Lecture 2

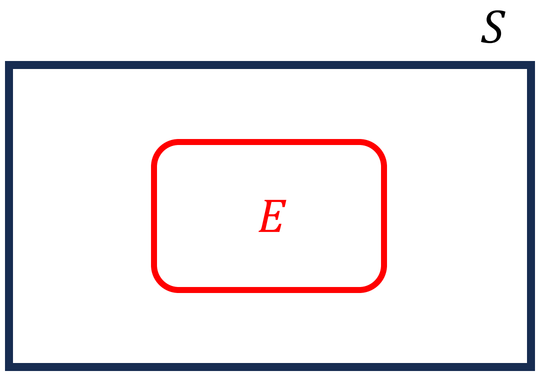
CS436/536: Introduction to Machine Learning

Zhaohan Xi

Binghamton University

zxi1@binghamton.edu

Quick Note on Probability



- Sample Space S

  The set of all possible outcomes of an experiment

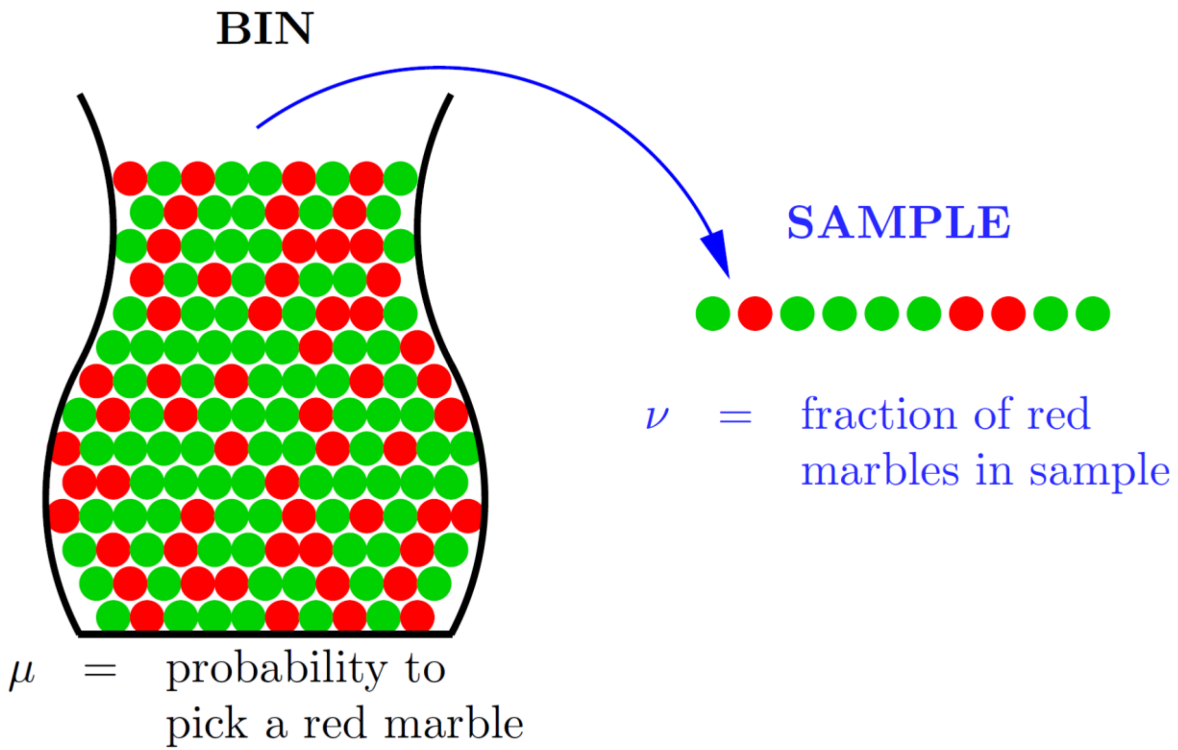
  The outcomes are mutually exclusive

Ex. $\{H, T\}$ is the sample space for the experiment of tossing a single coin

Ex. The set of all possible transactions or itemsets

- Event E is any subset of the sample space S

Estimating Population Mean from Sample Mean



Pick a *random* sample of N marbles with replacement *independently*

Observe the fraction of red marbles ν

Note: the only random quantity here is ν. μ is fixed (albeit unknown)

What does ν tell us about μ?
Nothing for sure.
But…

Estimating Population Mean from Sample Mean

Can we say anything **for certain** about μ (outside the data) having observed ν (the data)?

- No.

It is *possible* to pick only red marbles while the bin has mostly green marbles

But not probable

- See the binomial distribution

- What is the relationship between ν and μ?

Probability to the Rescue: Hoeffding's Inequality

Hoeffding / Chernoff proved that ν tends to be close to μ, most of the time

$$\boxed{\mathbb{P}[|\nu - \mu| > \epsilon] \leq 2e^{-2\epsilon^2 N}, \text{ for any } \epsilon > 0}$$

i.e. ν is approximately correct most of the time

or in other words…

probably approximately correct (PAC)

We can learn *something*!

Hoeffding's Inequality

$\mathbb{P}[|\nu - \mu| > \epsilon] \leq 2e^{-2\epsilon^2 N}$, for any $\epsilon > 0$

- Sample $N = 1{,}000$ and observe ν

99% of the time	$\mu - 0.05 \leq \nu \leq \mu + 0.05$	$\mu \in [\nu - 0.05, \nu + 0.05]$
99.9999996% of the time	$\mu - 0.10 \leq \nu \leq \mu + 0.10$	$\mu \in [\nu - 0.10, \nu + 0.10]$

Hoeffding's Inequality

$$\boxed{\mathbb{P}[|\nu - \mu| > \epsilon] \leq 2e^{-2\epsilon^2 N}, \text{ for any } \epsilon > 0}$$

- Samples must be *independent*

If the data is constructed arbitrarily, we cannot say anything about μ

Hoeffding's bound is:

- Independent of μ
- Independent of size of bin
- Depends only on
  - size of the dataset N
  - tolerance ϵ

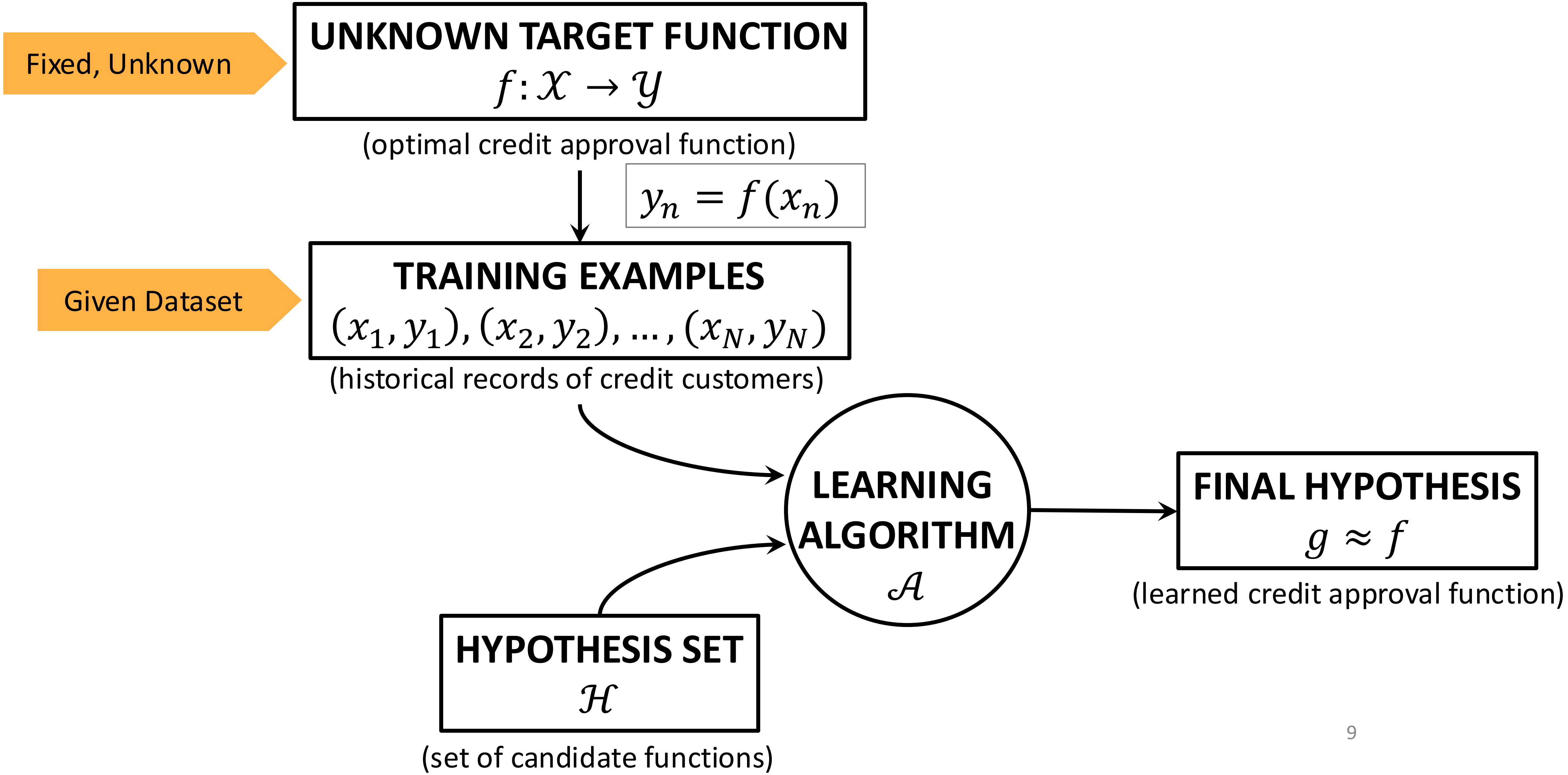
- If we desire a small ϵ, we will need large N

If $N \to \infty$, $\mu \approx \nu$ with *very* high probability

What does this have to do with ML?

- Want: Pick a function g that approximates f out-of-sample
- What a learning algorithm does:
  - Pick a function $g \in \mathcal{H}$
- How do we know if g is any good?
  - Evaluate its in-sample (training) error
- How do we evaluate in-sample error?
  - Using a sample, data generated at random
- Can we be sure that the data is truly representative of the whole population?

Learning Problem Setup



Learning

- Start with a set of candidate hypotheses $\mathcal{H}$ which likely represent f

$\mathcal{H} = \{h_1, h_2, \dots\}$ The hypothesis set or *model*

- Select a hypothesis g from $\mathcal{H}$

A Decision Problem: What is the Criterion?

Using a *learning algorithm*

A Computational Problem

- Use g for new customers

Hope that $g \approx f$

$\mathcal{X}, \mathcal{Y}$ and $\mathcal{D}$ are ***given*** by the learning problem

The target function f is **fixed but unknown**

We choose $\mathcal{H}$ and the learning algorithm

Credit Approval

- Using salary, debt, years in residence, etc., approve for credit or not
- Nobody has an optimal credit approval formula
- But banks have data
  - Customer information
  - Credit history

age	33 years
salary	50,000
debt	27,500
years employed	1
years at residence	2
…	…

Approve for credit?

Credit Approval

age	33 years
salary	50,000
debt	27,500
years employed	1
years at residence	2
…	…

Approve for credit?

Compute a "credit score"

age	**salary**	**debt**	**…**
x_1	x_2	x_3	…
w_1	w_2	w_3	…

$$creditscore = w_1 x_1 + w_2 x_2 + w_3 x_3 + \cdots$$

A Simple Learning Model

- Input vector $\boldsymbol{x} = [x_1, x_2, \dots, x_d]^T$
- Compute a "credit score" by giving

importance weights to the different inputs: $creditscore = \sum_{i=1}^{d} w_i x_i$

- Decision rule:
  - If $creditscore > threshold$: **Approve** credit (good credit score)
  - If $creditscore < threshold$: **Deny** credit (poor credit score)

- How to choose the importance weights w_i?
  - input x_i is important in deciding credit approval $\Rightarrow$ large w_i
  - input x_i has a beneficial effect to credit $\Rightarrow w_i > 0$ (weighs positively)
  - input x_i has an adverse effect on credit $\Rightarrow w_i < 0$ (weighs negatively)

A Simple Learning Model

- Decision rule:
  - If $creditscore > threshold$: **Approve** credit (good credit score) $\Rightarrow$ output **+1**
  - If $creditscore < threshold$: **Deny** credit (poor credit score) $\Rightarrow$ output **-1**

- Can be written formally as:

$$h(\boldsymbol{x}) = sign\left(\left(\sum_{i=1}^{d} w_i x_i\right) - threshold\right)$$

Simplifying a little…

$$h(\boldsymbol{x}) = sign\left(\left(\sum_{i=1}^{d} w_i x_i\right) + w_0 1\right)$$

w_0 is a "bias weight" which corresponds to the threshold: Approve if $\sum_{i=1}^{d} w_i x_i > w_0$

A Simple Learning Model

$$h(\boldsymbol{x}) = sign\left(\left(\sum_{i=1}^{d} w_i x_i\right) + w_0 1\right)$$

$$= sign(w_0 1 + w_1 x_1 + w_2 x_2 + \cdots + w_d x_d)$$

$$\boldsymbol{w} = \begin{bmatrix} w_0 \\ w_1 \\ w_2 \\ \ldots \\ w_d \end{bmatrix} \in \mathbb{R}^{d+1} \qquad \boldsymbol{x} = \begin{bmatrix} x_0 \\ x_1 \\ x_2 \\ \ldots \\ x_d \end{bmatrix} \in 1 \times \mathbb{R}^{d} \text{ (where } x_0 = 1\text{)}$$

$$h(\boldsymbol{x}) = sign(\boldsymbol{w}^T \boldsymbol{x})$$

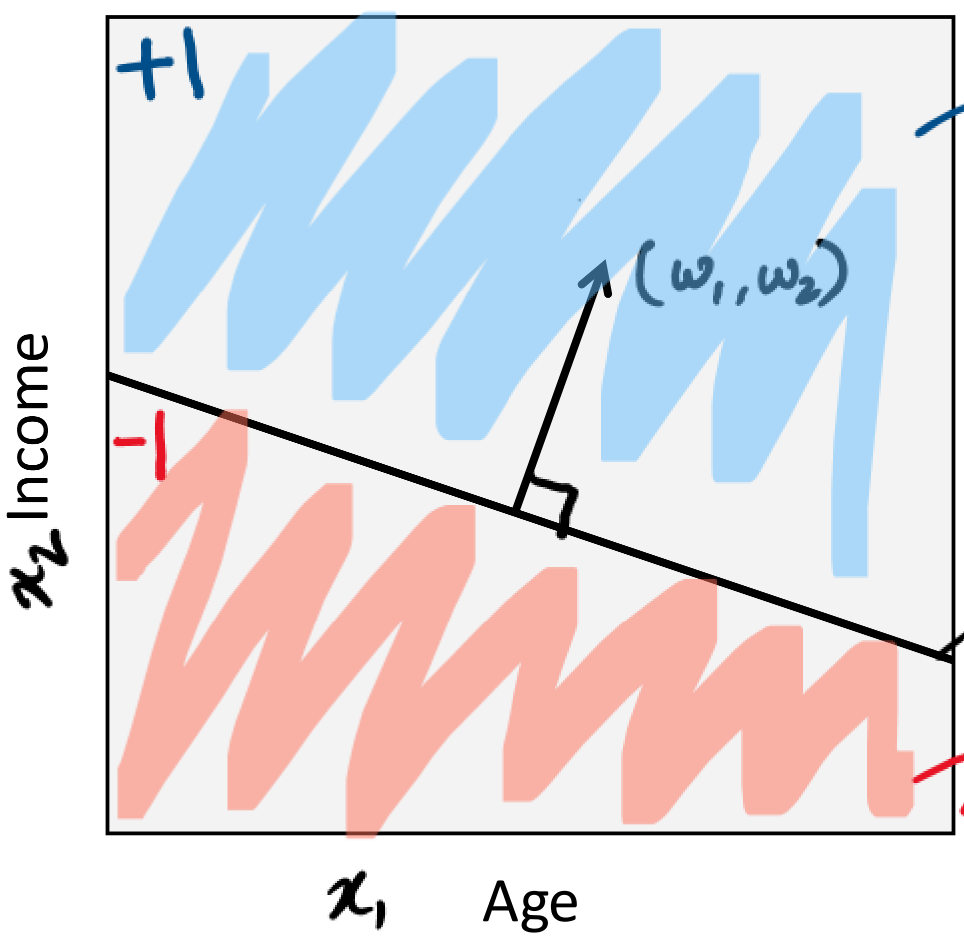
The Perceptron Hypothesis Set

- We define a hypothesis set $\mathcal{H}$

$$\mathcal{H} = \{h(x) = sign(\boldsymbol{w}^T \boldsymbol{x})\}$$

The ***perceptron*** or ***linear separator***

Geometry of The Perceptron in $\mathbb{R}^2$



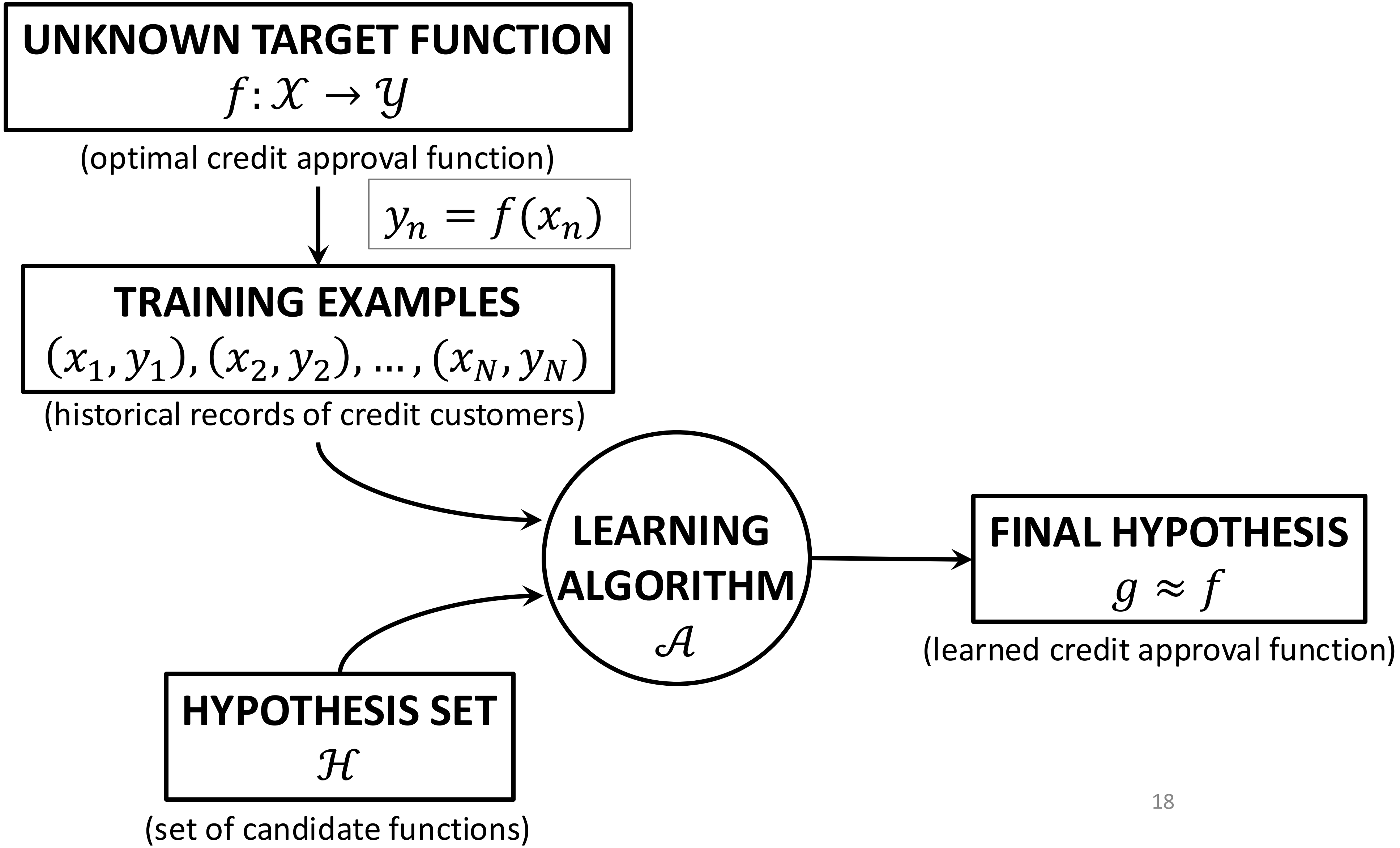
$w_1x_1 + w_2x_2 + w_0 > 0$

A line separates **+1** from **-1**

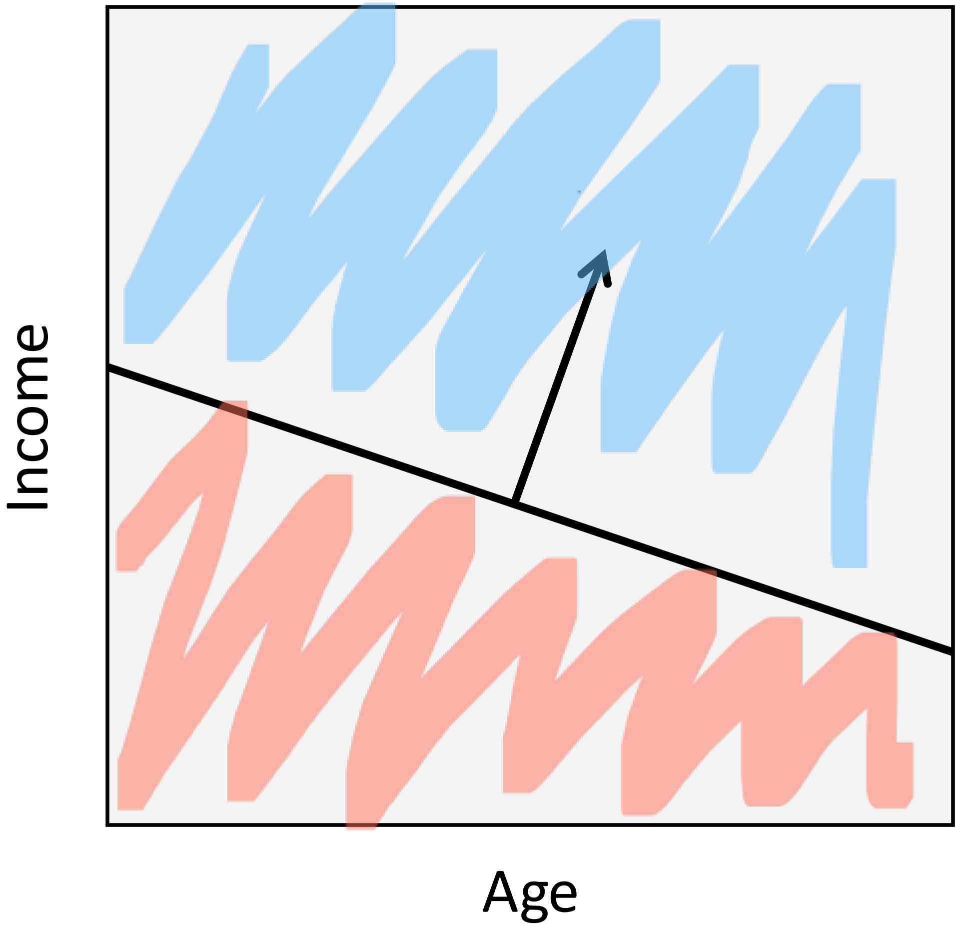
$w_1x_1 + w_2x_2 + w_0 = 0$

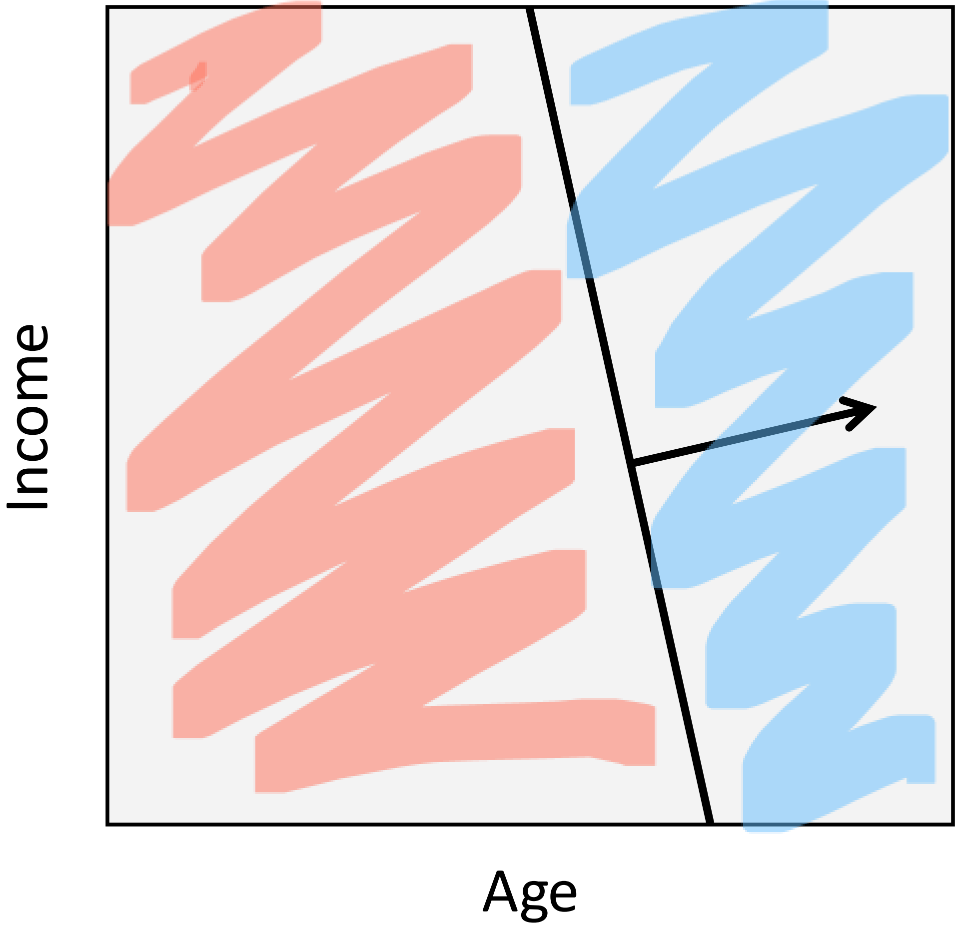
$w_1x_1 + w_2x_2 + w_0 < 0$

Learning Problem Setup



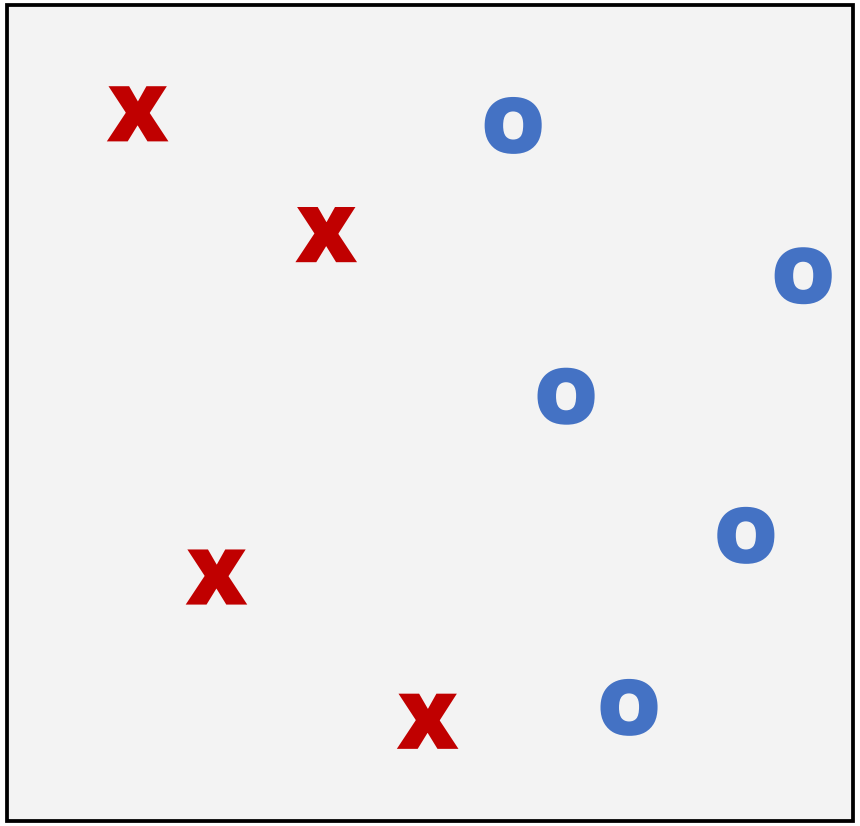
Geometry of The Perceptron in $\mathbb{R}^2$



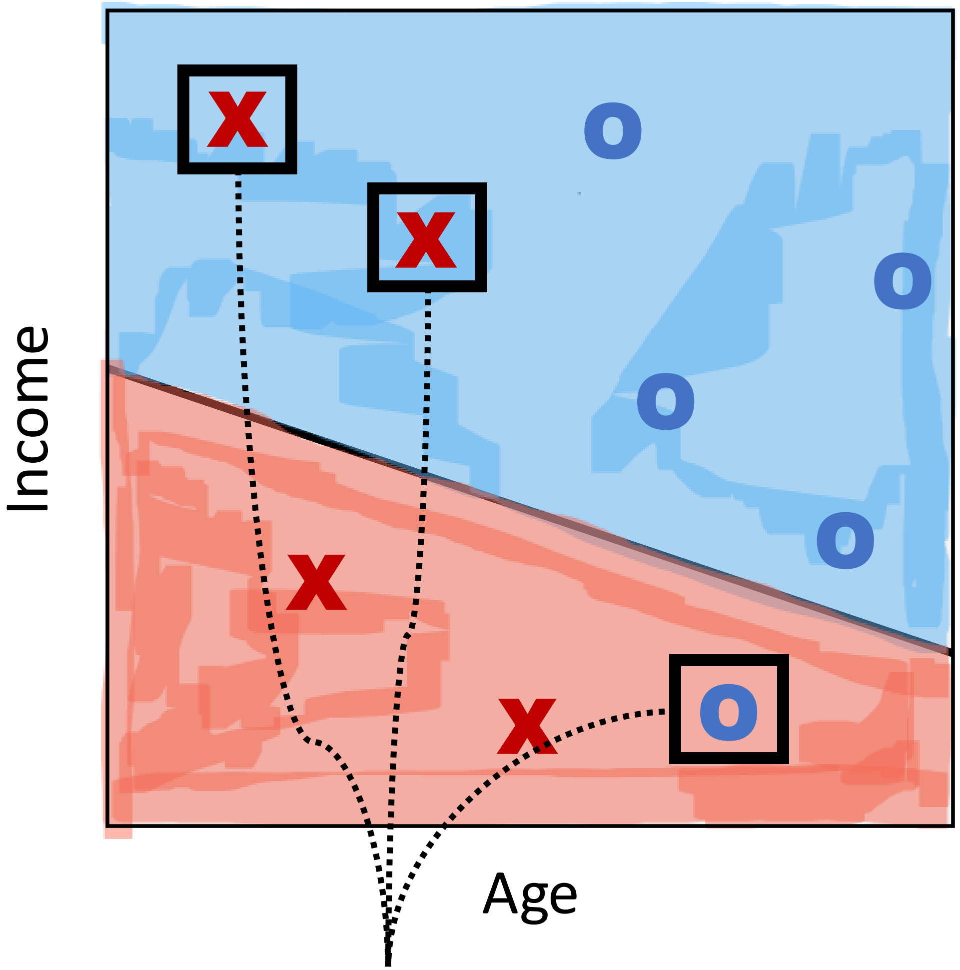


Which one to pick?

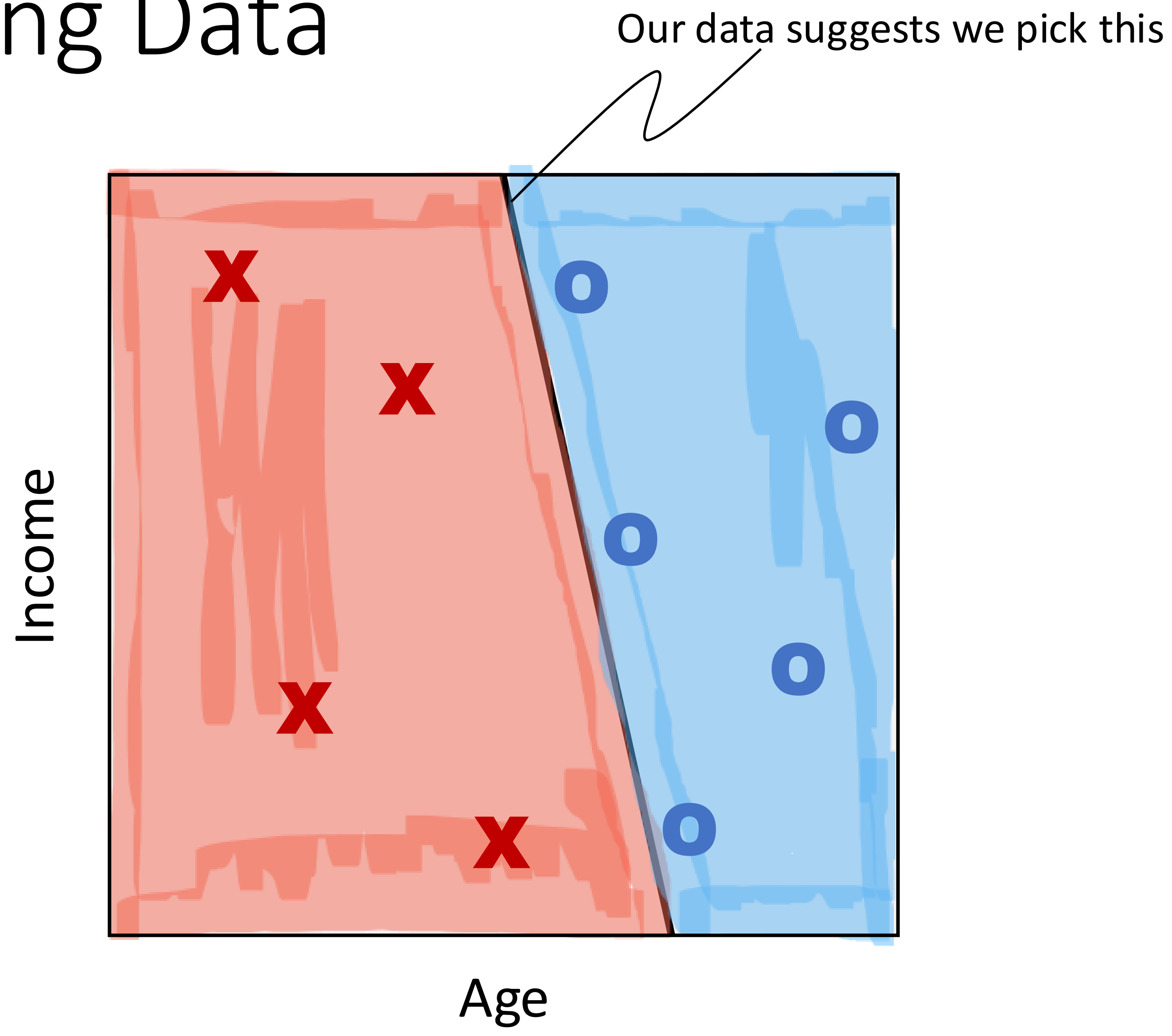
Using Data



Select a Hypothesis Using Data



misclassified data



perfectly classified

How to Learn a Final Hypothesis g from $\mathcal{H}$?

- Want: Select g from $\mathcal{H}$ so that $g \approx f$
- Certainly want $g \approx f$ on the dataset $\mathcal{D}$, i.e.,

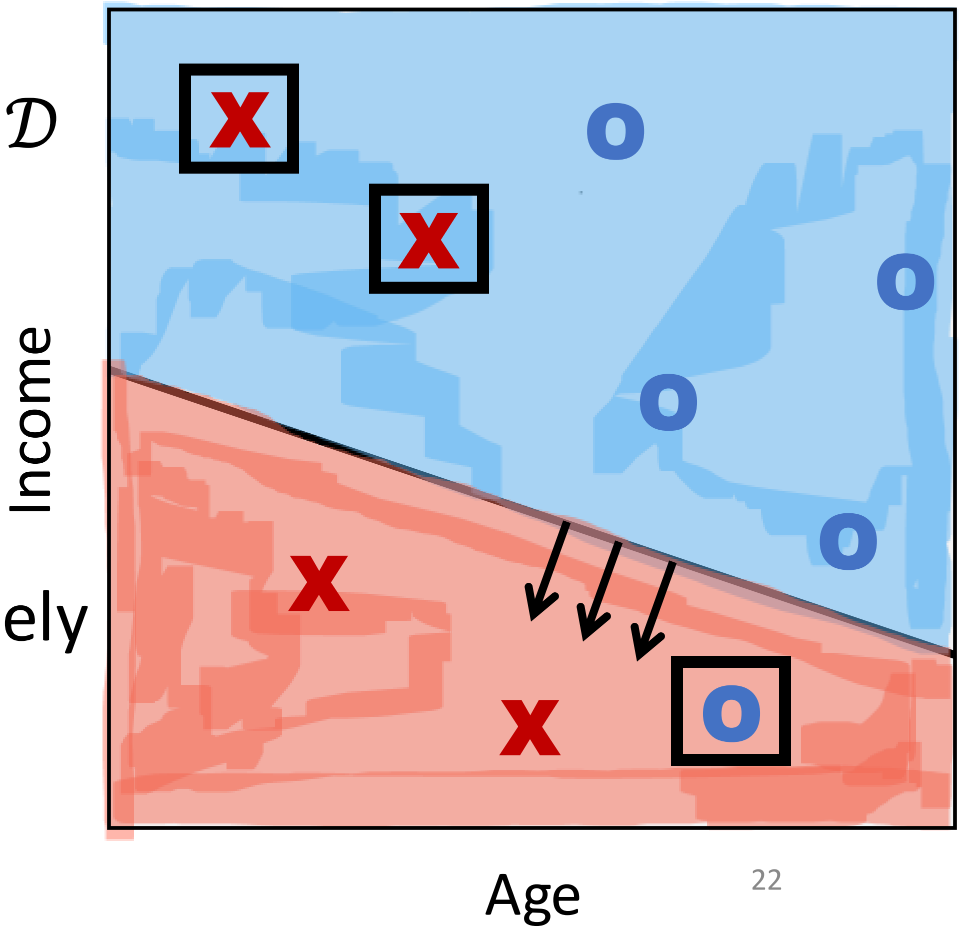
$$g(\boldsymbol{x}_n) = y_n \text{ for each } (x_n, y_n) \text{ in } \mathcal{D}$$

- But $\mathcal{H}$ is uncountably infinite

How to find g in the infinite hypothesis set $\mathcal{H}$?

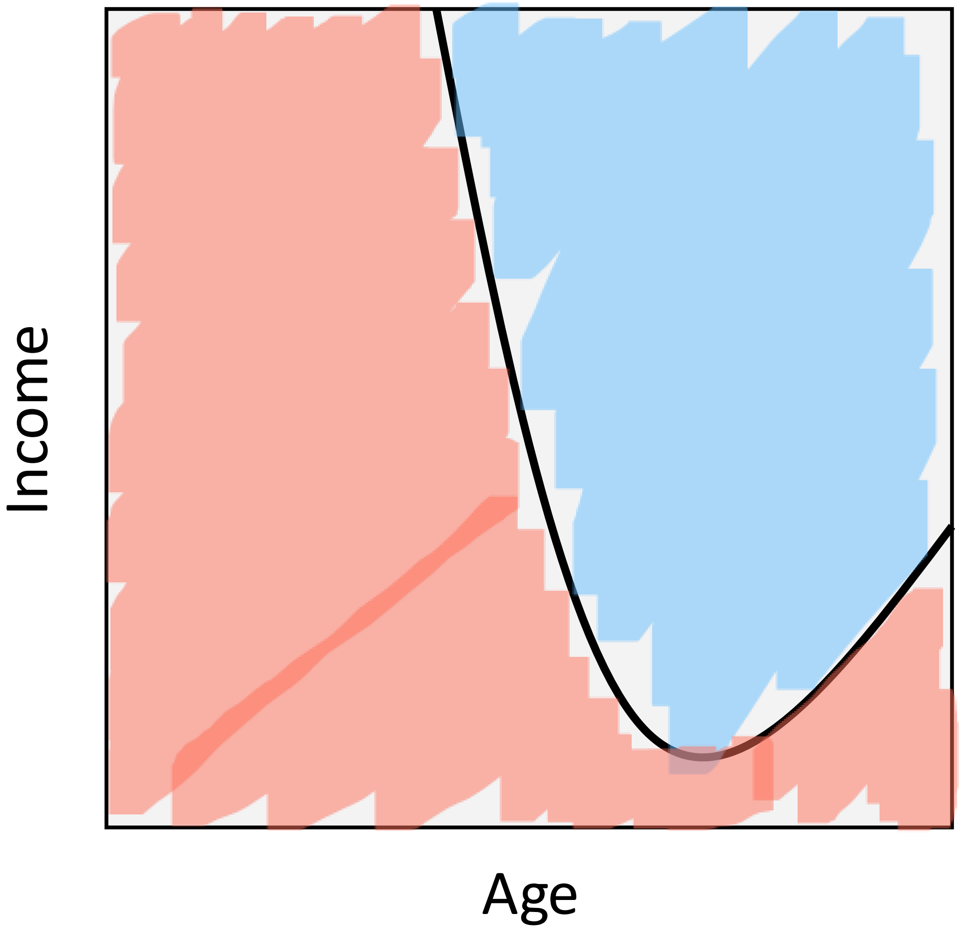
Start with *some* weights and improve it iteratively

(Coming Soon: The Perceptron Learning Algorithm)

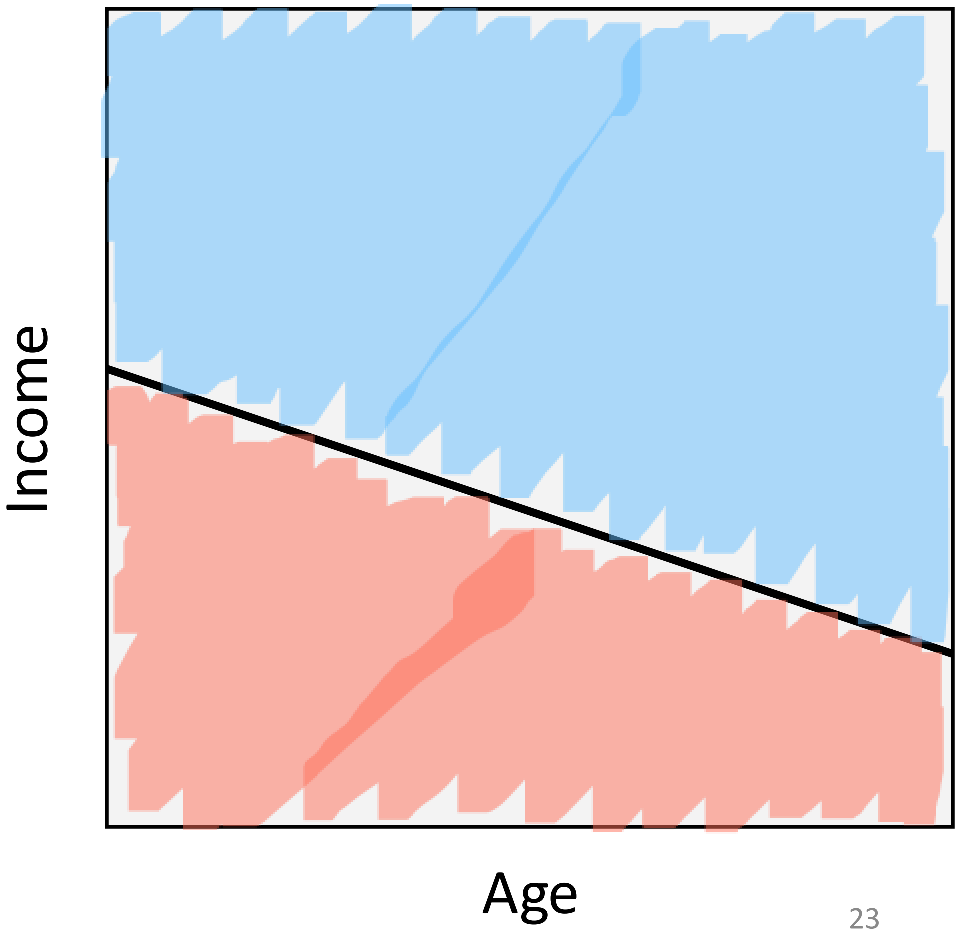


How Does the Bin Model Relate to Learning?

Unknown target function f



A known fixed hypothesis h



How Does the Bin Model Relate to Learning?

- Unknown, fixed target function $f: \mathcal{X} \rightarrow \mathcal{Y}$
- For any $h \in \mathcal{H}$: keep it fixed
  - Suppose we compare $h(\boldsymbol{x})$ to $f(\boldsymbol{x})$ on each point $\boldsymbol{x} \in \mathcal{X}$

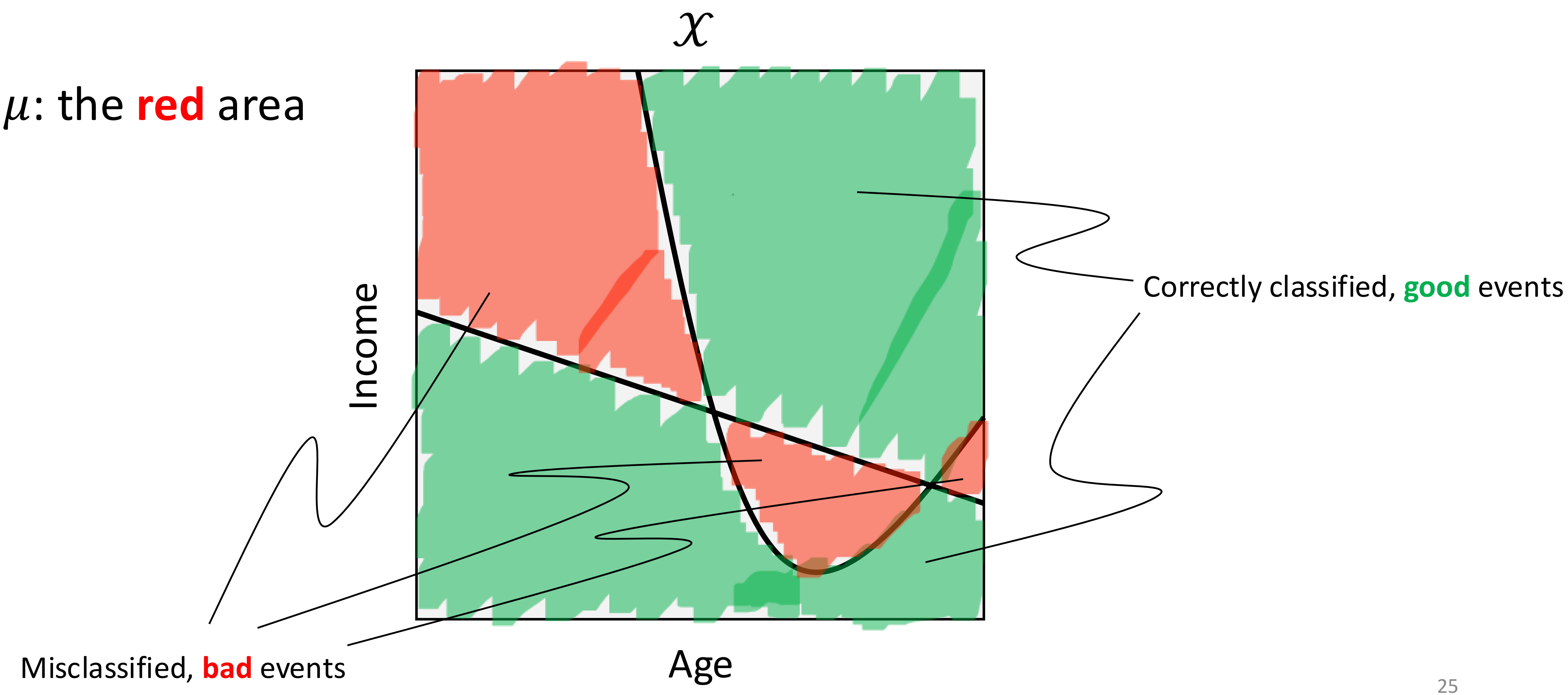
  - If $h(\boldsymbol{x}) = f(\boldsymbol{x})$, color $\boldsymbol{x}$ **green**
  - Otherwise, if $h(\boldsymbol{x}) \neq f(\boldsymbol{x})$, color $\boldsymbol{x}$ **red**

  - μ: the fraction of all possible data points that are **red**

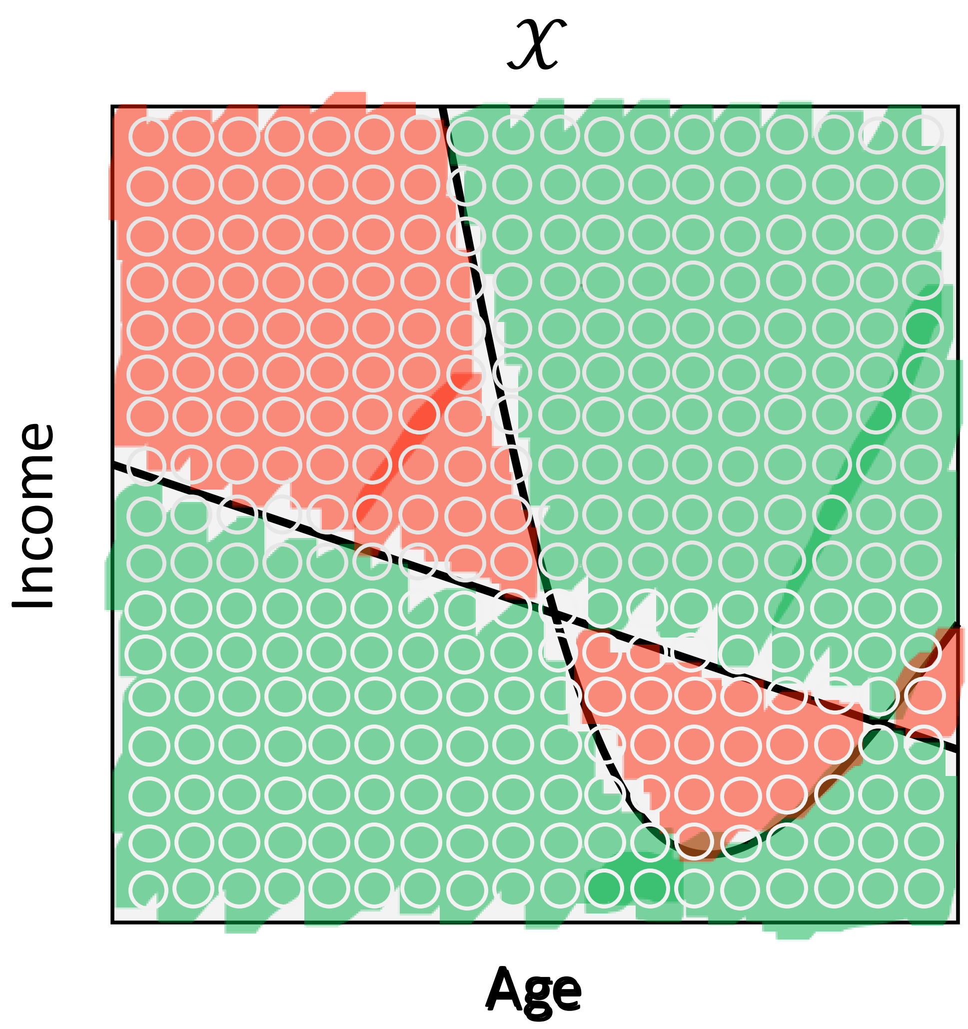
  This is the out-of-sample error of h

How Does the Bin Model Relate to Learning?

μ: the **red** area



The Error Function



Green: $h(\boldsymbol{x}) = f(\boldsymbol{x})$

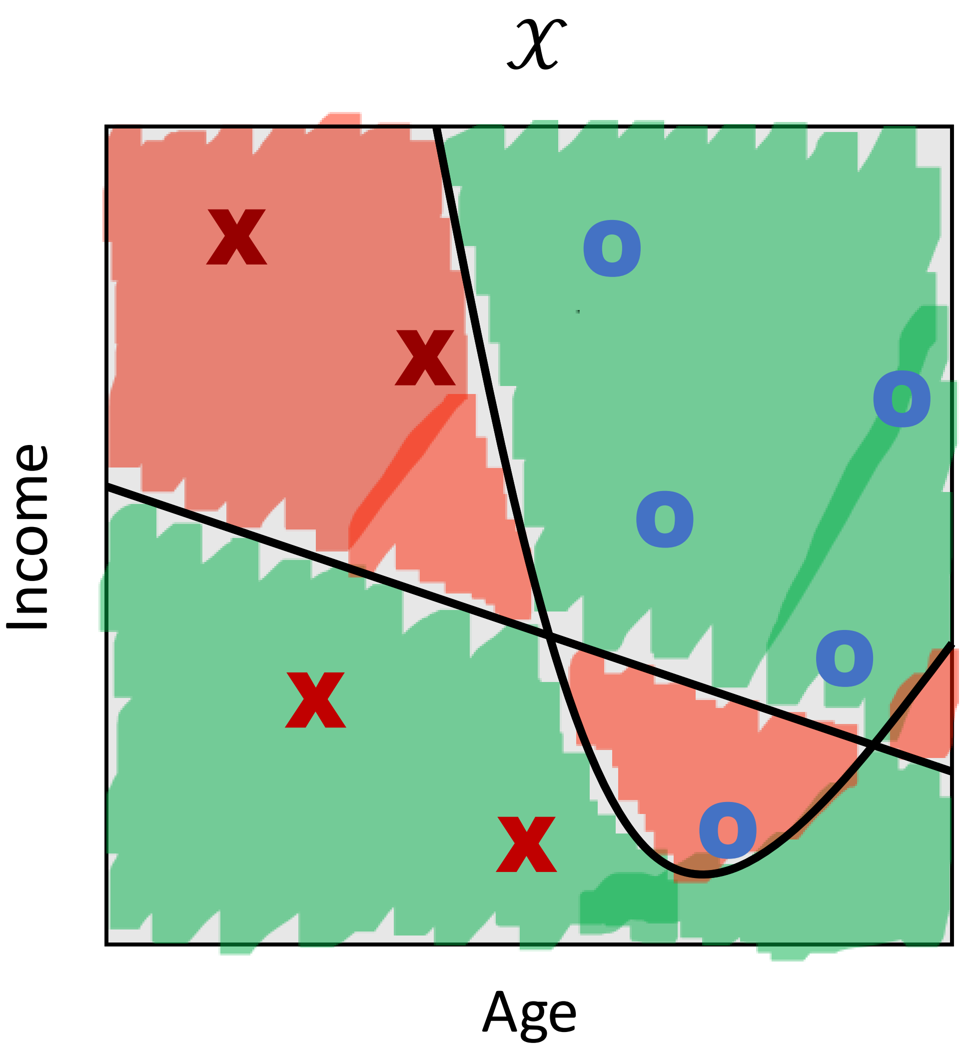
Red: $h(\boldsymbol{x}) \neq f(\boldsymbol{x})$

$$E_{out}(h) = \mathbb{P}_{\boldsymbol{x}}[h(\boldsymbol{x}) \neq f(\boldsymbol{x})]$$

(size of red region)

But this is UNKNOWN

The Error Function



Green: $h(\boldsymbol{x}) = f(\boldsymbol{x})$

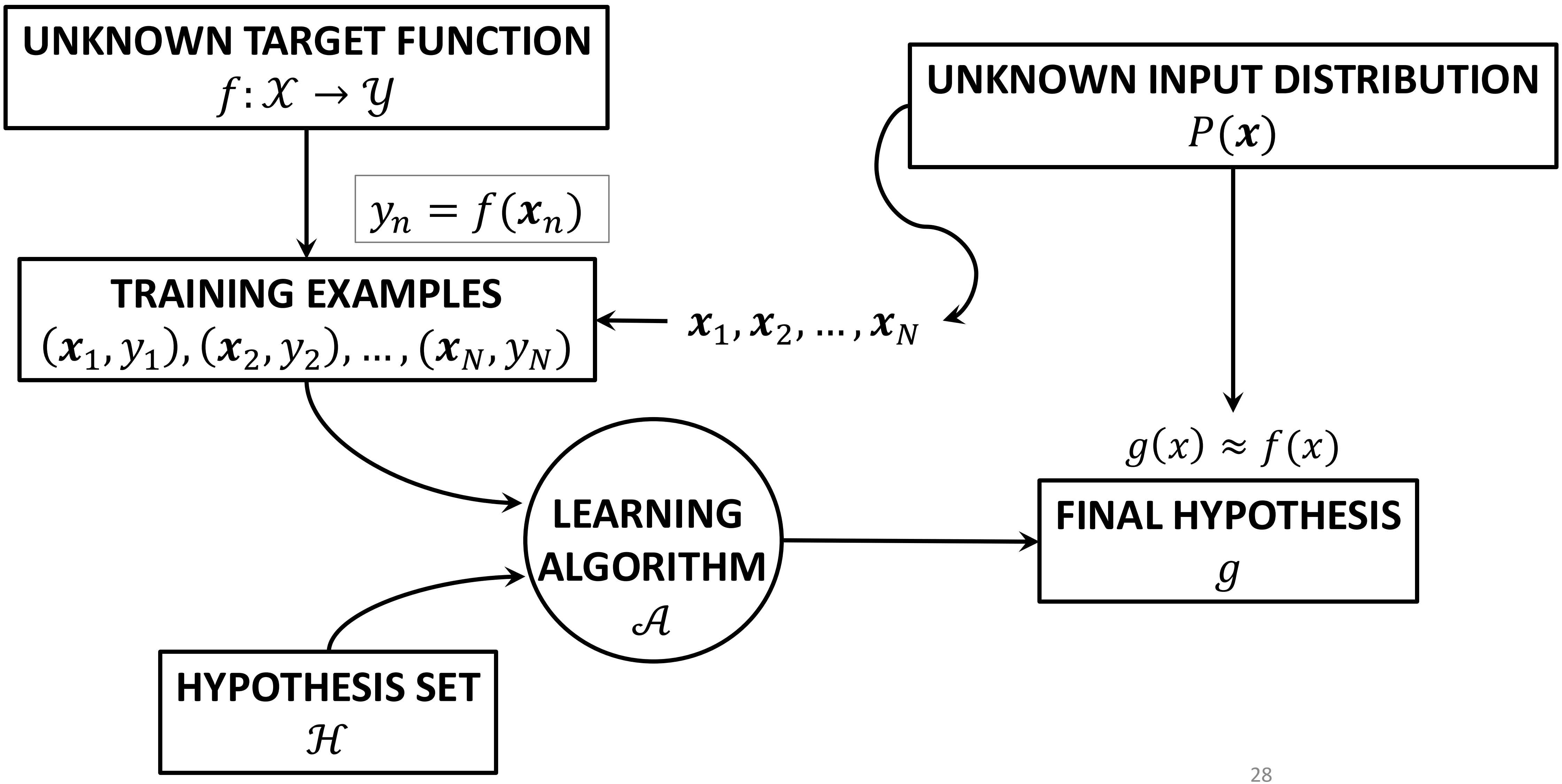
Red: $h(\boldsymbol{x}) \neq f(\boldsymbol{x})$

$E_{in}(h)$ = fraction of sampled data points in **red** region
i.e. misclassified data points

$$E_{in}(h) = \frac{1}{N}\sum_{n=1}^{N} [\![h(\boldsymbol{x}_n) \neq f(\boldsymbol{x}_n)]\!]$$

We know this

Learning Problem Setup with Probability



How does the Bin Model Relate to Learning?

Learning

- input space $\mathcal{X}$
- $\boldsymbol{x}$ for which $h(\boldsymbol{x}) = f(\boldsymbol{x})$
- $\boldsymbol{x}$ for which $h(\boldsymbol{x}) \neq f(\boldsymbol{x})$

- sample according to $P(\boldsymbol{x})$
- data set $\mathcal{D}$ of size N

- $E_{out}(h) = \mathbb{P}_x[h(x) \neq f(x)]$
- $E_{in}(h) = \frac{1}{N}\sum_{n=1}^{N}[\![h(\boldsymbol{x}_n) \neq f(\boldsymbol{x}_n)]\!]$

Bin

- Bin
- green marble
- red marble

- randomly pick a marble
- sample of N marbles

- μ = probability of picking red
- ν = fraction of red observed

Hoeffding's Inequality for Learning

For a ***fixed*** hypothesis h

$$\boxed{\mathbb{P}[|E_{in}(h) - E_{out}(h)| > \epsilon] \leq 2e^{-2\epsilon^2 N}, \text{ for any } \epsilon > 0}$$

- If $E_{in} \approx 0$ then $E_{out} \approx 0$ i.e. $\mathbb{P}_{\boldsymbol{x}}[h(\boldsymbol{x}) \neq f(\boldsymbol{x})]$ with high probability i.e. $f \approx h$ over all of $\mathcal{X}$

Now: Given h, we can **verify** whether it is "good"

Hoeffding's Inequality for ~~Learning~~ Verification

For a ***fixed*** hypothesis h

$$\boxed{\mathbb{P}[|E_{in}(h) - E_{out}(h)| > \epsilon] \leq 2e^{-2\epsilon^2 N}, \text{ for any } \epsilon > 0}$$

- If $E_{in} \approx 0$ then $E_{out} \approx 0$ i.e. $\mathbb{P}_{\boldsymbol{x}}[h(\boldsymbol{x}) \neq f(\boldsymbol{x})]$ with high probability i.e. $f \approx h$ over all of $\mathcal{X}$

Now: Given h, we can **verify** whether it is "good"