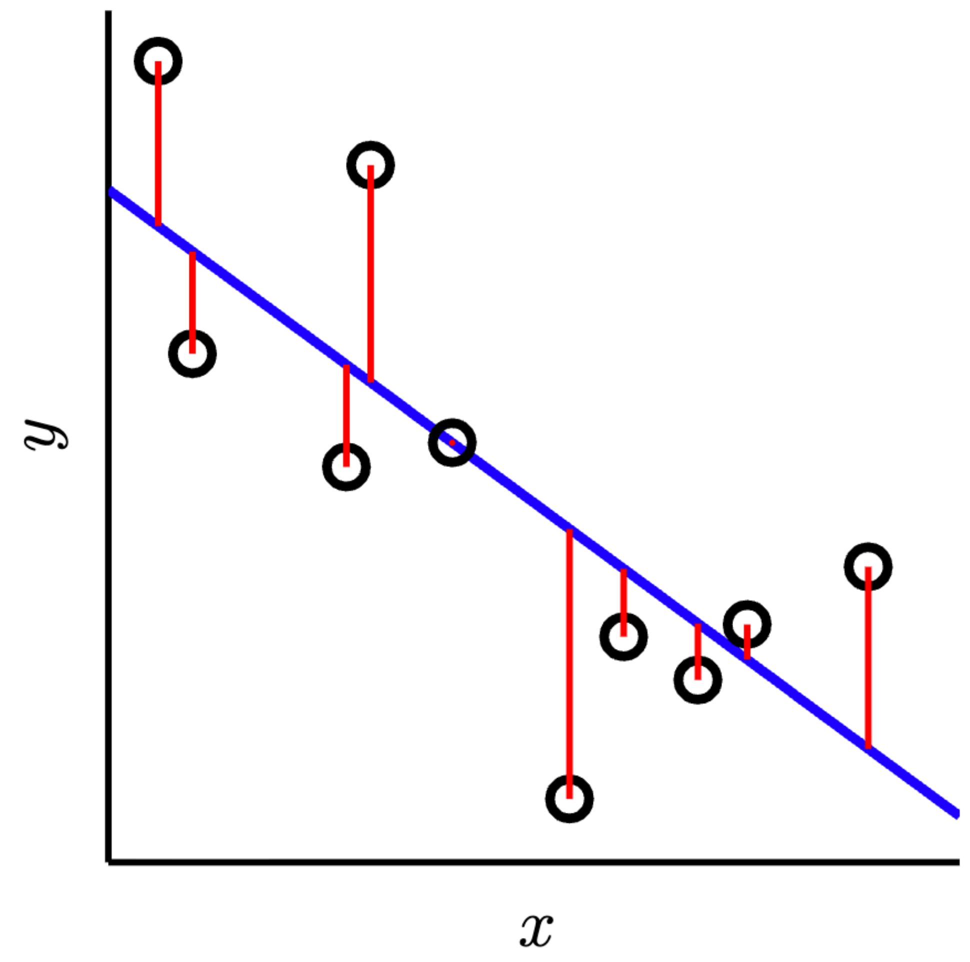
$$\underbrace{\mathrm{X} = \begin{bmatrix} —\mathbf{x}_1— \\ —\mathbf{x}_2— \\ \vdots \\ —\mathbf{x}_N— \end{bmatrix}}_{\text{data matrix, } N \times (d+1)} \qquad \underbrace{\mathbf{y} = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_N \end{bmatrix}}_{\text{target vector}} \qquad \underbrace{\hat{\mathbf{y}} = \begin{bmatrix} \hat{y}_1 \\ \hat{y}_2 \\ \vdots \\ \hat{y}_N \end{bmatrix} = \begin{bmatrix} \mathbf{w}^{\mathrm{T}}\mathbf{x}_1 \\ \mathbf{w}^{\mathrm{T}}\mathbf{x}_2 \\ \vdots \\ \mathbf{w}^{\mathrm{T}}\mathbf{x}_N \end{bmatrix} = \mathrm{X}\mathbf{w}}_{\text{in-sample predictions}}$$

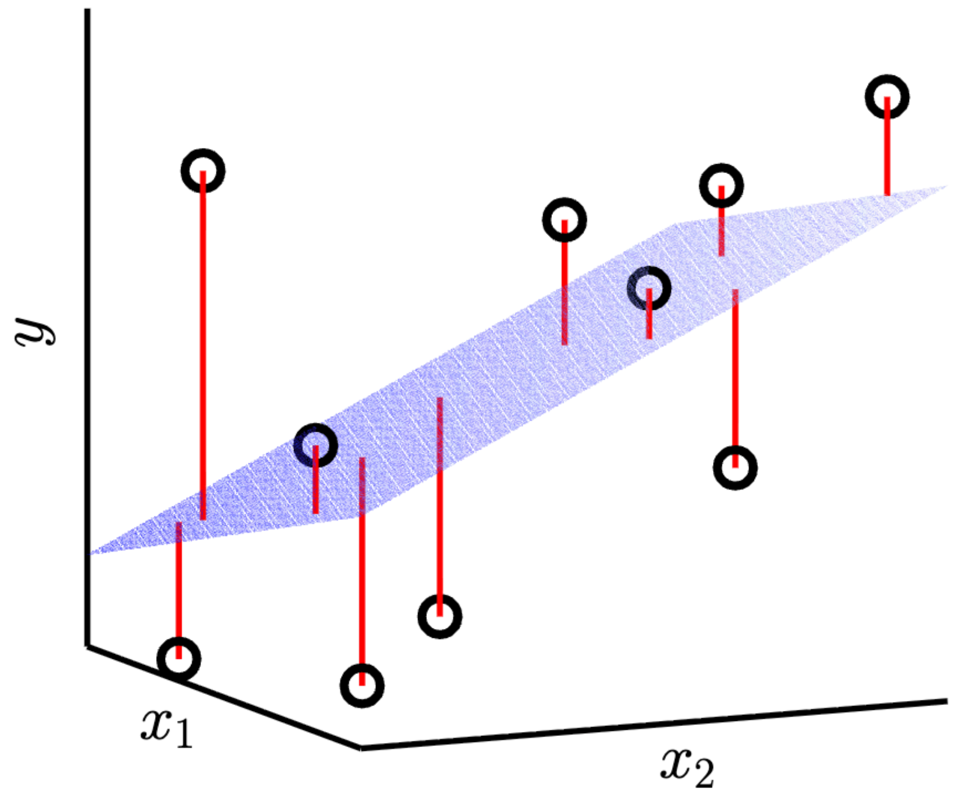
$$\begin{aligned} E_{\text{in}}(w) &= \frac{1}{N}\sum_{n=1}^{N}(\hat{y}_n - y_n)^2 \\ &= \frac{1}{N}\|\hat{y} - y\|_2^2 \\ &= \frac{1}{N}\|Xw - y\|_2^2 \\ &= \frac{1}{N}(w^\top X^\top Xw - 2w^\top X^\top y + y^\top y) \end{aligned}$$

Optimization

- The linear regression algorithm is derived by minimizing $E_{\text{in}}(w)$ over all possible $w \in \mathbb{R}^{d+1}$, as formalized by the following optimization problem

$$w_{lin} = \underset{w \in \mathbb{R}^{d+1}}{\operatorname{argmin}} \; E_{\text{in}}(w)$$





Linear Regression Solution

$$E_{\text{in}}(w) = \frac{1}{N}(w^\top X^\top X w - 2w^\top X^\top y + y^\top y)$$

$$\nabla E_{\text{in}}(w) = \frac{2}{N}(X^\top X w - X^\top y)$$

- Setting $\nabla E_{\text{in}}(w) = 0$, one should solve for w that satisfies

$$X^\top X w = X^\top y$$

- If $X^\top X$ is invertible,

$$w = X^\dagger y,$$

where $X^\dagger = (X^\top X)^{-1} X^\top$ is the pseudo-inverse of X.

Vector Calculus: To minimize $E_{\text{in}}(\mathbf{w})$, set $\nabla_{\mathbf{w}} E_{\text{in}}(\mathbf{w}) = \mathbf{0}$.

$$\nabla_{\mathbf{w}}(\mathbf{w}^{\mathsf{T}} \mathrm{A} \mathbf{w}) = (\mathrm{A} + \mathrm{A}^{\mathsf{T}})\mathbf{w}, \qquad \nabla_{\mathbf{w}}(\mathbf{w}^{\mathsf{T}} \mathbf{b}) = \mathbf{b}.$$

$\mathrm{A} = \mathrm{X}^{\mathsf{T}}\mathrm{X}$ and $\mathbf{b} = \mathrm{X}^{\mathsf{T}}\mathbf{y}$:

Linear regression algorithm

Linear Regression Algorithm:

1. Construct the matrix X and the vector $\mathbf{y}$ from the data set $(\mathbf{x}_1, y_1), \cdots, (\mathbf{x}_N, y_N)$, where each $\mathbf{x}$ includes the $x_0 = 1$ coordinate,

$$\mathrm{X} = \underbrace{\begin{bmatrix} \text{---}\mathbf{x}_1\text{---} \\ \text{---}\mathbf{x}_2\text{---} \\ \vdots \\ \text{---}\mathbf{x}_N\text{---} \end{bmatrix}}_{\text{data matrix}}, \qquad \mathbf{y} = \underbrace{\begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_N \end{bmatrix}}_{\text{target vector}}.$$

2. Compute the pseudo inverse $\mathrm{X}^\dagger$ of the matrix X. If $\mathrm{X}^\mathrm{T}\mathrm{X}$ is invertible,

$$\mathrm{X}^\dagger = (\mathrm{X}^\mathrm{T}\mathrm{X})^{-1}\mathrm{X}^\mathrm{T}$$

3. Return $\mathbf{w}_{\text{lin}} = \mathrm{X}^\dagger\mathbf{y}$.

Linear regression algorithm

- It may seem that, compared with the perceptron learning algorithm, linear regression doesn't really look like *learning*.
- As long as the hypothesis w has a decent out-of-sample error, then learning has occurred.
- Linear regression is a rare case where we have an *analytic* formula for learning that is easy to evaluate.

Hat matrix

- The linear regression weight vector w_{lin} is an attempt to map the inputs X to the outputs y.

- However, w_{lin} does not produce y exactly, but produces an estimate

$$\hat{y} = Xw_{lin} = X(X^\top X)^{-1} X^\top y$$

- Let $H = X(X^\top X)^{-1} X^\top$, we have

$$\hat{y} = Hy$$

Logistic Regression

- The core of the linear model is the **signal** $s = w^\top x$ that combines the input variables linearly.

➡ classification: $h(x) = \text{sign}(w^\top x)$ output is bounded

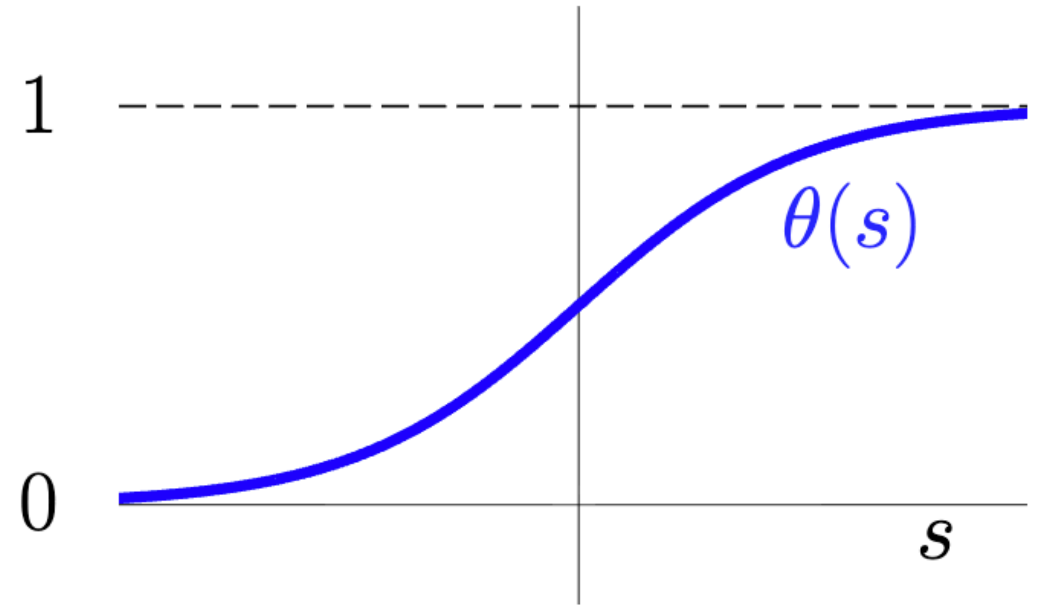
➡ regression: $h(x) = w^\top x$ output is real

➡ logistic regression: $h(x) = \theta(w^\top x)$ output is bounded and real

Logistic function

logistic function

$$\theta(s) = \frac{e^s}{1 + e^s} \in [0,1]$$



- The output can be interpreted as a probability for a binary event.
- The logistic function θ is referred to as a soft threshold, in contrast to the hard threshold in classification.
- It is also called a sigmoid because its shape looks like a flattened out 's'.
- $\theta(-s) = 1 - \theta(s)$

Assignment 1

- The homework questions are selected from the textbook Learning From Data.
- Please provide complete answers according to the corresponding question numbers.
- Due Date: 10/08 9:10 AM

Project information

- Proposal: approximately 2 page, including the problem statement, motivations (limitations for existing methods) and your potential contributions.
- Final project: full paper as regular paper, 5 pages excluding references.
- Both proposal and final project need to follow two-column formatting (will release soon).

Project information

- Please find your partners and register the following google sheet before 9/19 23:59.
- **You are considered the sole member of the project if you miss the registration deadline.**

https://docs.google.com/spreadsheets/d/1-Iqfb-ynEePjcD9as9B8opR4ynXsbgodOodlpBQz6hw/edit?usp=sharing