LEARNING PROBLEM

Bùi Tiến Lên

2023

KHOA CÔNG NGHỆ THÔNG TIN
TRƯỜNG ĐẠI HỌC KHOA HỌC TỰ NHIÊN

Contents

Notation

symbol	meaning
$a, b, c, N \dots$	scalar number
$\boldsymbol{w}, \boldsymbol{v}, \boldsymbol{x}, \boldsymbol{y} \dots$	column vector
$\boldsymbol{X}, \boldsymbol{Y} \dots$	matrix
$\mathbb{R}$	set of real numbers
$\mathbb{Z}$	set of integer numbers
$\mathbb{N}$	set of natural numbers
$\mathbb{R}^D$	set of vectors
$\mathcal{X}, \mathcal{Y}, \dots$	set
$\mathcal{A}$	algorithm

operator	meaning
$\boldsymbol{w}^\intercal$	transpose
$\boldsymbol{X}\boldsymbol{Y}$	matrix multiplication
$\boldsymbol{X}^{-1}$	inverse

Learning Components

Credit Approval

- Suppose that a bank receives thousands of credit card applications every day, and it wants to automate the process of evaluating them.
- Applicant information

age	23 years
gender	male
annual salary	$30000
years in residence	1 year
years in job	1 year
current debt	$15000
...	...

- Approve credit?

Problem Statement

Formalization

- Input: $\boldsymbol{x}$ (*customer application*)
- Output: y (*good/bad customer?* or $\{1, -1\}$)
- Data $(\boldsymbol{x}_1, y_1), (\boldsymbol{x}_2, y_2), \ldots(\boldsymbol{x}_N, y_N)$ (*historical records*)
- Target function: $f : \mathcal{X} \to \mathcal{Y}$ (*ideal credit approval formula*)
- Best approximate function $g : \mathcal{X} \to \mathcal{Y}$ (*formula to be used*)

Inductive Bias

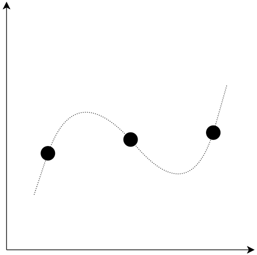
Theorem 1 (No Free Lunch Theorems)

An unbiased learner can never generalize.

Concept 1

An inductive bias of a learner is the set of assumptions a learner uses to predict results given inputs it has not yet encountered.

- **Consider**: arbitrarily wiggly functions or random truth tables.



x_1	x_2	x_3	y
0	0	0	0
0	0	1	?
0	1	0	1
0	1	1	1
1	0	0	0
1	0	1	?
1	1	0	1
1	1	1	?

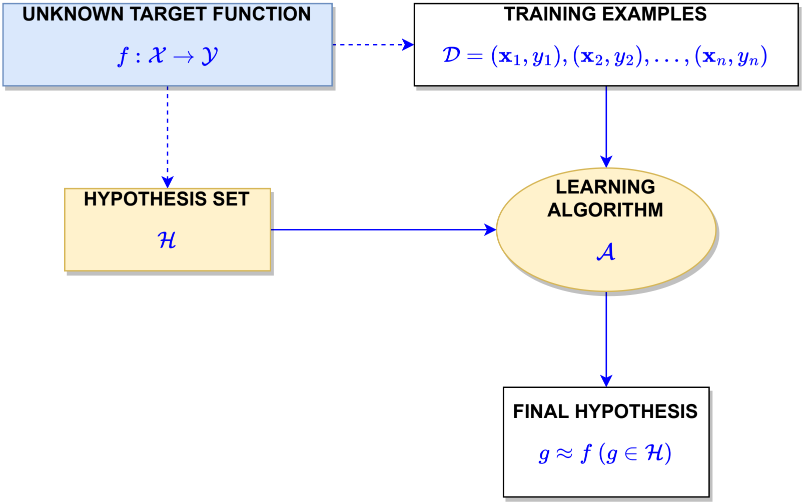
Inductive Bias (cont.)

Inductive Learning Hypothesis

Generalization is possible.

- If a machine performs well on most **training data** AND it is not too complex, it will probably do well on **similar test data**.

Components of Learning

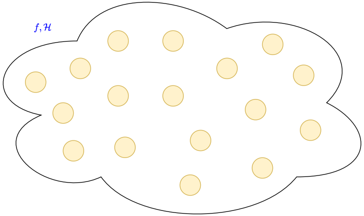


Learning Model

The two components are referred as the **learning model**

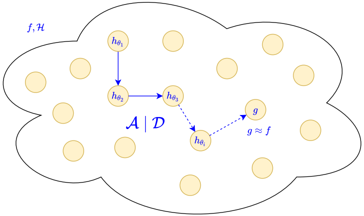
- The **hypothesis set** $\mathcal{H}$ is a set of functions that is potentially similar to f

$$\mathcal{H} = \{h_{\theta_1}, h_{\theta_2}, \ldots\}$$



- The **learning algorithm** $\mathcal{A}$ is a **search algorithm** which finds $g \in \mathcal{H}$ such that

$$g \overset{best}{\approx} f$$



What is hypothesis set

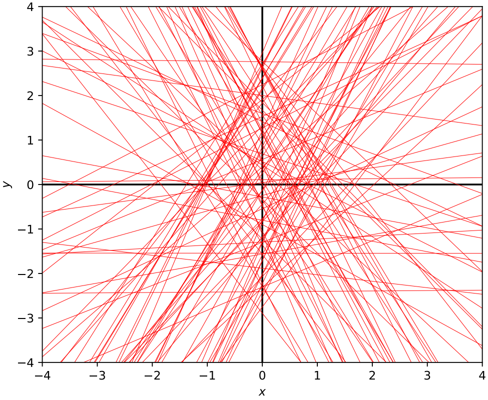
Concept 2

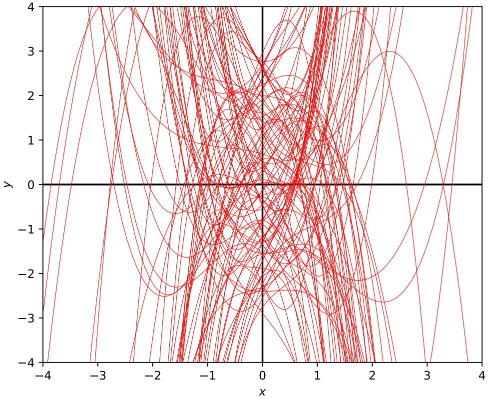
Hypothesis set is a set of potential functions, models or solutions

- Hypothesis set can be **finite**. For example
  - $\{guilty, \ not \ guilty\}$
  - $\{\text{accept}, \ reject\}$
  - $\{\text{happy}, \ \text{sad}\}$
  - $\{1, 2, 3, 4, 5, 6\}$

What is hypothesis set (cont.)

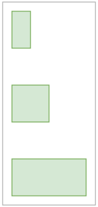
- Hypothesis set can be **infinite**. For example, sets of functions $y = \theta_0 + \theta_1 x$ and $y = \theta_0 + \theta_1 x + \theta_2 x^2 + \theta_3 x^3$

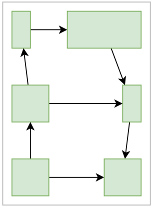




Parameter representations

- Each element of hypothesis set often indexed by **parameters** or **weights** ($\boldsymbol{\theta}$ or $\boldsymbol{w}$)
- Two basic representations for parameters: **factored**, and **structured**
  1. Factored: a paramater set consists of a vector of attribute values; values can be boolean, real-valued, or one of a fixed set of symbols.
  2. Structured: a paramater set includes objects, each of which may have attributes of its own as well as relationships to other objects.





A Simple Learning Model

- Hypothesis Set
- Learning Algorithm

A Simple Hypothesis Set

We starts with the simple model (**the perceptron model**)

- For input $\boldsymbol{x} = (x_1, \ldots, x_d)$ (*attributes of a customer*)

$$\begin{aligned} \text{Approve credit if } & \sum_{i=1}^{d} w_i x_i \geq \textit{threshold} \\ \text{Deny credit if } & \sum_{i=1}^{d} w_i x_i < \textit{threshold} \end{aligned} \quad (1)$$

- This linear formula $h \in \mathcal{H}$ can be written as

$$h(\boldsymbol{x}) = h_{\boldsymbol{w},threshold}(\boldsymbol{x}) = \text{sign}\left(\left(\sum_{i=1}^{d} w_i x_i\right) - \textit{threshold}\right) \quad (2)$$

A Simple Hypothesis Set (cont.)

- Set $w_0 = -threshold$

$$h(x) = h_{\boldsymbol{w}}(x) = \text{sign}\left(\left(\sum_{i=1}^{d} w_i x_i\right) + w_0\right) \tag{3}$$

- Introduce an artificial coordinate $x_0 = 1$

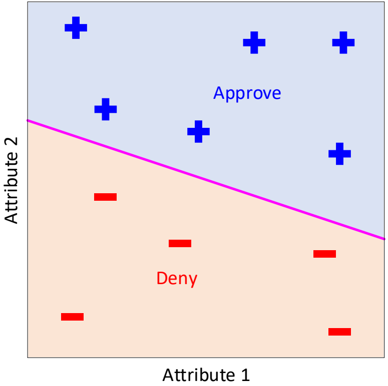
$$h(x) = h_{\boldsymbol{w}}(x) = \text{sign}\left(\sum_{i=0}^{d} w_i x_i\right) \tag{4}$$

- In vector form, the perceptron implements

$$h(x) = h_{\boldsymbol{w}}(x) = \text{sign}\left(\boldsymbol{w}^{\intercal}\boldsymbol{x}\right) \tag{5}$$

2D Model Visualization

- **Decision boundaries**: line
- **Decision regions**: approve and deny regions



A Simple Learning Algorithm

- **The performance measure**: *the error rate*
- We uses the simple learning algorithm (**perceptron learning algorithm - PLA**) to find $\boldsymbol{w}$

$$\arg\min_{\boldsymbol{w}} E(h_{\boldsymbol{w}}(x), y \mid \mathcal{D}) \quad (6)$$

A Simple Learning Algorithm (cont.)

- Given the training set

$$\mathcal{D} = \{(\boldsymbol{x}_1, y_1), (\boldsymbol{x}_2, y_2), \ldots (\boldsymbol{x}_N, y_N)\}$$

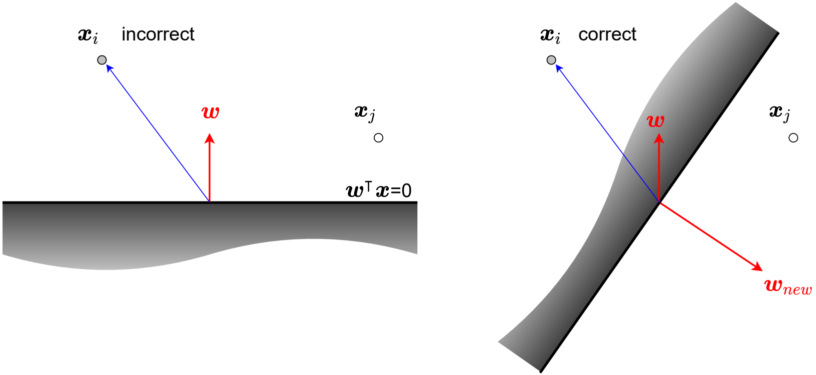
1. **Init $\boldsymbol{w}$**
2. **Repeat until satisfied**
   - At iteration $t = 1, 2, 3, \ldots,$ pick a *misclassified* point $(\boldsymbol{x}_i, y_i)$

$$sign(\boldsymbol{w}^\intercal \boldsymbol{x}_i) \neq y_i \tag{7}$$

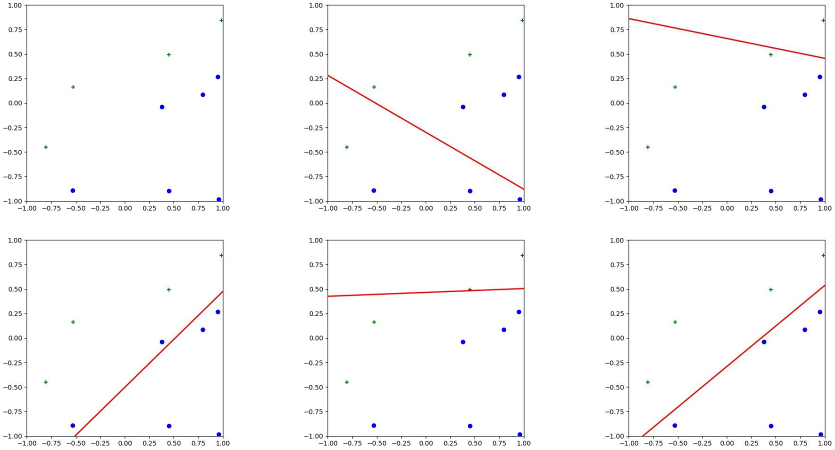
   - and update the weight vector

$$\boldsymbol{w} \leftarrow \boldsymbol{w} + y_i \boldsymbol{x}_i \tag{8}$$

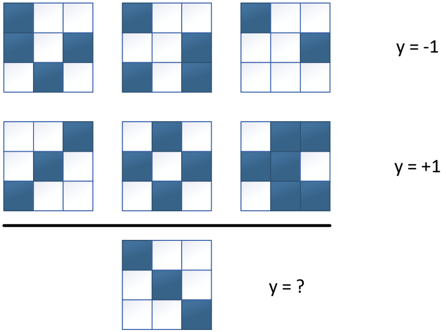
A Simple Explanation



Is It Learning Algorithm?



A Learning Puzzle



Feasibility Of Learning

- Probability to the rescue

Feasibility Of Learning

The **feasibility of learning** is thus split into two questions:

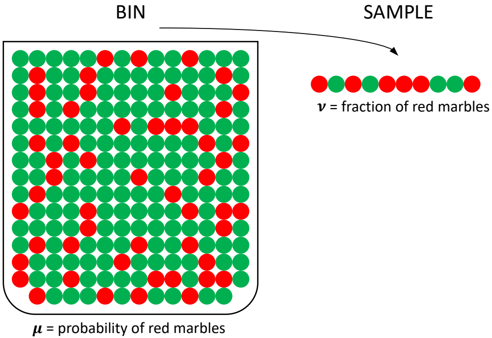
1. Can we make **the performance** good enough?
   - run our learning algorithm on the actual data $\mathcal{D}$ and see how good we can get.
2. Can we make sure that **the performance** inside of $\mathcal{D}$ is close enough to **the performance** outside of $\mathcal{D}$?
   - probability theory

A Related Experiment - Bin Problem

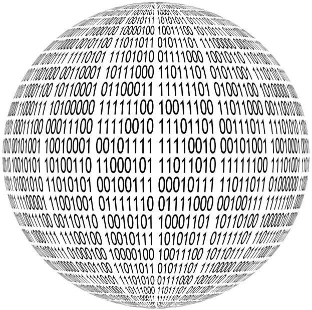
- Consider a **BIN** with red and green marbles

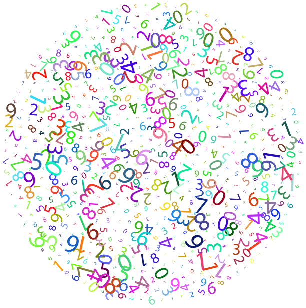
$$P[\text{picking a red marble}] = \mu$$
$$P[\text{picking a green marble}] = 1 - \mu$$

- The value of μ is unknown to us
- We pick N marbles independently
- The fraction of red marbles in **SAMPLE** $= \nu$



Does ν say anything about μ?





- **No!** (certain answer): Sample can be mostly red while bin is mostly red
- **Yes!** (uncertain answer): Sample frequency ν is likely close to bin frequency μ

What does ν say about μ?

- In a big sample (large N), ν is probably close μ (within ϵ)
- Formally,

$$P[|\nu - \mu| > \epsilon] \leq 2e^{-2\epsilon^2 N} \texttt{ for any } \epsilon > 0 \tag{9}$$

  This is called **Hoeffding's Inequality**
- **Bound** does not depend on μ; tradeoff: N, ϵ and the bound
- We have

$$\nu \approx \mu \Longrightarrow \mu \approx \nu$$

- In other words, the statement "$\mu = \nu$" is **probably approximately correct** (P.A.C)

Connection to Learning

Bin problem	Learning problem
The unknown is a number μ	The unknown is a function $f : \mathcal{X} \to \mathcal{Y}$
a marble $\bullet$	a point $\boldsymbol{x} \in \mathcal{X}$
$\color{green}\bullet$	hypothesis got it right $h(x) = f(x)$
$\color{red}\bullet$	hypothesis got it wrong $h(x) \neq f(x)$

Connection to Learning (cont.)

- The *error rate* within the sample $\mathcal{D}$, which corresponds to ν in the bin model, will be called the *in-sample error*

$$E_{in}(h) = \text{fraction of } \mathcal{D} \text{ where } f \text{ and } h \text{ disagree}$$
$$= \frac{1}{N} \sum_{n=1}^{N} \mathbb{I}(h(\boldsymbol{x}_n) \neq f(\boldsymbol{x}_n))$$

where $\mathbb{I}(\ldots) = 1$ if the statement is **true**, and $\mathbb{I}(\ldots) = 0$ if the statement is **false**

- In the same way, we define the *out-of-sample error* , (domain $\mathcal{X}$)

$$E_{out}(h) = P(h(\boldsymbol{x}) \neq f(\boldsymbol{x})), \boldsymbol{x} \in \mathcal{X}$$

which corresponds to μ in the bin model.

Connection to Learning (cont.)

- The Hoeffding inequality becomes:

$$P[|E_{in}(h) - E_{out}(h)| > \epsilon] \leq 2e^{-2\epsilon^2 N} \text{ for any } \epsilon > 0 \tag{10}$$

*In a big sample $\mathcal{D}$, **the performance** inside of $\mathcal{D}$ is close enough to **the performance** outside of $\mathcal{D}$*

Risk and Emprical Risk

- Loss function
- Empirical risk
- Regularizer

Loss function

Concept 3

Given a hypothesis $\hat{y} = h(\boldsymbol{x}) \in \mathcal{H}$, a non-negative real-valued **loss function** $\ell(\hat{y}, y)$ which measures how different the prediction $\hat{y}$ of a hypothesis is from the true outcome y.

Loss Functions for Binary Classification

- Zero-one loss

$$\mathbb{I}(h(\boldsymbol{x}) \neq y) \tag{11}$$

- Log loss (logistic regression)

$$\log(1 + e^{-h(\boldsymbol{x})y}) \tag{12}$$

- Exponential loss (AdaBoost)

$$e^{-h(\boldsymbol{x})y} \tag{13}$$

Loss Functions for Regression

- Squared loss

$$(h(\boldsymbol{x}) - y)^2 \tag{14}$$

- Absolute loss

$$|h(\boldsymbol{x}) - y| \tag{15}$$

Risk

Concept 4

The **risk** E associated with hypothesis $h(\boldsymbol{x})$ is defined as the expectation of the loss function

$$E(h) = \mathbb{E}[\ell(h(\boldsymbol{x}), y)] = \int \ell(h(\boldsymbol{x}), y) dp(\boldsymbol{x}, y) \tag{16}$$

Empirical Risk

Concept 5

The **empirical risk** $\hat{E}$ is the average of the loss function on the training set $\mathcal{D} = \{(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), ...(\mathbf{x}_N, y_N)\}$

$$\hat{E} = \frac{1}{N} \sum_{i=1}^{N} \ell(h_{\mathbf{w}}(\mathbf{x}_i), y_i) \tag{17}$$

Theorem 2

The empirial risk is unbiased estimate of the risk

Empirical Risk (cont.)

Concept 6

Empirical risk of hypothesis $h_{\mathbf{w}}(x)$ with a loss function ℓ and a regularizer *reg*

$$\hat{E} = \frac{1}{N}\sum_{i=1}^{N} \underbrace{\ell(h_{\mathbf{w}}(\mathbf{x}_i), y_i)}_{Loss} + \underbrace{\lambda reg(\mathbf{w})}_{Regularizer} \quad (18)$$

The empirical risk minimization principle

Principle

The learning algorithm should choose a hypothesis $h_{\mathbf{w}}$ which minimizes the empirical risk

$$h_{\mathbf{w}} = \arg\min_{h_{\mathbf{w}} \in \mathcal{H}} \hat{E}(h_{\mathbf{w}} \mid \mathcal{D}) \tag{19}$$

Regularizers

Theorem 3

For each $\lambda \geq 0$, *there exists* $B \geq 0$. *such that the two formulations are equivalent,*

$$\arg\min_{\boldsymbol{w}} \sum_{i=1}^{N} \ell(h_{\boldsymbol{w}}(\boldsymbol{x}_i), y_i) + \lambda reg(\boldsymbol{w}) \tag{20}$$

$$\arg\min_{\boldsymbol{w}} \sum_{i=1}^{N} \ell(h_{\boldsymbol{w}}(\boldsymbol{x}_i), y_i) \text{ subject to } reg(\boldsymbol{w}) \leq B \tag{21}$$

Regularizers (cont.)

- L_2-regularization

$$reg(\mathbf{w}) = \mathbf{w}^\top \mathbf{w} = \|\mathbf{w}\|_2^2 \tag{22}$$

- L_1-regularization

$$reg(\mathbf{w}) = \|\mathbf{w}\|_1 \tag{23}$$

References

Goodfellow, I., Bengio, Y., and Courville, A. (2016).
Deep learning.
MIT press.

Lê, B. and Tô, V. (2014).
Cở sở trí tuệ nhân tạo.
Nhà xuất bản Khoa học và Kỹ thuật.

Russell, S. and Norvig, P. (2021).
Artificial intelligence: a modern approach.
Pearson Education Limited.