M2608.001300
Machine Learning Fundamentals & Applications

[1: The Learning Problem]

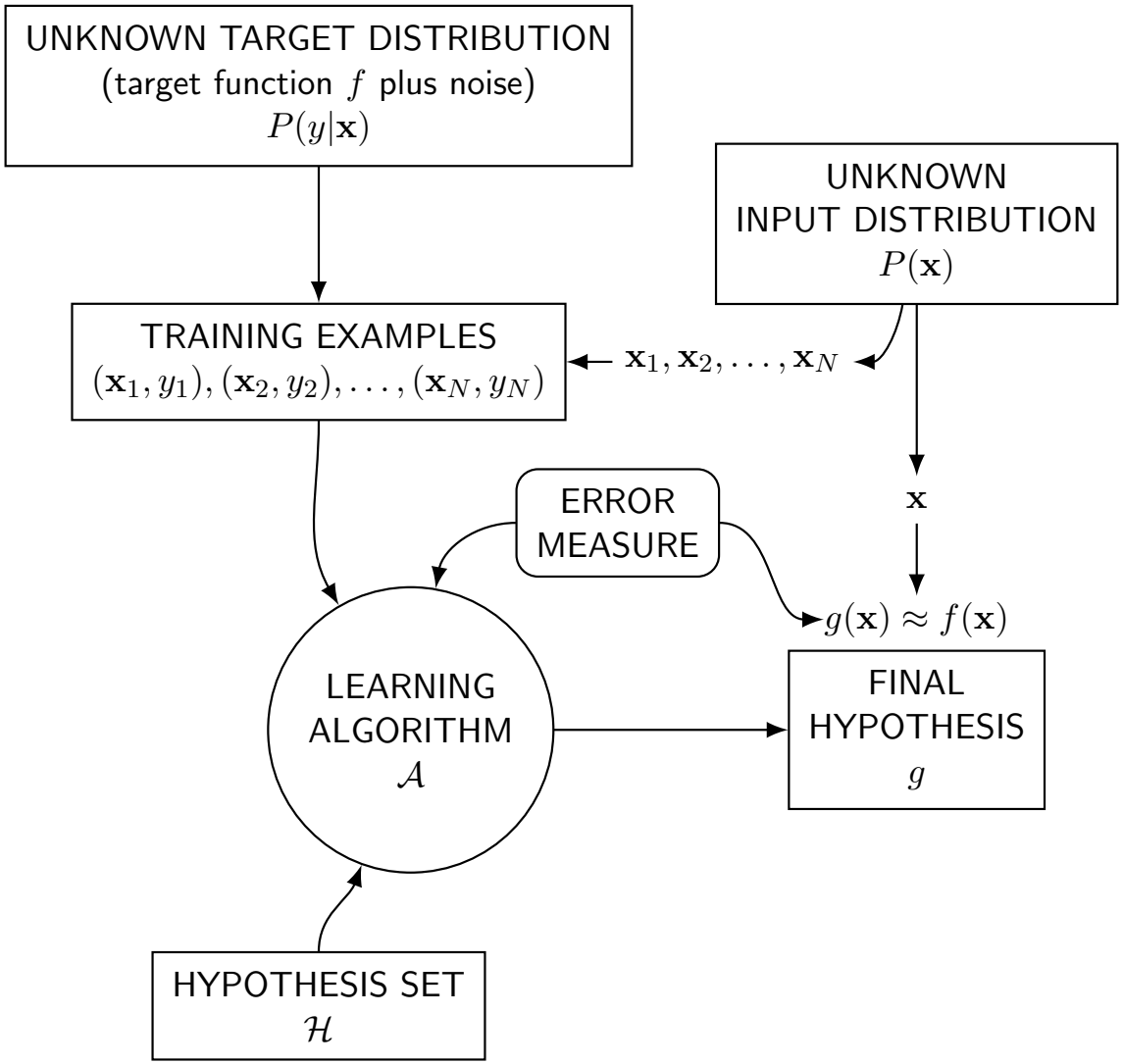
Electrical and Computer Engineering
Seoul National University

Outline

Readings

- *Learning from Data* by Abu-Mostafa, Magdon-Ismail, and Lin
  - Chapter 1: The Learning Problem (Sections 1.1 & 1.2)

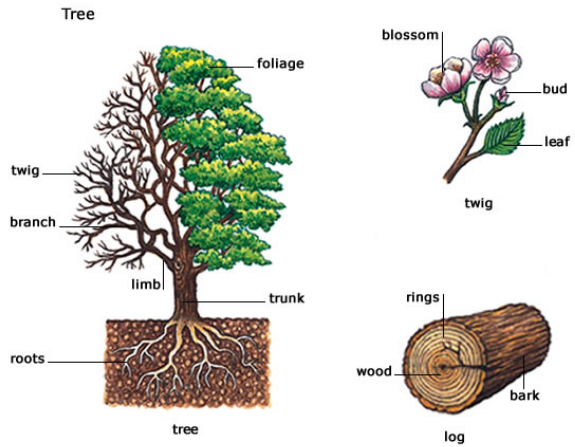
The big picture



Outline

What is a tree?

- show a picture to a three-year-old and ask if there is a tree
  - probably get the ~~correct~~ answer
- ask a thirty-year-old what the definition of a tree is
  - probably get an inconclusive (Incorrect) answer



Learning from data

- we learned what a tree is from 'data'
  - not by studying the mathematical definition of trees
  - but by looking at trees
- learning from data is used when
  - we have no analytic (exact, theoretical) solution
  - but have data to construct an empirical (휴리스틱) solution
- the premise covers a lot of territory

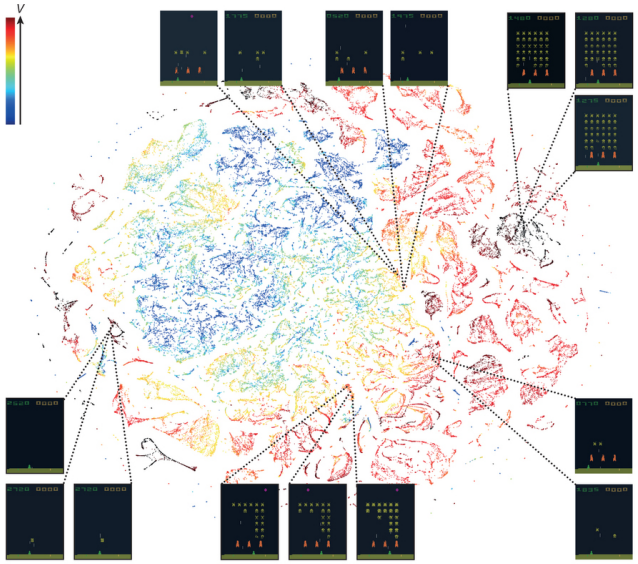
Example: predicting how a viewer will rate a movie

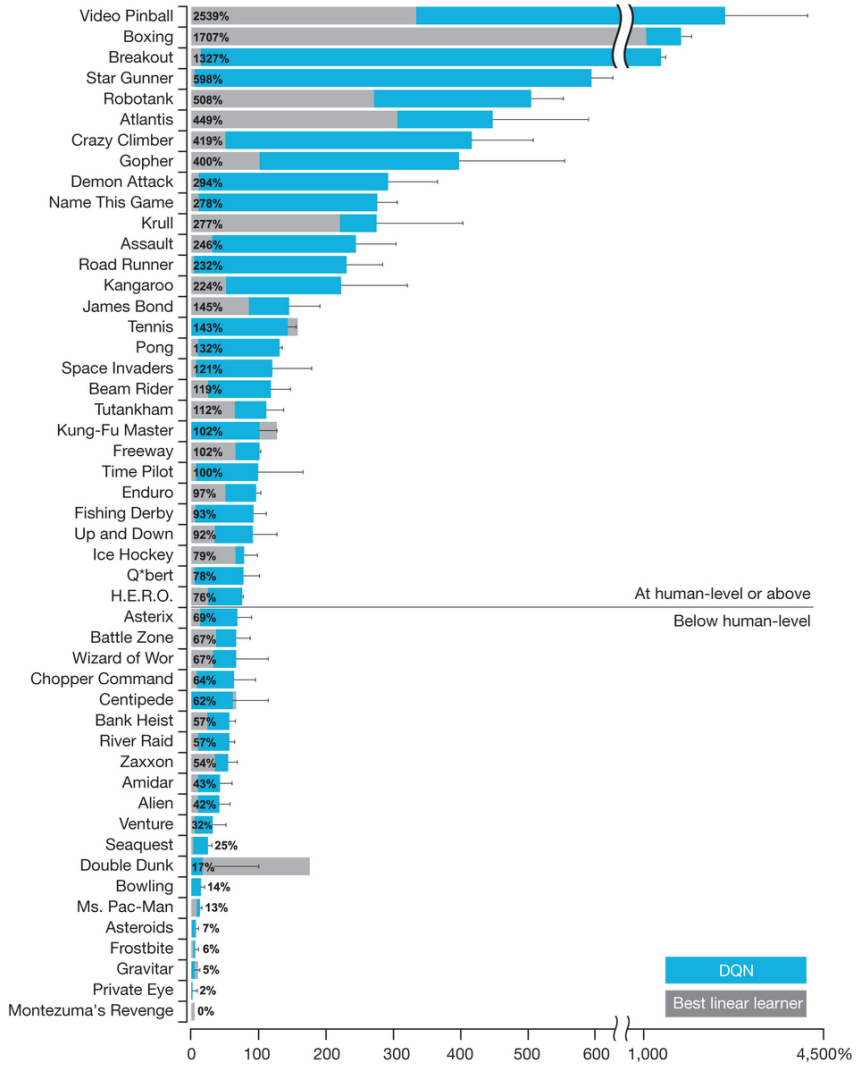
- good recommender systems are important
  - 20% sales are from recommendation (Amazon.com)
  - 10% improvement = 1 million dollar prize (Netflix)

- the main difficulty: an analytic solution is hard to get
  - the criteria viewers use to rate movies: quite complex
  - modeling those explicitly: not easy
- however, historical data reveal a lot about how people rate movies
- the power of learning from data
  - the entire process can be automated without any need for analyzing movie content or viewer taste

Example: human-level control (game play)

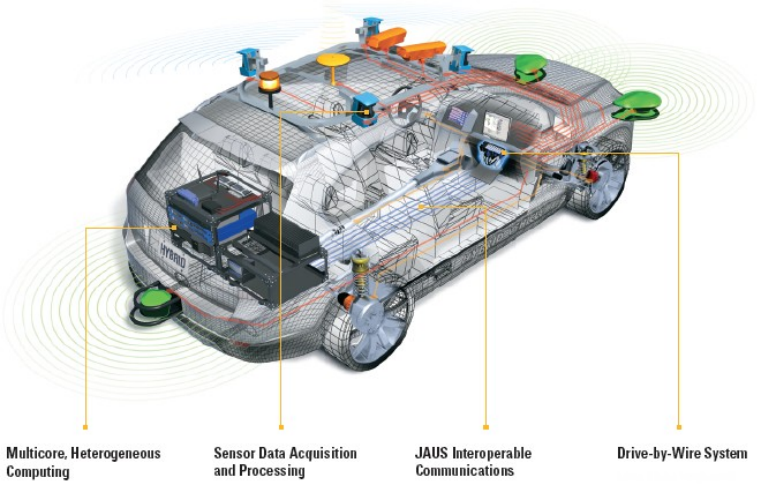
- deep reinforcement learning
  - by Google (2015)

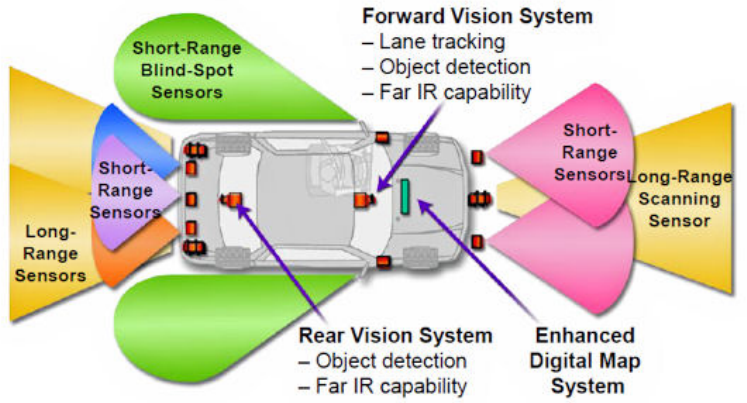





nature
At last — a computer program that can beat a champion Go player PAGE 484
ALL SYSTEMS GO
SONGBIRDS À LA CARTE
SAFEGUARD TRANSPARENCY
WHEN GENES GOT 'SELFISH'

Example: autonomous vehicles





The essence of learning from data

1. we have data
2. a pattern exists therein
3. we cannot pin it down mathmetically

Exercise

Which of the following problems are best suited for the learning approach?

(i) Classifying numbers into primes and non-primes.

(ii) Detecting potential fraud in credit card charges.

(iii) Determining the time it would take a falling object to hit the ground.

(iv) Determining the optimal cycle for traffic lights in a busy intersection.

Answer: ii) iv).

Outline

Running example: credit approval

- in order to abstract the common core of the learning problem
  - pick one application and use it
  - as a metaphor for the different components of the problem

- our metaphor: credit approval
  - applicant information $\longrightarrow$

- approve credit or not?

feature	value
age	23 years
gender	female
annual salary	$30,000
years in residence	1 year
years in job	1 year
current debt	$15,000
...	...

Components of learning: formalization

- let $\mathcal{X} = \mathbb{R}^d$ be the input space
  - $\mathbb{R}^d$: the d-dimensional Euclidean space
  - input vector $\mathbf{x} \in \mathcal{X}$: $\mathbf{x} = (x_1, x_2, \ldots, x_d)$
- let $\mathcal{Y} = \{+1, -1\}$ be the output space
  - denotes a binary (yes/no) decision
- in our credit example
  - coordinates of input $\mathbf{x}$:
    salary, debt, and other fields in a credit application
  - binary output y: approving or denying credit

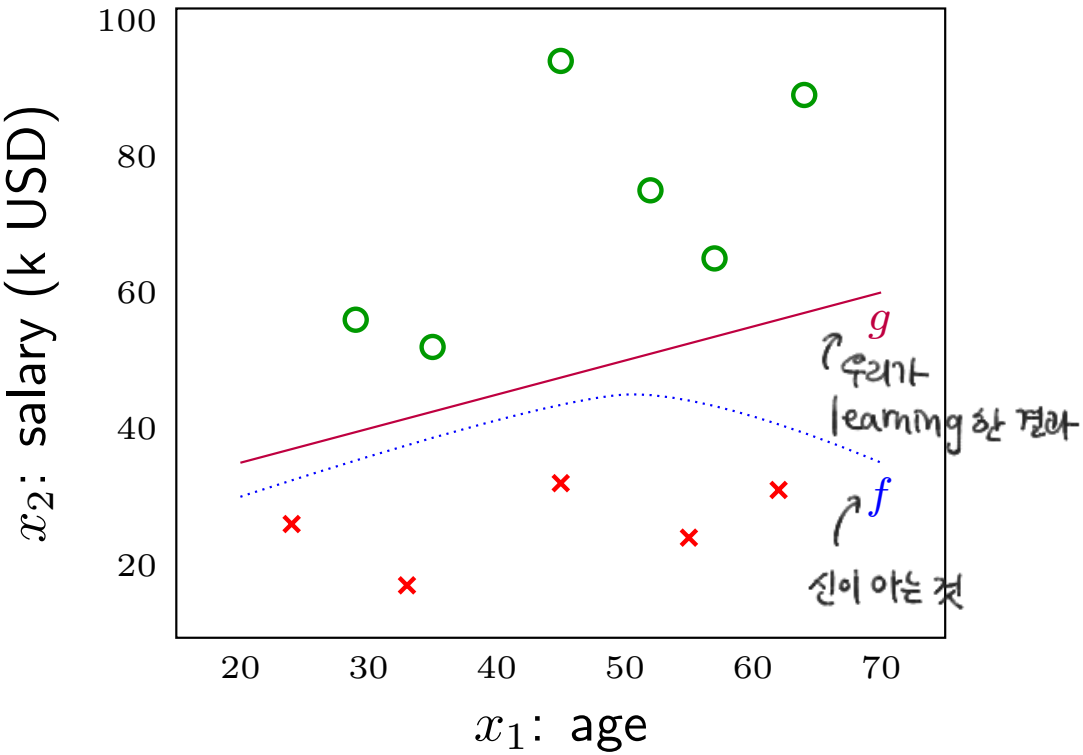
component	symbol	credit approval metaphor
input	$\mathbf{x}$	customer application
output	y	approve or deny
target function	$f : \mathcal{X} \to \mathcal{Y}$	ideal credit approval formula
data	$(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_N, y_N)$	historical records
hypothesis	$g : \mathcal{X} \to \mathcal{Y}$	formula to be used

- f: unknown target function (신이 아는 바로 그것)
- $\mathcal{X}$: input space (set of all possible inputs $\mathbf{x}$)
- $\mathcal{Y}$: output space (set of all possible outputs)
- N: the number of input-output examples (*i.e.*, training examples)
- $\mathcal{D} \triangleq \{(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_N, y_N)\}$: data set where $y_n = f(\mathbf{x}_n)$

Example

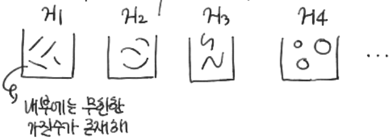
- $\mathbf{x} = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}$ where x_1: age and x_2: annual salary in USD
- $N = 11$, $d = 2$, $\mathcal{X} = \mathbb{R}^2$, and $\mathcal{Y} = \{\text{approve}, \text{deny}\}$
- data set $\mathcal{D}$:

n	x_1	x_2	y
1	29	56k	approve
2	64	89k	approve
3	33	17k	deny
4	45	94k	approve
5	24	26k	deny
6	55	24k	deny
7	35	52k	approve
8	57	65k	approve
9	45	32k	deny
10	52	75k	approve
11	62	31k	deny



Components of learning: learning algorithm

- learning algorithm $\mathcal{A}$
  - uses $\mathcal{D}$ to pick a formula $g : \mathcal{X} \to \mathcal{Y}$ that approximates f
  - chooses g from a set of candidate formula under consideration, which we call hypothesis set $\mathcal{H}$
- for example
  - $\mathcal{H}$ could be the set of all linear formulas
  - from this set the algorithm chooses the best linear fit to $\mathcal{D}$



- when a new customer applies for credit, the bank makes a decision
  - baed on g (learned), not on f (unknown)
    우리의 목적은 g를 learn 하는것
- the decision will be good only to the extent that
  - g faithfully replicates f

- the learning algorithm chooses g that
  - best matches f on *training* examples of previous customers
  - hope: this g will continue to match f on new customers

※ $f: X \to Y$는 사상은
함수가 아니라
conditional probability.

Basic setup

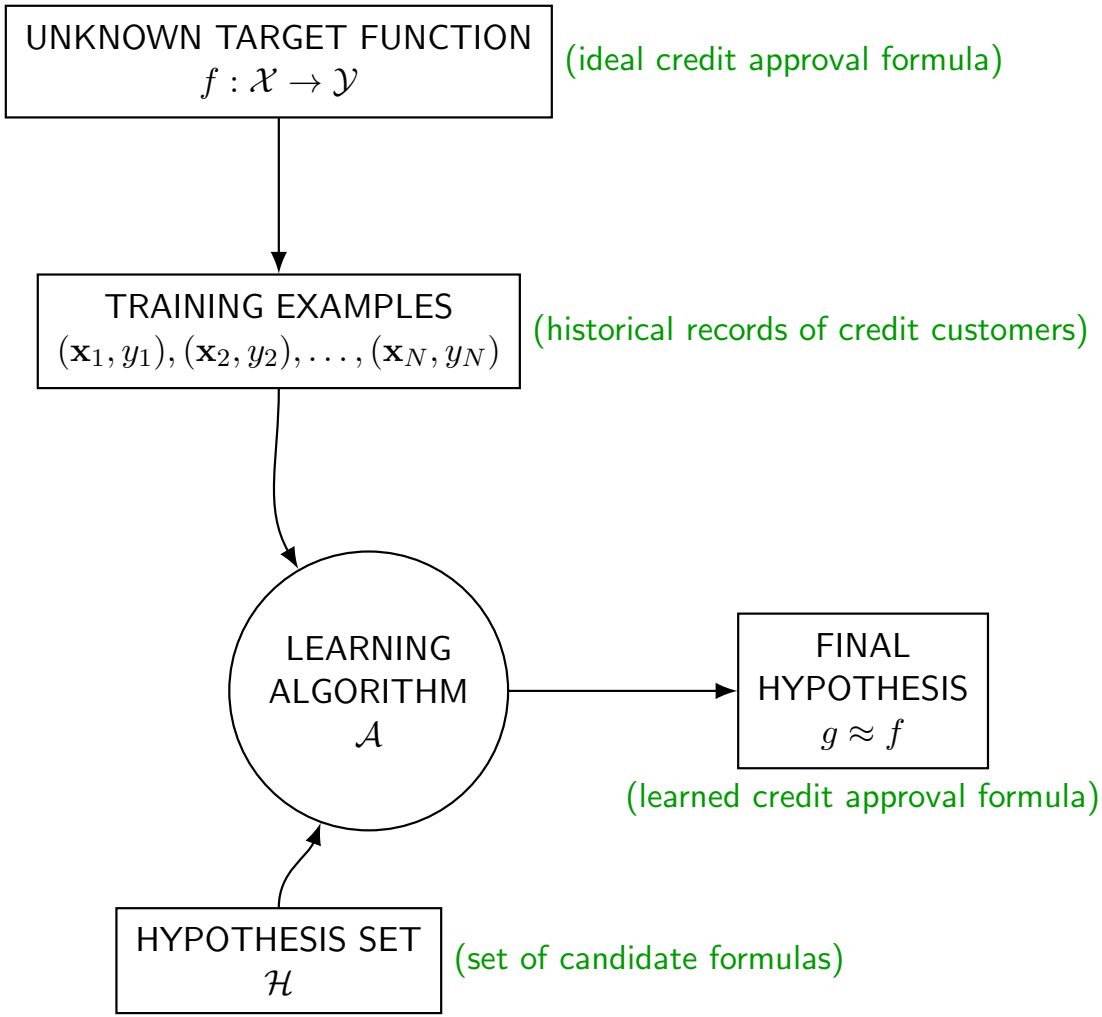
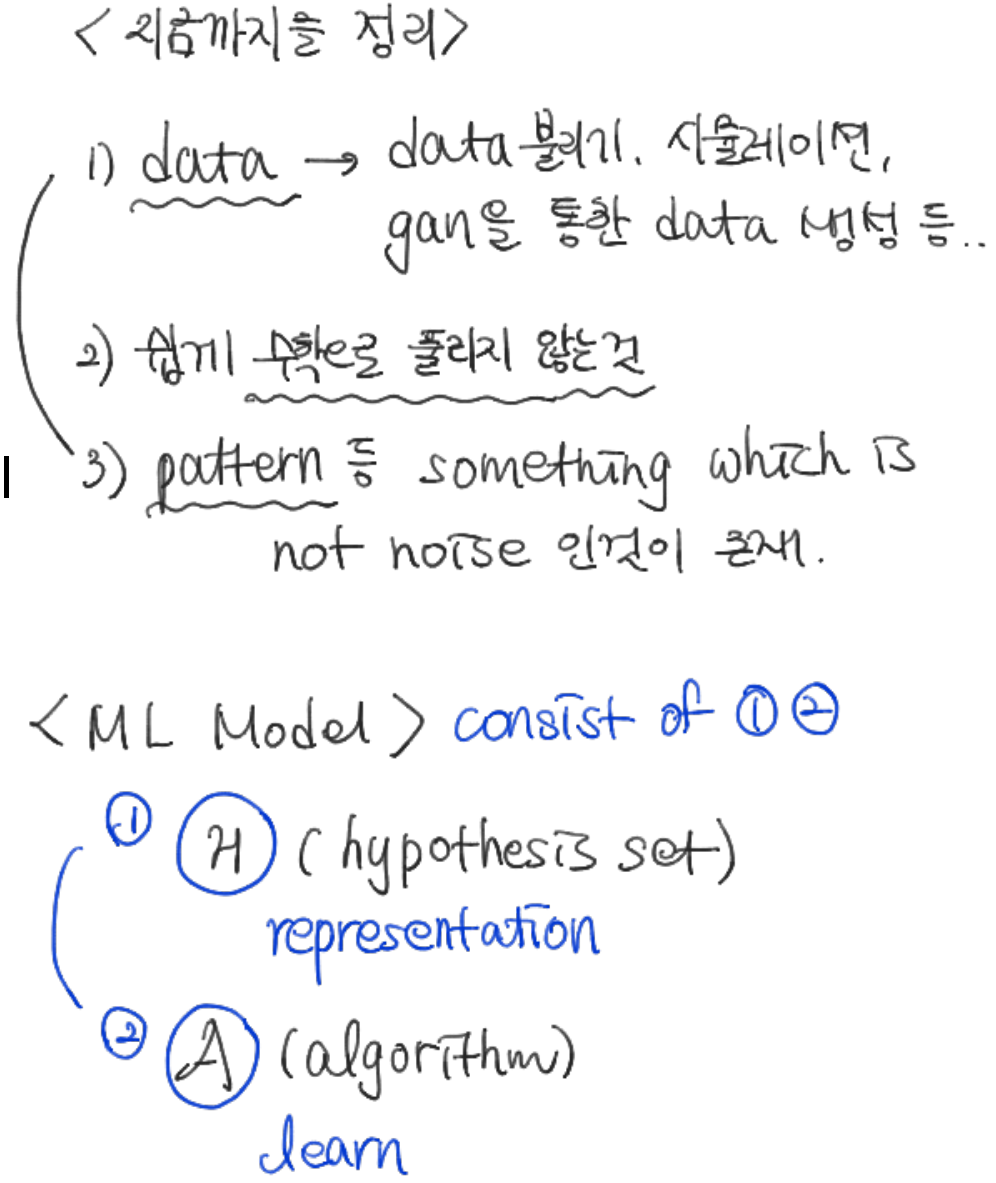


Figure 1: basic setup of the learning problem

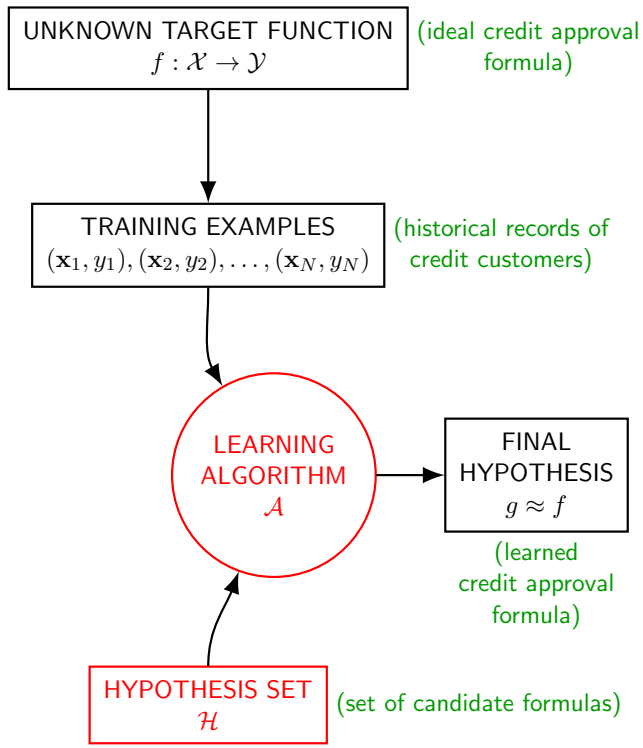
Outline



Learning model: solution components

- let's consider the different components in Figure 1

- given a specific learning problem
  - target function: unknown
  - training examples: come with the problem
  - learning algorithm, hypothesis set: not dictated by the problem



- (H) hypothesis set and (A) learning algorithm
  - two solution tools we get to choose
  - together called the "*learning model*"

Hypothesis set

- we specify the hypothesis set $\mathcal{H}$ through a functional form $h(\mathbf{x})$
  - all the hypotheses $h \in \mathcal{H}$ share this form

parametric model

- the functional form $h(\mathbf{x})$:
  - gives different weights to the different coordinates of $\mathbf{x}$
  - reflects their relative importance in the credit decision

- our choice of $h(\mathbf{x})$ here: a linear model
  - $\mathcal{H}$: a set of lines
  - key question: linear in what? (linear in w)

Outline

Making a decision

- to make a decision
  - weighted coordinates are combined to form a 'credit score'
  - the resulting score is then compared to a threshold (임계치, 기준)
- in our credit approval example
  - for input $\mathbf{x} = (x_1, \ldots, x_d)$, 'attributes of a customer':

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

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

보통 linear 하다는 것은 w를 이야기함.

The 'perceptron'

- this linear formula $h \in \mathcal{H}$ can be written more compactly as

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

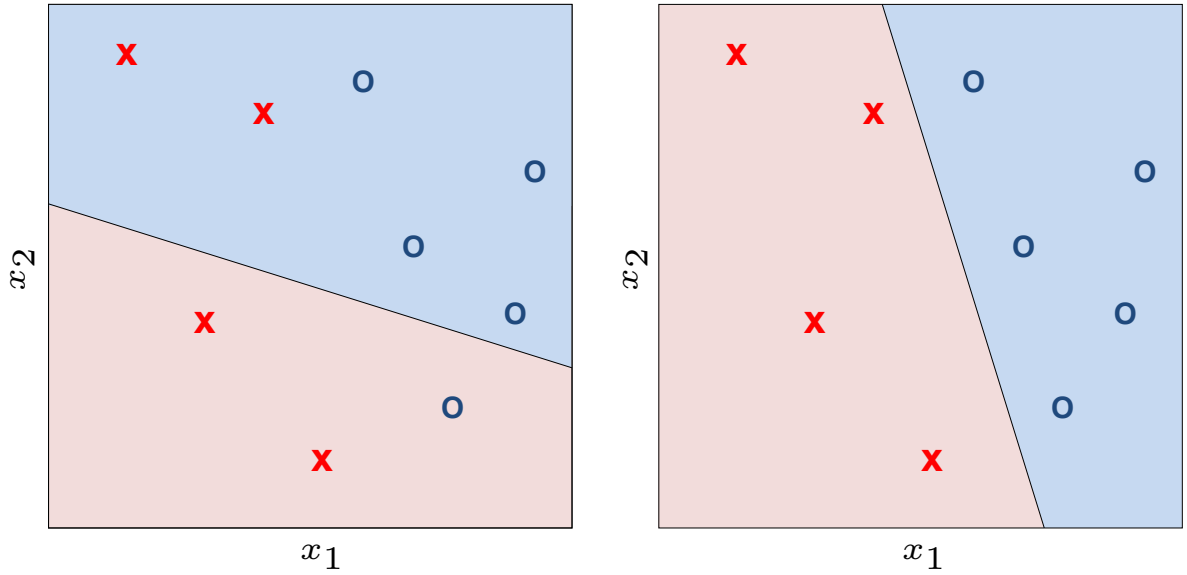
$$= \text{sign}\left(\left(\sum_{i=1}^{d} w_i x_i\right) + b\right) \tag{2}$$

where b is called the bias and $\text{sign}(s)$[1] $= \begin{cases} +1 & \text{if } s > 0 \\ -1 & \text{if } s < 0 \end{cases}$

- this model of $\mathcal{H}$ is called the *perceptron*

[1] value of $\text{sign}(s)$ when $s = 0$ is a simple technicality we can ignore for now

Two-dimensional case



(a) misclassified data (b) perfectly classified data

Figure 2: perceptron classification of linearly separable data in 2d space

직선으로 나눌수 있는 data

- the plane is split by a line into two regions
  - $+1$ decision region (blue) and -1 decision region (red)

- different values for parameters w_1, w_2, b
  - correspond to different lines $w_1 x_1 + w_2 x_2 + b = 0$

- for simplification
  - treat bias b as a weight $w_0 \equiv b$
  - introduce an artificial coordinate $x_0 \equiv 1$

- with this convention, $\mathbf{w}^{\mathrm{T}}\mathbf{x} = \sum_{i=0}^{d} w_i x_i$
  - this gives the perceptron in vector form:

$$h(\mathbf{x}) = \mathrm{sign}(\mathbf{w}^{\mathrm{T}}\mathbf{x}) \tag{3}$$

The roles of the learning alorithm

- search $\mathcal{H}$
  - by looking for weights and bias that perform well on data set
- produce the final hypothesis $g \in \mathcal{H}$
  - g is defined by the optimal (best) choices of weights and bias

Perceptron learning algorithm (PLA)

- objective
  - determine the optimal $\mathbf{w}$ based on the data to produce g
- assumption: the data set is linearly seperable
  - there is a vector $\mathbf{w}$ that makes (3) achieve the correct decision $h(\mathbf{x}_n) = y_n$ on all training examples (Figure 2)
- perceptron learning algorithm (PLA)
  - an (반복적인) iterative algorithm
  - guaranteed to converge for linearly separable data

How PLA works

- the perceptron implements

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

- given the training set

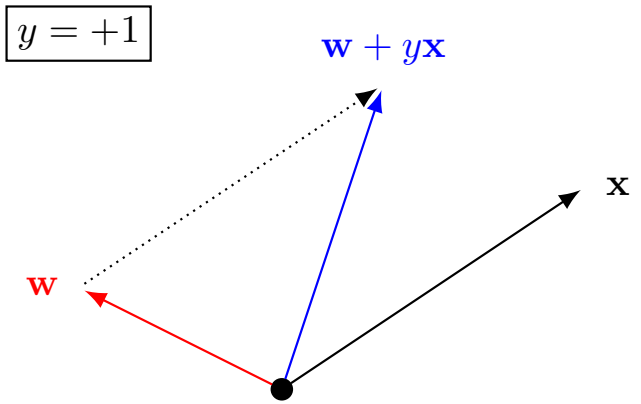
$$(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), \ldots, (\mathbf{x}_N, y_N)$$

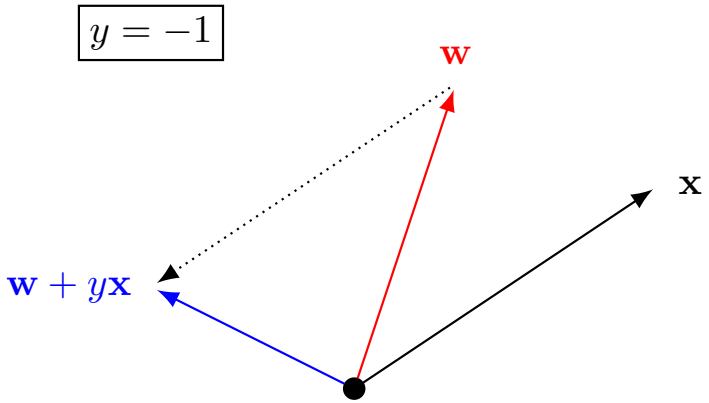
- PLA picks a misclassified point

$$\text{sign}(\mathbf{w}^{\text{T}}\mathbf{x}_n) \neq y_n \qquad (4)$$

and updates the weight vector:

$$\mathbf{w}(t+1) = \mathbf{w}(t) + y\mathbf{x} \qquad (5)$$





- since the selected example is misclassified, we have

$$y_n \neq \text{sign}(\mathbf{w}_n^{\text{T}}\mathbf{x}_n) \tag{6}$$

- as depicted in the figure above, the rule moves the boundary
  - in the direction of classifying $\mathbf{x}_n$ correctly

- the algorithm continues with further iterations
  - until there are no longer misclassified examples in the data set
  - what if the data set is not linearly separable?

PLA를 적용하는 것은 가능하지만
PLA의 iteration이 끝나지 않는다.

Iterations of PLA

- at iteration $t = 1, 2, \ldots,$ pick a unclassified point from

$$(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), \ldots, (\mathbf{x}_N, y_N)$$

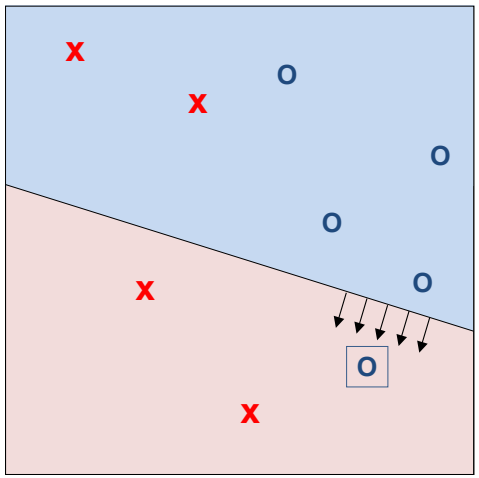
  and run a PLA iteration on it

- one iteration of the PLA:

$$\mathbf{w}(t+1) \leftarrow \mathbf{w}(t) + y_n \mathbf{x}_n \tag{7}$$

  where $(\mathbf{x}_n, y_n)$ is a misclassified training point

- that's it!



Outline

Learning paradigms

- basic premise of learning from data
  - use of observations to uncover an underlying process
  - very broad and difficult to fit into a single framework

- different learning paradigms have arisen
  - supervised learning (라벨이 있는것)
  - unsupervised learning (라벨이 없는 것)
  - reinforcement learning (interaction이 있는것.
    : panalty 와 award 가 있다)

Supervised learning

- the most studied and most utilized type of learning
  - our main focus for a while

- supervised learning setting
  - training data contains explicit examples of what the correct output should be for given inputs

- learning is 'supervised' in that
  - some 'supervisor' has taken the trouble to look at each input and determine the correct output
  - the correct 'label' is available for each training sample

ex): 사진분류 (개/고양이)

- most well-known approaches: classification & regression

categorical data를 분류하는것.

continuous한 data를 예측하는것.

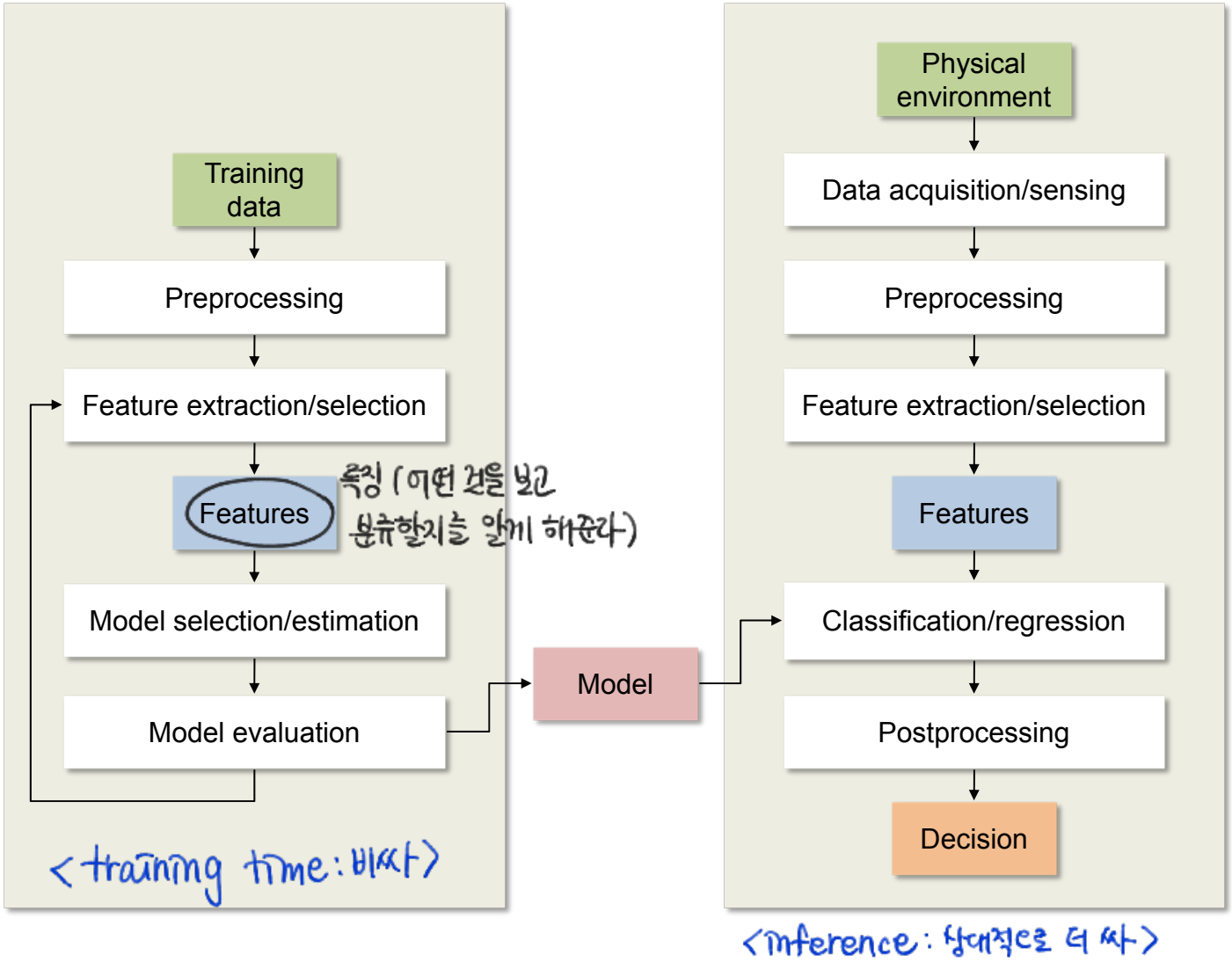
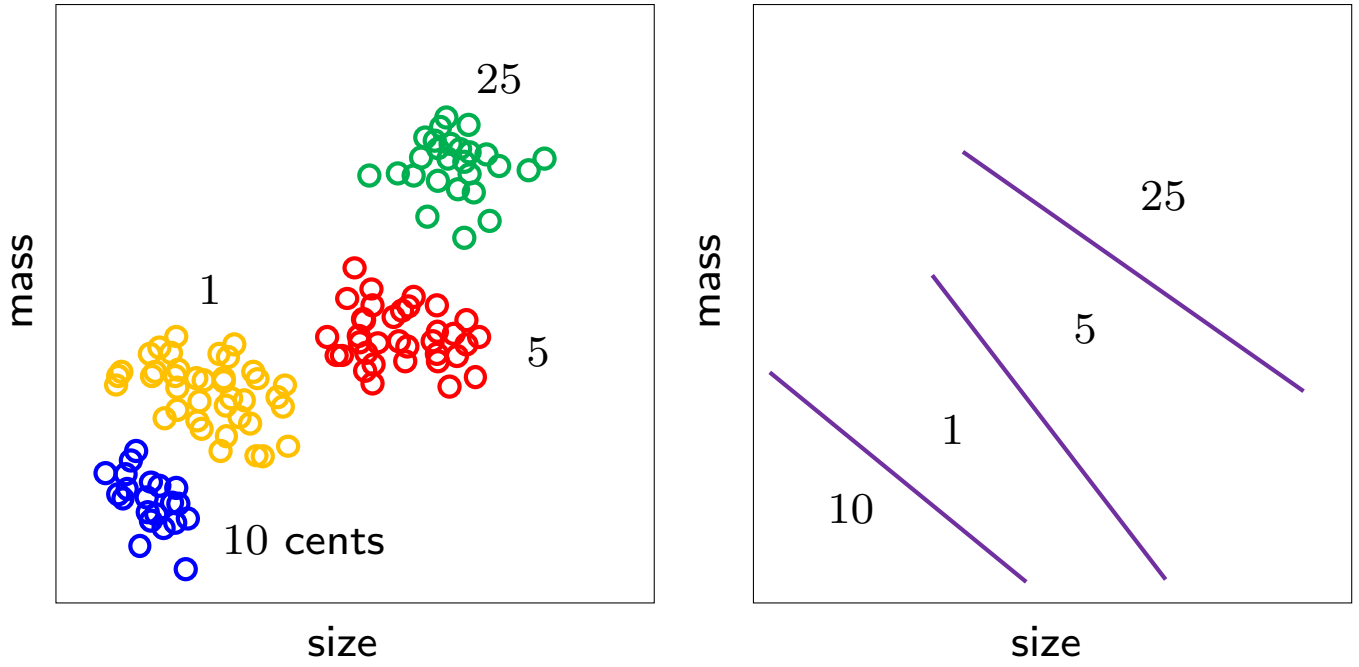


Figure 3: typical supervised learning procedure

- example from vending machines: coin recognition



(a) coin data (b) learned classifier

Figure 4: the learning approach to coin classification.

Reinforcement learning

강화학습

→ robotics, game,
체어 등에서 사용됨.

- when training data contain no correct output for each input
  - no longer in a supervised learning setting

- ex) a toddler learning not to touch a hot cup of tea
  - training examples do not say what to do
  - nevertheless, she uses the examples to reinforce better actions
  - eventually she learns what she should do in similar situations

- this characterizes *reinforcement* learning
  - the training example does not contain the target output
  - but instead contains some possible output
    together with a measure of how good that output is

- compare how a training example looks:
  - supervised learning:
    (input, correct output)
  - reinforcement learning:
    (input, some output, grade for this output)

output에 대한 평가 주어져.
그것을 다시 반영해서 학습한다.

- reinforcement learning is especially useful for learning a game, 바둑, 체스

Unsupervised learning

- the training data do not contain any output information at all
  - instead of (input, correct output), we get (input, ?)
  - that is, we are just given input examples $\mathbf{x}_1, \ldots, \mathbf{x}_N$

그냥 data만 쭉 주어져.

- how could we possibly learn anything from mere inputs?

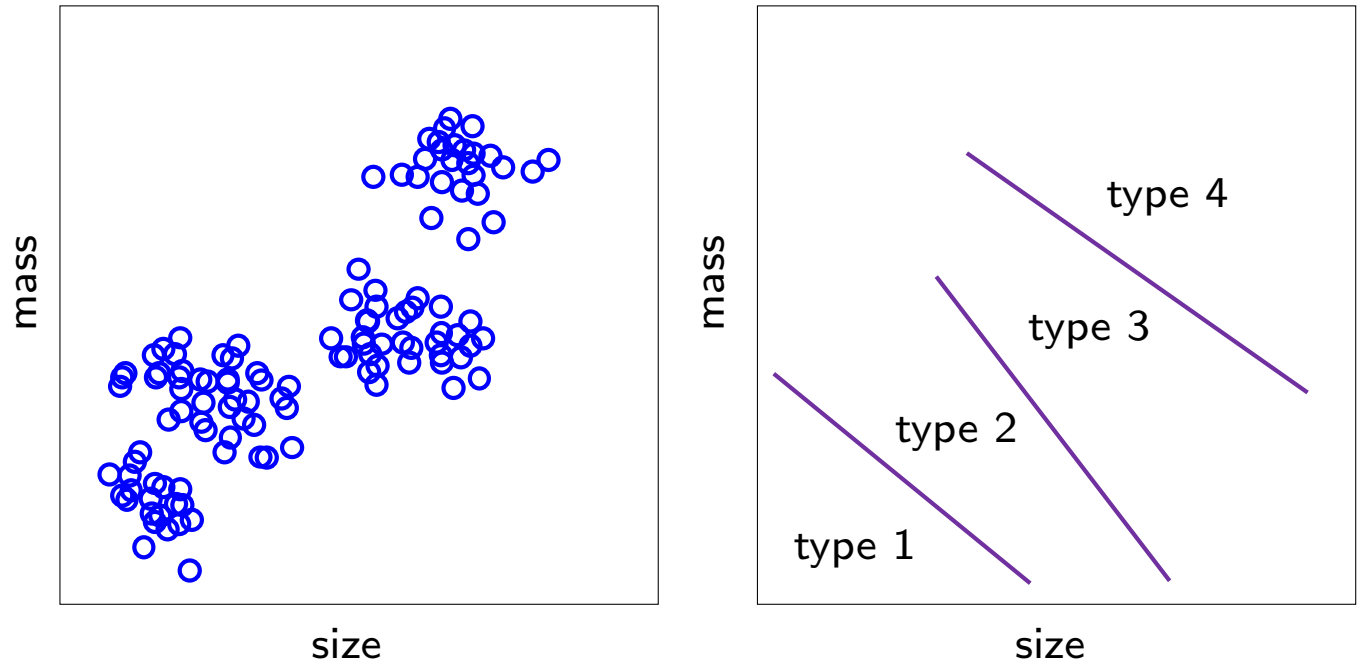
- approaches to unsupervised learning
  - clustering (*e.g.*, k-means, mixture models, hierarchical)
  - hidden Markov models (HMMs)
  - feature extraction (*e.g.*, PCA, ICA, SVD)

(군집, 유사성을)

이런 unsupervised 와 같은 말로 사용됨

- variant: semi-supervised learning

Example: coin clustering problem



(a) unlabeled coin data (b) unsupervised learning

Figure 5: the decision regions in unsupervised learning may be identical to those in supervised learning, but without the labels

- unlabeled data (라벨링되지 X) (코인의 정보: 무게나 크기, 완 있고 종류는 모르는 상황)
  - we do not know the denomination of any coin

- however, the correct clustering is less obvious now
  - and even the number of clusters may be ambiguous

- nonetheless, this example shows that
  - we can learn *something* from the inputs by themselves

Unsupervised learning can be viewed as

- spontaneously finding patterns and structure in input data
  - ex) categorize a set of books into topics
- a precursor to supervised learning
  - ex) learning Spanish without knowing the meaning first and then taking Spanish lessons will be easier to follow
- a way to create a higher-level representation of the data
  - ex) automated feature extraction

unsupervised 는 deep learning 의 initial point 를 잘 잡을수 있다.

이를 이용해 abstract out 하면 high dimension data 를 simple하게 바꿀수 있다. 예) 사람 얼굴 인식.

Statistics

AI — ML — DL
통계학 — SL (statistic learning)
DB — DM (data mining)

- shares the basic premise of learning from data
  - use of observations to uncover an underlying process
  - the process: a probability distribution
  - the observations: sampled from that distribution
- emphasis is given to situations where
  - most questions can be answered within rigorous proofs

comparison:

- statistics
  - focuses on idealized models (simplified) and analyzes them in great detail
- machine learning
  - makes less restrictive assumptions
  - deals with more general models than in statistics
  - ends up with weaker results that are broadly applicable

Data mining

- a practical field that focuses on
  - finding patterns, correlations, or anomalies
  - often in large relational databases
- examples
  - look at medical records to detect a long-term drug effect
  - look at credit card spending patterns to detect potential fraud
  - recommender systems

comparison:

- data mining vs machine learning
  - technically, the same
  - DM: more emphasis on _____ than on prediction
  - DBs are usually huge $\Rightarrow$ computational issues critical in DM

Machine learning versus statistical learning (James *et al.*)

- ML: subfield of artificial intelligence, SL: subfield of statistics
  - much overlap between the two

- different focuses:
  - ML: large-scale applications and prediction accuracy
  - SL: models and their interpretability, and precision and uncertainty → 수학적 증명을 기반으로 하는 model.

- recent trend: "cross-fertilization"
  - distinction has become blurred
  - ML has the upper hand in marketing!

Machine learning versus data mining (Wikipedia)

- two terms are commonly confused
  - often employ the same methods and overlap significantly
- they can be roughly defined as follows:
  - ML focuses on prediction, based on known properties learned from the training data
  - DM focuses on the discovery of (previously) unknown properties in the data; the analysis step of Knowledge Discovery in Databases (KDD)

patterns, knowlege..

Exercise

What types of learning, if any, best describe the following three scenarios:

(i) A coin classification system is created for a vending machine. In order to do this, the developers obtain exact coin specifications from the U.S. Mint and derive a statistical model of the size, weight, and denomination, which the vending machine then uses to classify its coins.

(ii) Instead of calling the U.S. Mint to obtain coin information, an algorithm is presented with a large set of labeled coins. The algorithm uses this data to infer decision boundaries which the vending machine then uses to classify its coins.

(iii) A computer develops a strategy for playing Tic-Tac-Toe by playing repeatedly and adjusting its strategy by penalizing moves that eventually lead to losing.

Answer: i) no learning, ii) supervised, iii) reinforcement

Outline

Summary

- learning is used when
  - we have data, and a pattern exists
  - but we can hardly pin it down mathematically
- learning model: hypothesis set and learning algorithm
  - our first example: perceptron and PLA
- types of machine learning

  supervised: (input, correct output/label)

  unsupervised: (input, no label)

  reinforcement: (input, some output, grade for this output)
- supervised learning: our main theme for a while
  - unknown target function $y = f(\mathbf{x})$
  - known training data set $(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_N, y_N)$
  - learning algorithm picks $g \approx f$ from hypothesis set $\mathcal{H}$