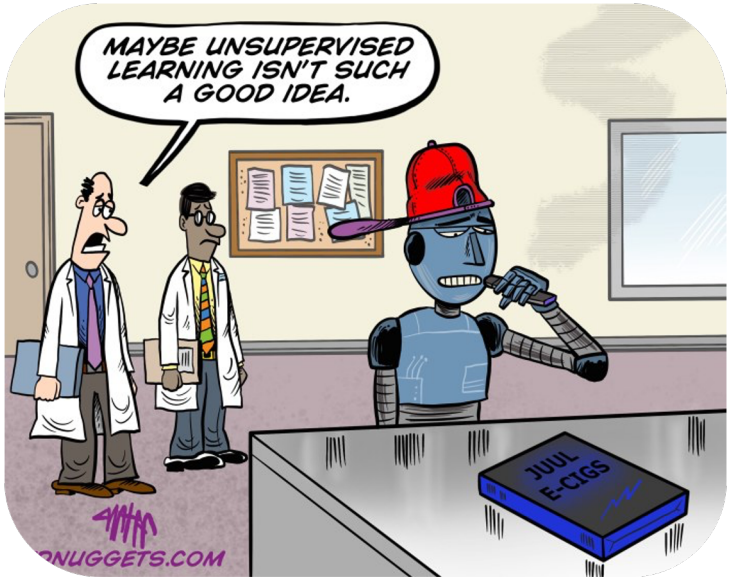
Data Science (COSE471) Spring 2021

The Learning Problem



Dept. of Computer Science and Engineering

Korea University

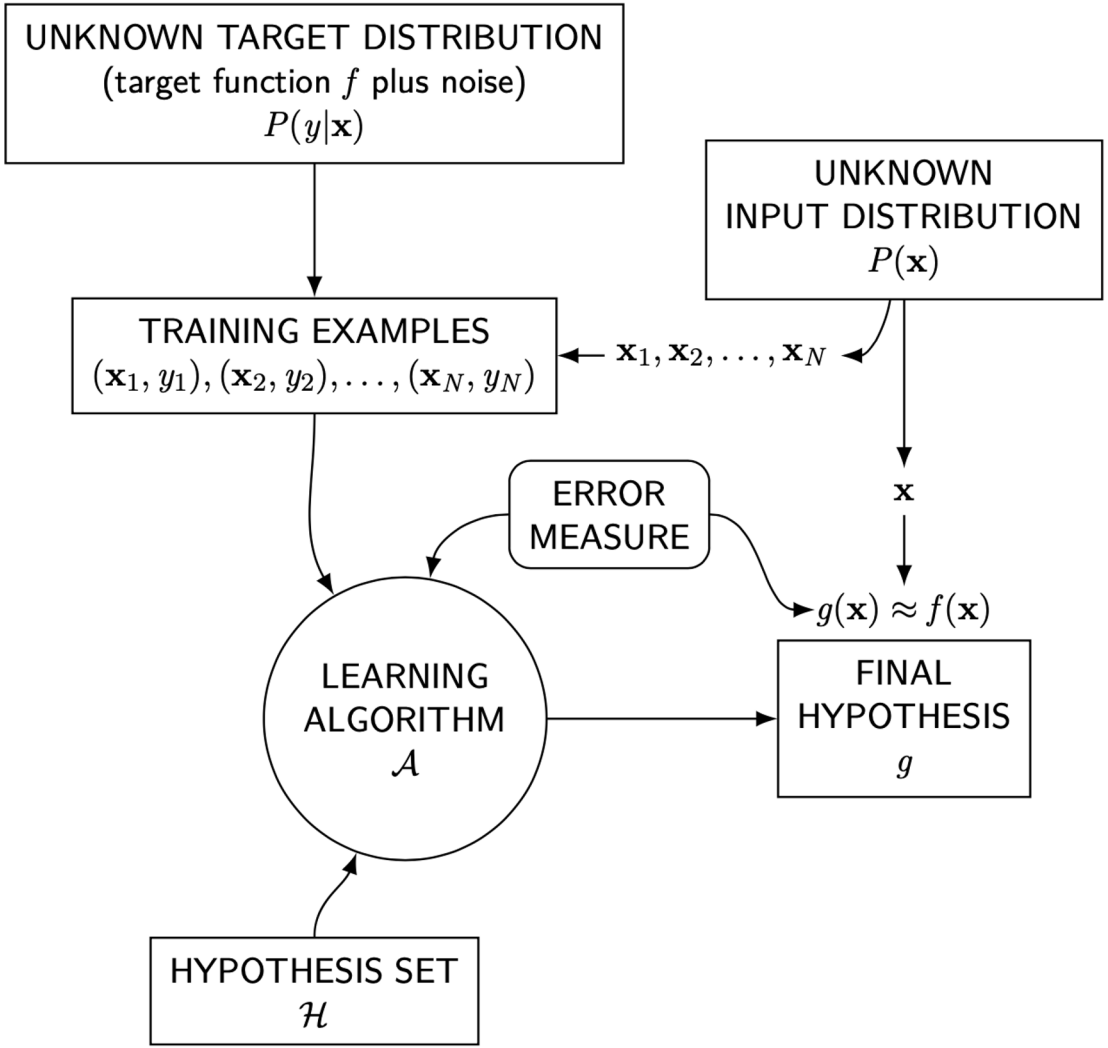
Outline

- Introduction
  - Learning from Data
  - Problem Setup
  - A Simple Learning Model
  - Perceptron
- Types of Learning
- Summary

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

- Introduction
  - Learning from Data
  - Problem Setup
  - A Simple Learning Model
  - Perceptron
- Types of Learning
- Summary

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 twenty-year-old what the definition of a tree is
  - Probably get an **inconclusive** answer

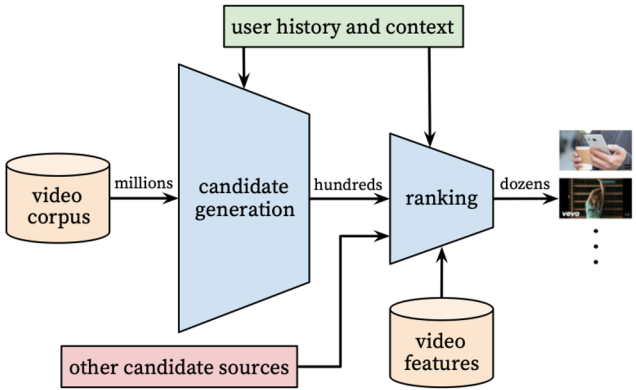
Learning from data

- 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** solution
  - But have data to construct an **empirical** solution
- The premise covers a lot of territory.

Example: Recommendation systems

- Good recommender systems are important
  - 20% sales are from recommendation (Amazon.com)
  - 10% improvement = 1 million dollar prize (Netflix)
  - Helping 1 billion users discover personalized content (Youtube.com)

Covington et al., “Deep Neural Networks for YouTube Recommendations,” RecSys, 2016.

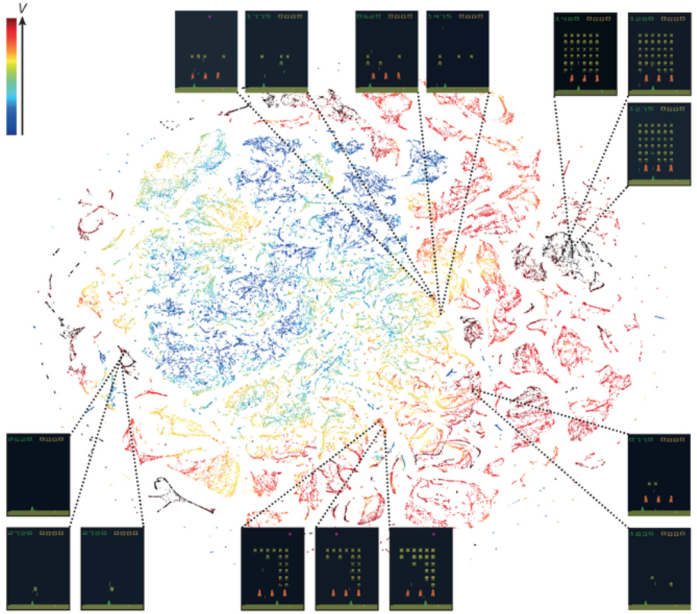


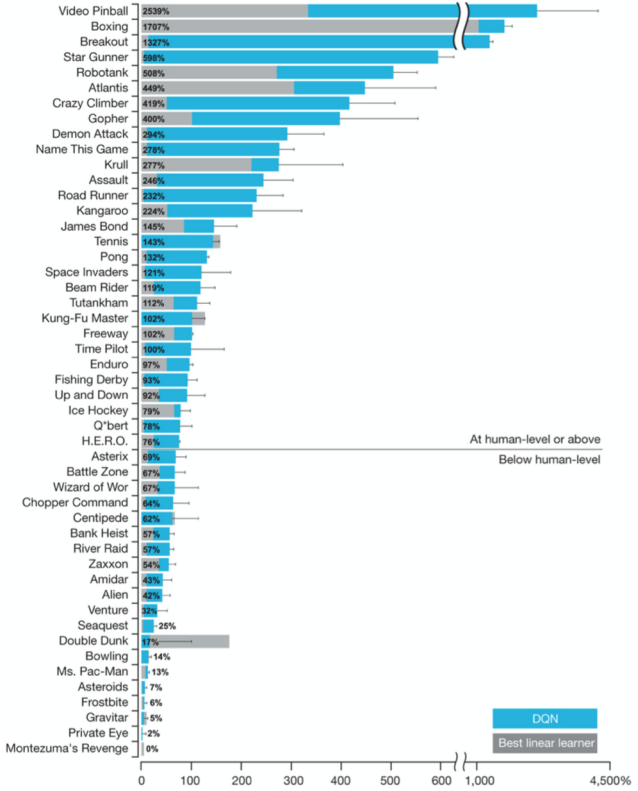
Why Learning from Data

- The main difficulty: an analytic solution is hard to get
  - The criteria users choose to view video clips: quite complex
  - Modeling those explicitly: not easy
- However, **historical data** reveal a lot about how people choose videos
- The power of learning from data
  - The entire process can be automated without any need for analyzing video content or viewer taste

Example: human-level control (game play)

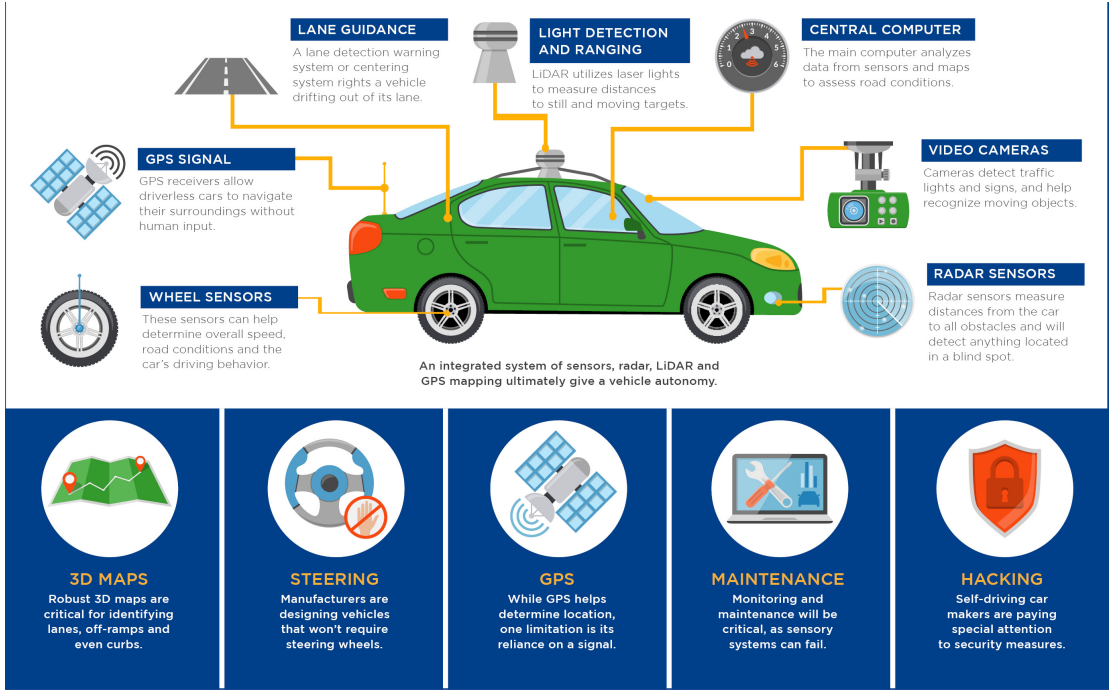
- Deep reinforcement learning
  - By Google (2015)





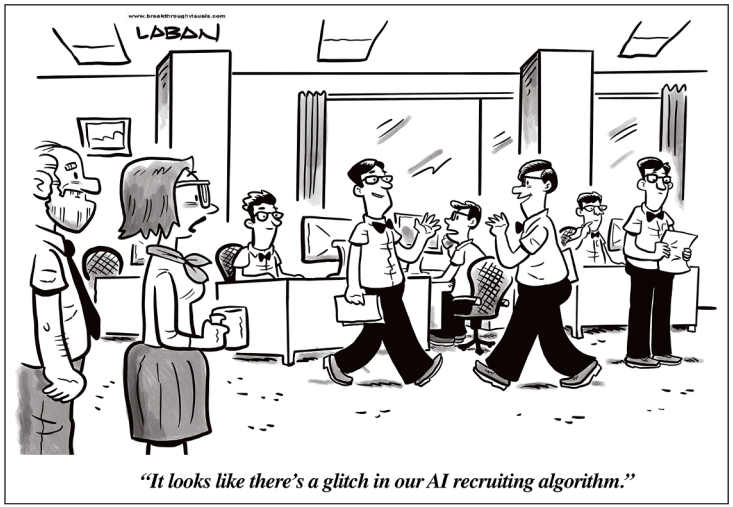
Example: Autonomous Cars

- Waymo ONE (2019)



The essence of learning from data

- We have data.
- A pattern exists therein
- We cannot pin it down mathmatically.



Exercise

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

a) Classifying numbers into primes and non-primes
b) Detecting potential fraud in credit card charges
c) Determining the time it would take a falling object to hit the ground
d) Determining the optimal cycle for traffic lights in a busy intersection

Answer:

Outline

- Introduction
  - Learning from Data
  - Problem Setup
  - A Simple Learning Model
  - Perceptron
- Types of Learning
- Summary

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
- Approve 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
  - Denote a binary decision
- In our credit example
  - Coordinates of input x:
    - Salary, debt, and other fields in a credit application
  - Binary output y : approving or denying credit

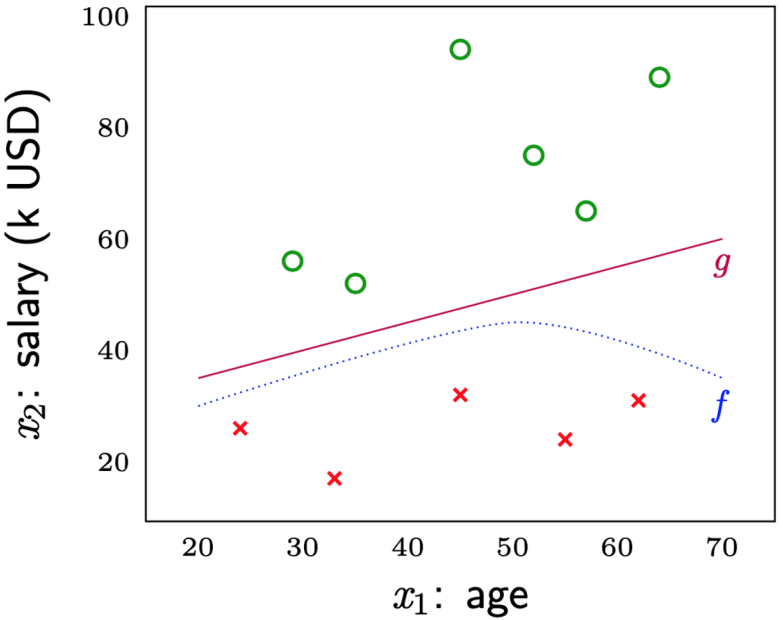
component	symbol	credit approval metaphor
input	$\mathbf{x}$	customer application
output	y	approve or deny
target function	$f : \mathcal{X} \to \mathcal{Y}$	ideal credit approval formula
data	$(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_N, y_N)$	historical records
hypothesis	$g : \mathcal{X} \to \mathcal{Y}$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

- f: unknown target function
- $\mathcal{X}$: 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



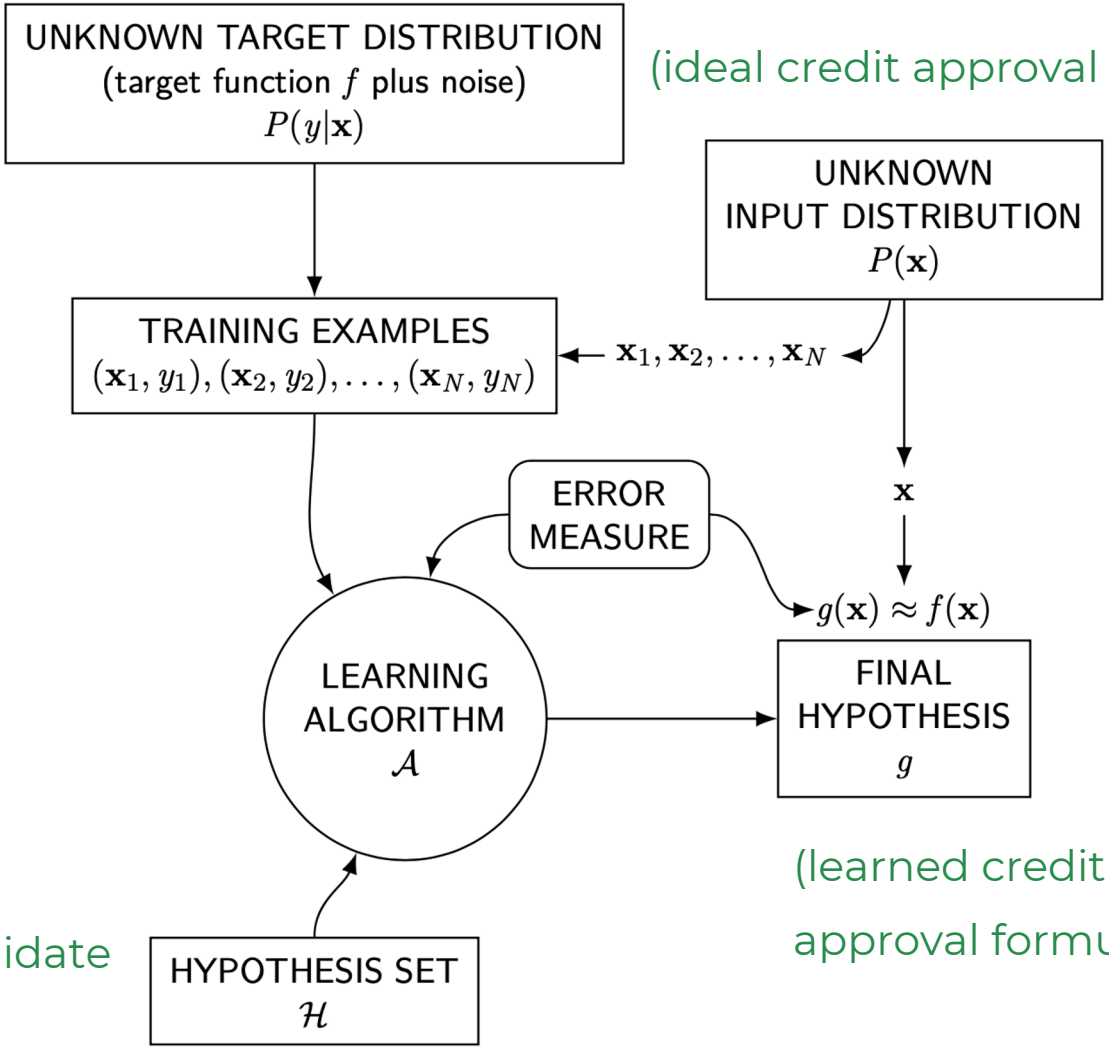
Components of learning: learning algorithm

- 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 $\mathcal{D}$

Components of learning: learning algorithm

- When a new customer applies for credit, the bank makes a decision
  - Based on g(learned), not on f (unknown)
- 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

Basic setup



(ideal credit approval formula)

(historical records of credit customers)

(learned credit approval formula)

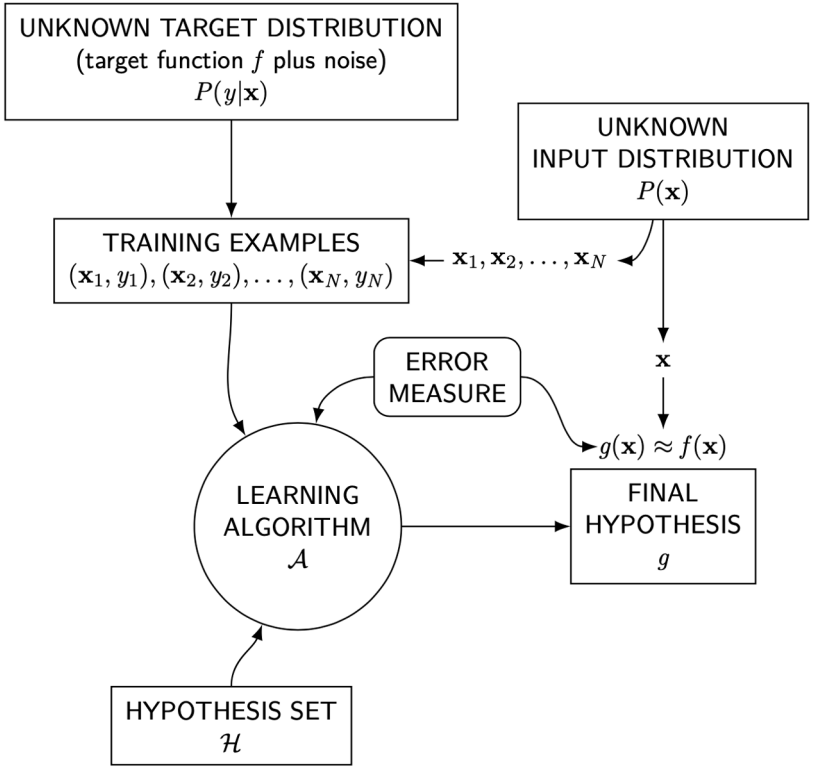
(set of candidate formulas)

Outline

- Introduction
  - Learning from Data
  - Problem Setup
  - A Simple Learning Model
  - Perceptron
- Types of Learning
- Summary

Learning model: solution components

- Given a specific learning problem
  - Target function: unknown.
  - Training examples: come with the problem
  - Learning algorithm, hypothesis set: not dictated by the problem
- Hypothesis set and learning algorithms.
  - 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 $h(\mathbf{x})$ -> y
  - All the hypotheses $h \in \mathcal{H}$ share this form
- The functional form $h(\mathbf{x})$: Sign[Sigma(i) Wi Xi]
  - 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?

Outline

- Introduction
  - Learning from Data
  - Problem Setup
  - A Simple Learning Model
  - Perceptron
- Types of Learning
- Summary

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.
- 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 $\sum_{i=1}^{d} w_i x_i < \text{threshold}$

The ‘perceptron’

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

linear model

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

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

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

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

Two-dimensional case

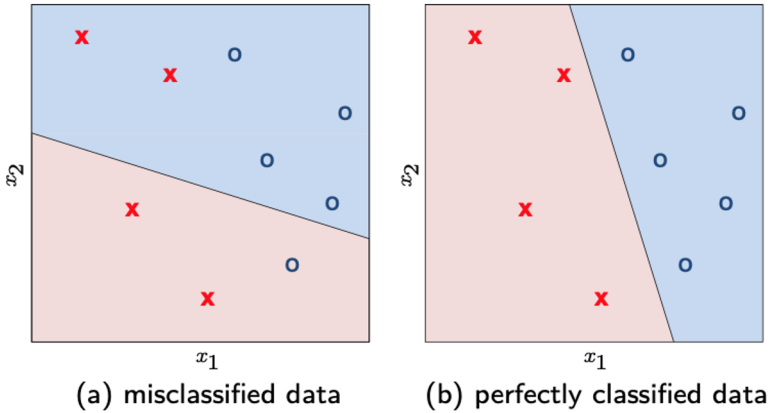


Figure 2: perceptron classification of linearly separable data in 2d space

- 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 x0 =(=3개짜리) 1

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

$$\boxed{h(\mathbf{x}) = \text{sign}(\mathbf{w}^\top \mathbf{x})} \qquad (3)$$

The roles of the learning algorithm

- search $\mathcal{H}$
  - by looking for weights and bias that perform well on training data set

- produce the final hypothesis $g \in \mathcal{H}$
  - g is defined by the optimal choices of weights and bias

Perceptron learning algorithm (PLA)

- objective
  - determine the optimal $\mathbf{w}$ based on the data to produce

- 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 iterative algorithm
  - guaranteed to converge for linearly separable data

How PLA works

들어온 x값을 토대로 추정하는 것
원래 값을 갖고 있으므로

- the perceptron implements

Transpose

$$h(\mathbf{x}) = \text{sign}(\mathbf{w}^\top \mathbf{x})$$ -> y1' != y1

- given the training set

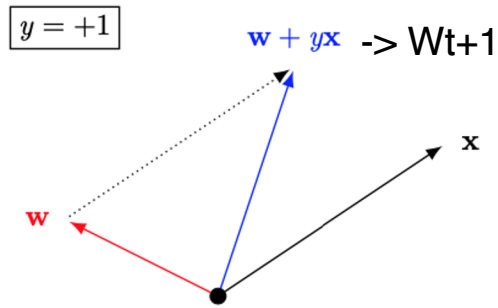
$$(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), \ldots, (\mathbf{x}_N, y_N)$$

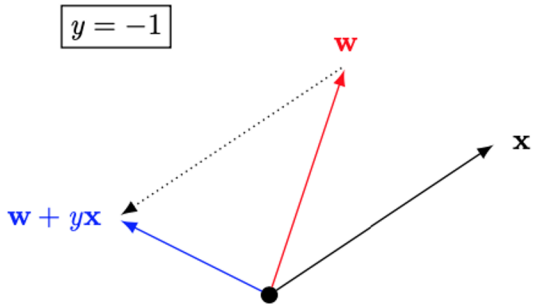
- PLA picks a misclassified point

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

and updates the weight vector:

Wt+1 <- Wt + YnXn (5)





- since the selected example is misclassified, we have

$$y_n \neq \text{sign}(\mathbf{w}_n^\top \mathbf{x}_n)$$

- 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?

Iterations of PLA

- at iteration t = 1, 2, ..., pick a misclassified point from

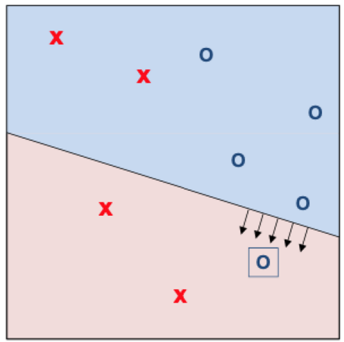
$$(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), \ldots, (\mathbf{x}_N, y_N)$$

  and run a PLA iteration on it

- one iteration of the PLA:

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

  where is a misclassified training point



- That's it

Outline

- Introduction
  - Learning from Data
  - Problem Setup
  - A Simple Learning Model
  - Perceptron
- Types of Learning
- Summary
- Appendix: Comparison

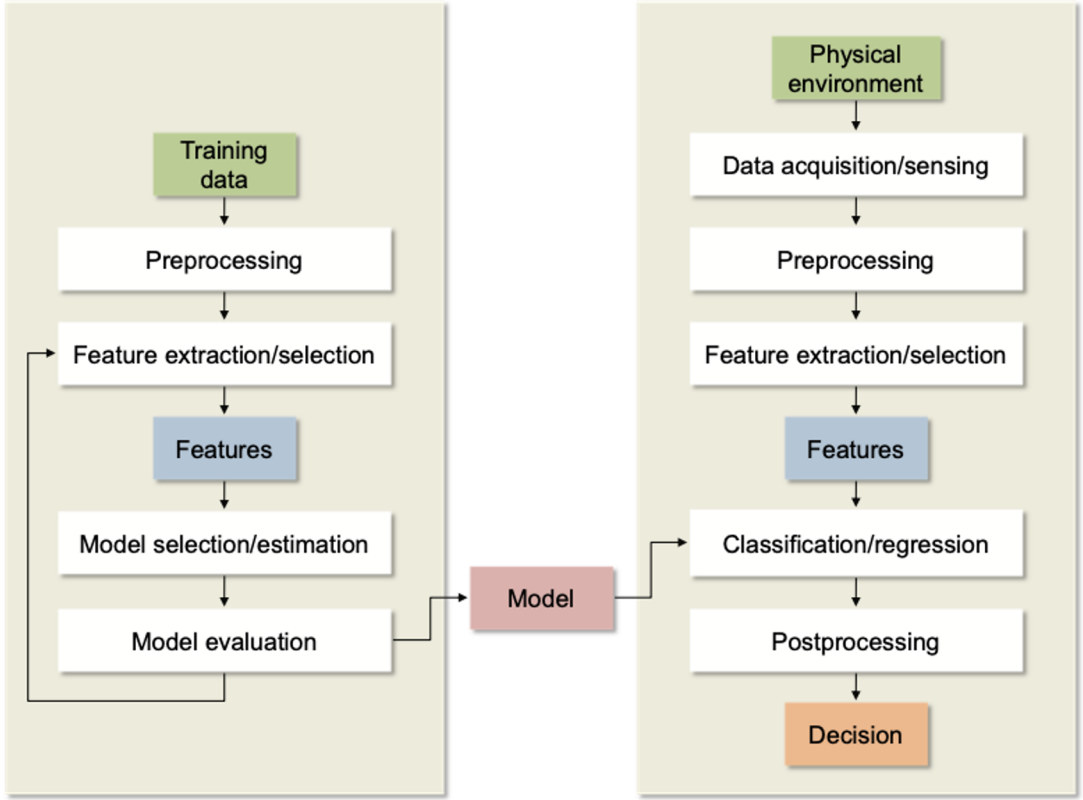
Learning paradigms

- Basic premise of learning from data
  - Use of observation to uncover an underlying process
  - Very broad and difficult to fit into a single framework
- Different learning paradigms have arisen
  - Supervised learning
  - Unsupervised learning < semi - supervised learning
  - Reinforcement learning

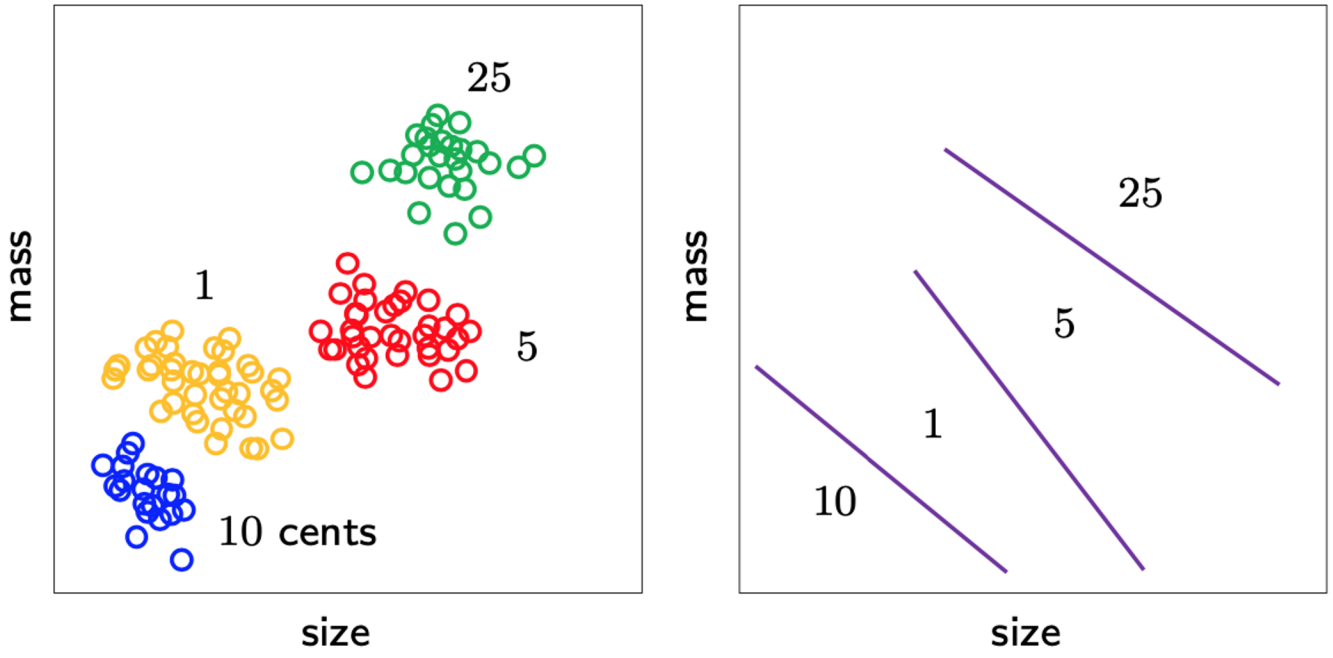
Supervised learning

- 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 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
- Most well-known approaches: regression & classification.

Typical supervised learning procedure



Example from vending machines: coin recognition



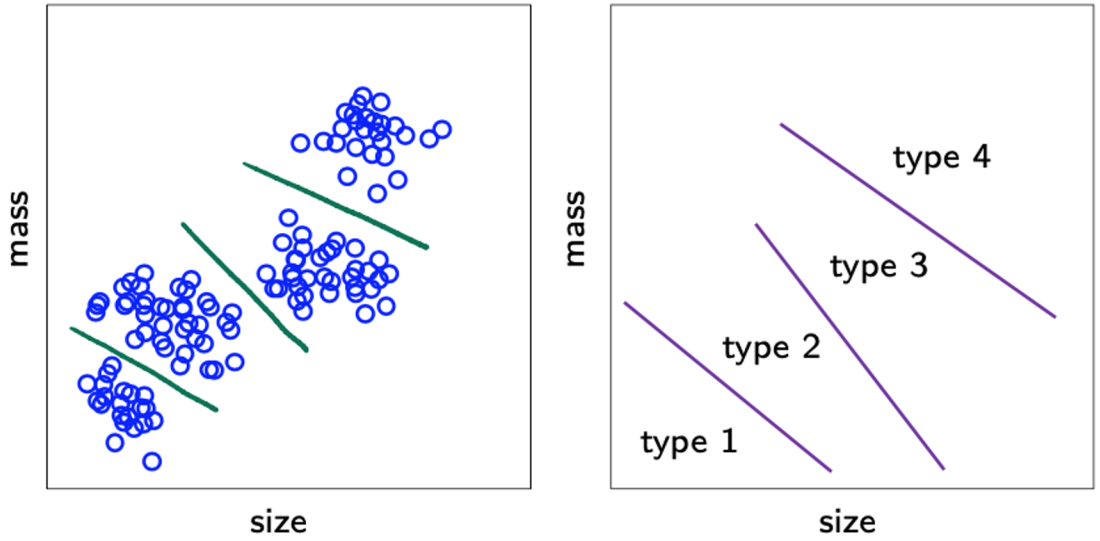
(a) coin data

(b) learned classifier

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
- how could we possibly learn anything from mere inputs?
- approaches to unsupervised learning
  - clustering (k-means, mixture models, hierarchical)
  - hidden Markov models (HMMs)
  - feature extraction (e.g. PCA, ICA, SVD)

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

Unsupervised learning

- unlabeled data
  - 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 pattern & structure in input data
  - e.g. categorize a set of books into topics

- a precursor to supervised learning
  - e.g. learning Spanish without knowing the meaning first and then taking Spanish lessons will be easier to follow

- a way to create higher-level representation of the data
  - e.g. automated feature extraction

Reinforcement Learning

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

- e.g. 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, reward for this output)

- reinforcement learning is especially useful for learning a _game_.

- What types of learning, if any, best describe the following three scenarios:
  - 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 coin.
  - Instead of calling the U.S. Mint to obtain coin information, an algorithm is presented with a larger set of labeled coins. The algorithm uses this data to infer decision boundaries which the vending machine then uses to classify its coins.
  - 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.

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 $\mathcal{H}$