기계학습 기초 및 응용

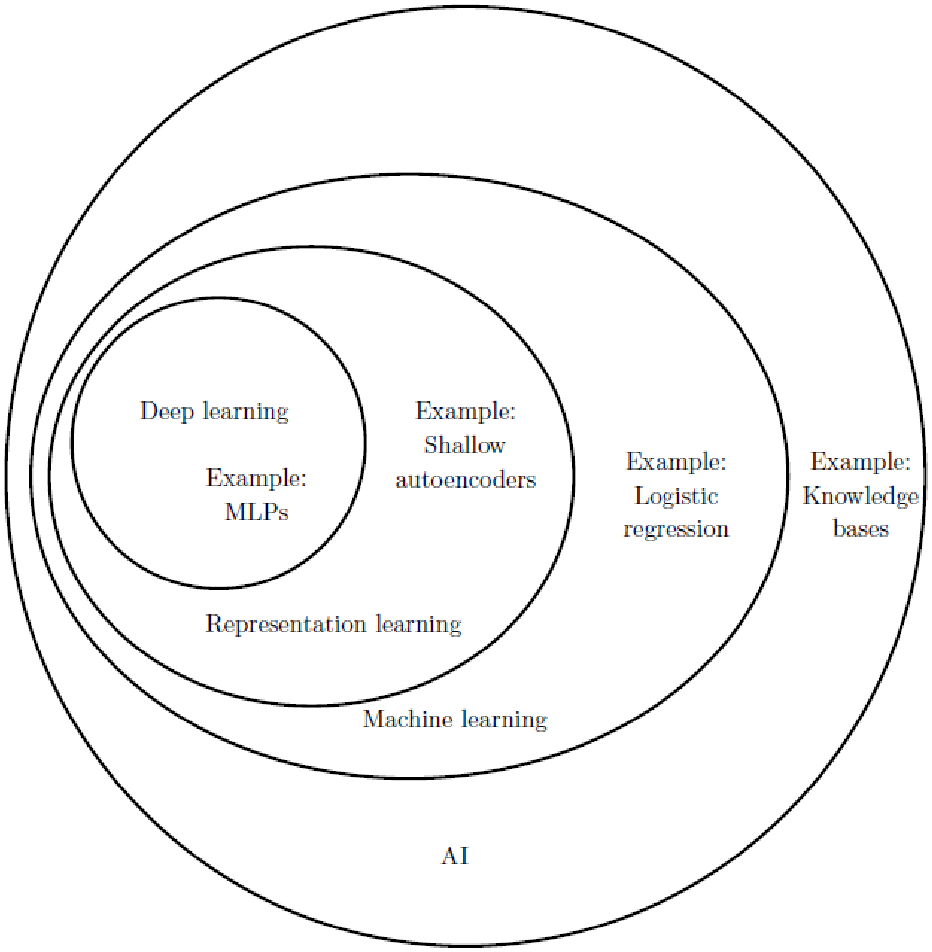
학습 문제

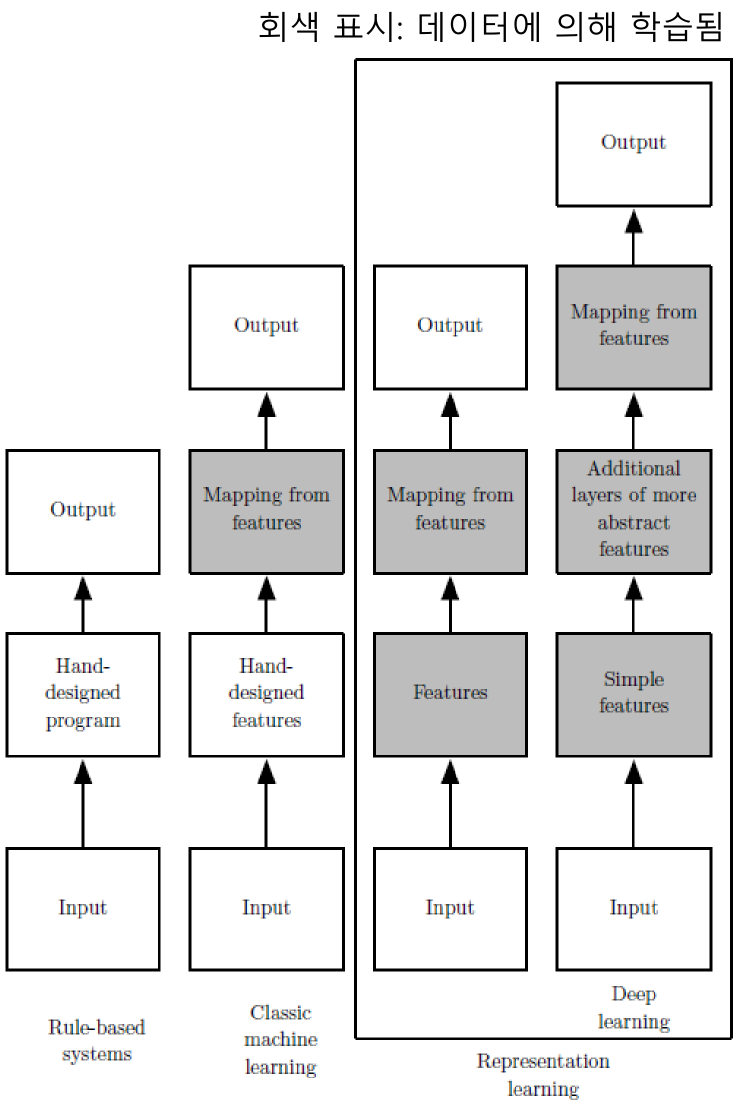
소프트웨어융합대학
소프트웨어학부

국민대학교
KOOKMIN UNIVERSITY

인공지능과 기계학습

■ 인공지능 분류

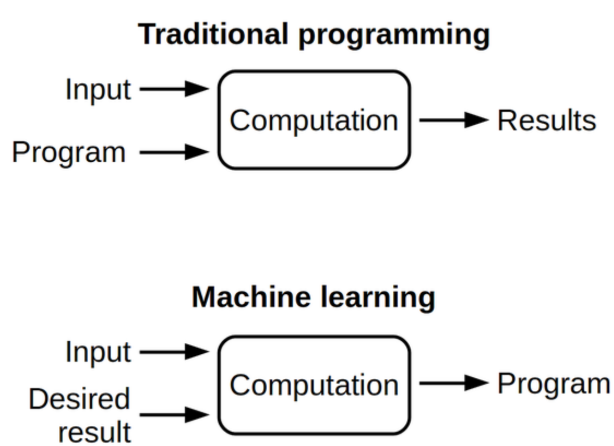




기계학습

- 기계학습Machine learning 정의
  - 인공지능의 파생 방법
  - 기기를 인간처럼 학습시켜 스스로 규칙을 형성
    - 명시적인 프로그래밍 없이 데이터를 학습
    - 학습: 특정 작업task의 성능performance을 점진적으로 개선
  - A computer program
    - improve their **performance P** (accuracy, error rate,…)
    - at some **task T** (classification, regression, detection,…)
    - with **experience E** (data)

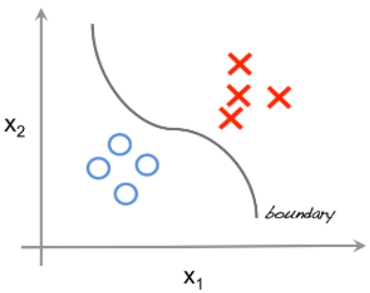
→ well-defined learning task: <**P**, **T**, **E**>



기계학습 문제

기계학습 문제 분류 비교

교사 학습

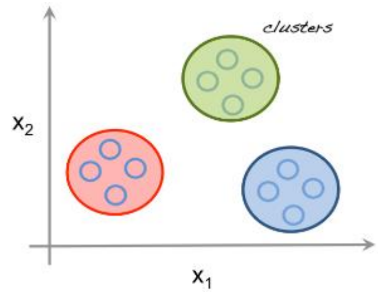


given (x, y)
x is data, y is its label

Goal:
learn a function
to map x → y

Examples:
Classification
Regression
Object detection
Segmentation
Image captioning

비교사 학습

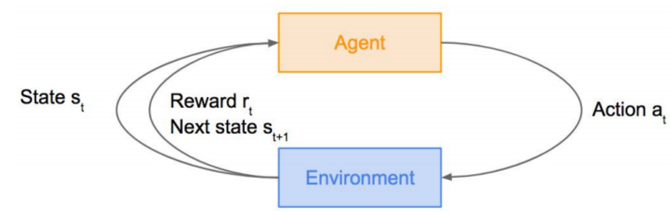


given (x)
just data, no label

Goal:
learn some underlying hidden
structure of the data

Examples:
Clustering
Dimensionality reduction
Feature learning
Density estimation

강화 학습

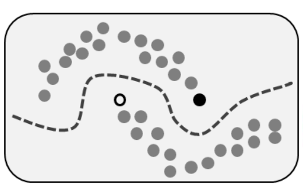


given
Problems involving an agent
interacting with an environment which
provides numeric reward signals

Goal:
Learn how to take actions in order to
maximize reward

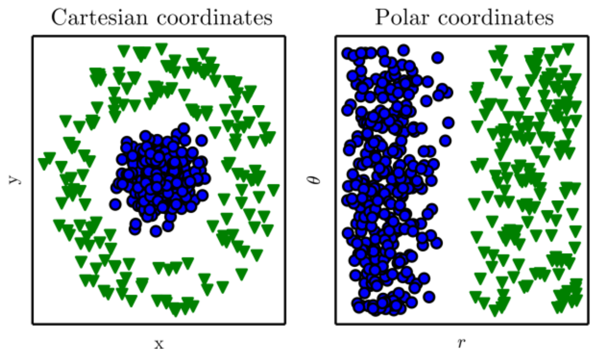
Examples:
Robotics
Self-driving

준교사 학습

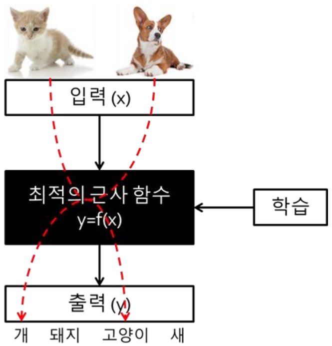


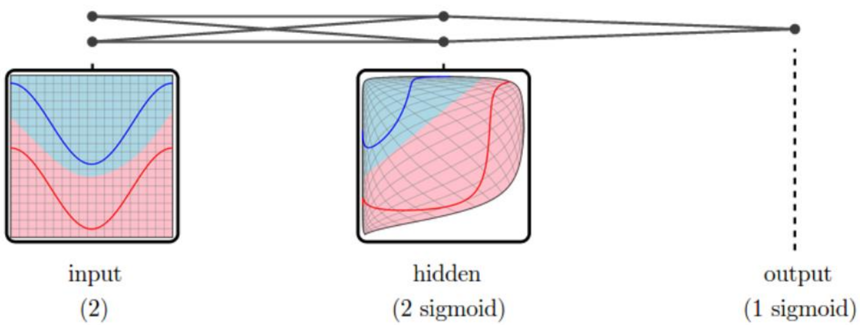
표현학습과 심층학습

- **표현학습** Representation learning
  - 기계학습의 파생 방법
  - 표현 문제 Representation matter
    - 표현의 차이 비교



  - 심층학습 deep learning: 표현학습 representation learning의 주요 방법
    : 표현에서 출력으로의 사상mapping뿐만 아니라 표현 자체를 학습하여 보다 좋은 성능을 가짐
    - 데이터에서 주어진 작업에 필요한 표현representation을 자동 추출
    - 데이터 중심 특징 data-driven feature의 계층적 학습





표현학습과 심층학습

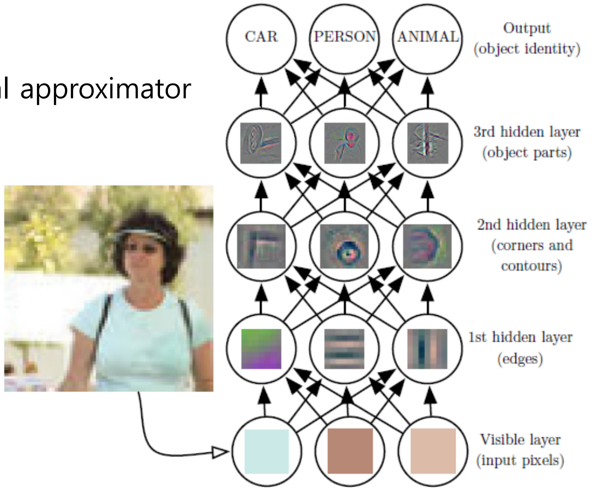
심층학습

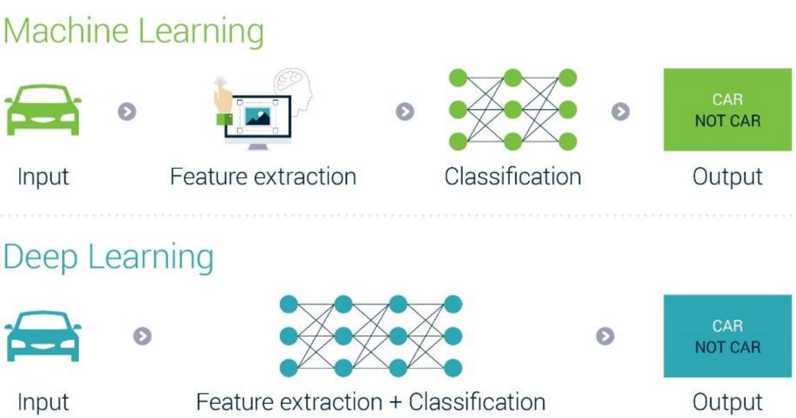
- 선형과 비선형 연산을 갖춘 깊은 인공신경망 deep artificial neural network
- 추상화 수준이 높아진 표현의 계층을 가짐
- 깊은 인공신경망 == 범용 근사 함수 universal approximator

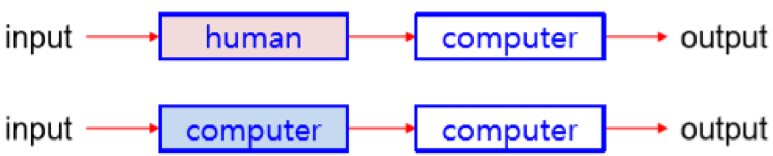
Image
pixel → edge → texton → motif → part → object

Text
character → word → word group → clause → sentence → story

Speech
sample → spectral band → sound → ... → phone → phoneme → word →

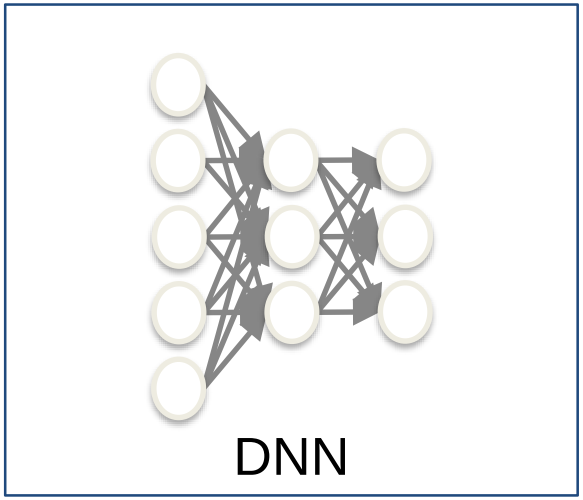




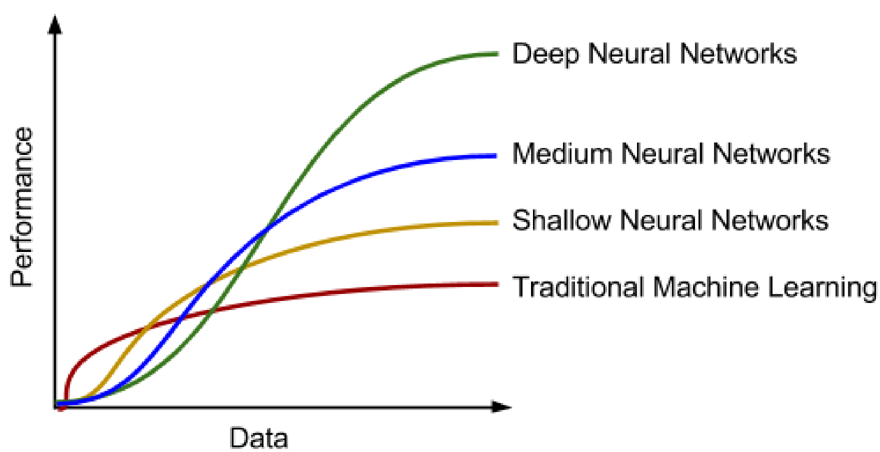


심층학습

- 심층학습의 성공 이유



- |data|와 성능 비교



기계학습 기초 (학습 문제)

기계학습 문제화

- 사례: 신용 승인 credit approval
  - given: 신청자 정보

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

  - task: 승인? 혹은 거절?

기계학습 기초 (학습 문제)

표기 정리

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 $y_n = f(\mathbf{x}_n)$

기계학습 기초 (학습 문제)

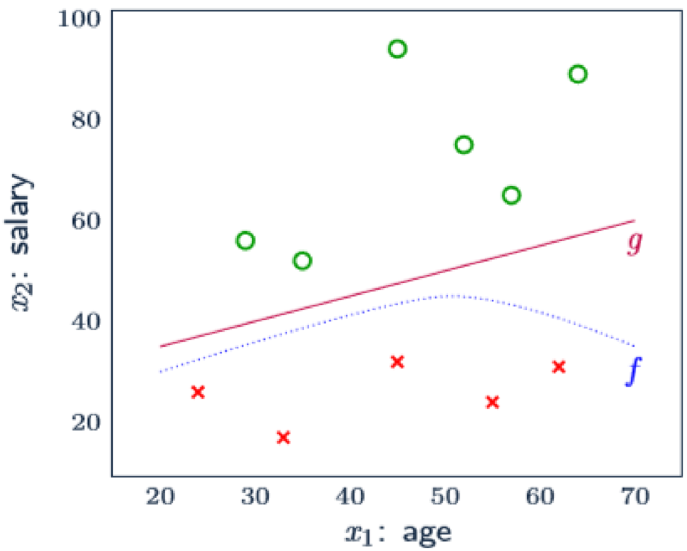
- 문제 해결

$\mathbf{x} = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}$ where x_1: age and x_2: annual salary in USD

$N = 11$, $d = 2$, $\mathcal{X} = \mathbb{R}^2$, and $\mathcal{Y} = \{\text{approve}, \text{deny}\}$

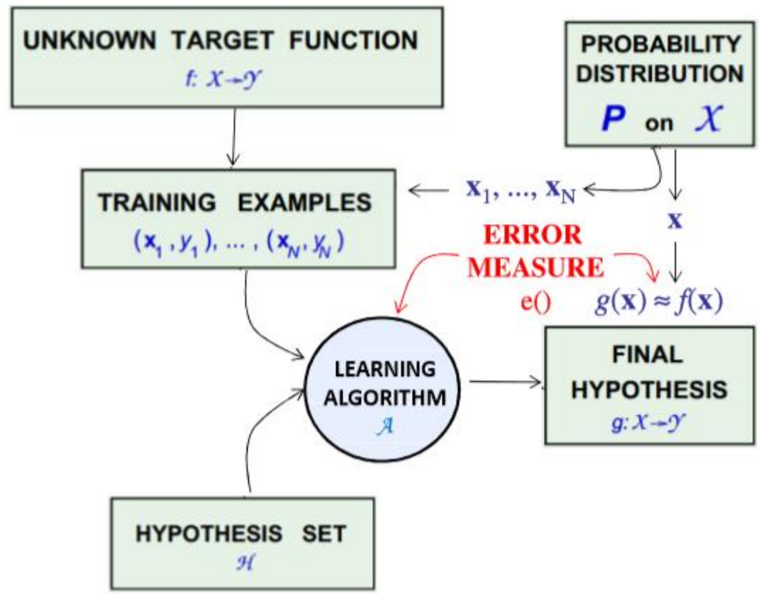
data set $\mathcal{D}$:

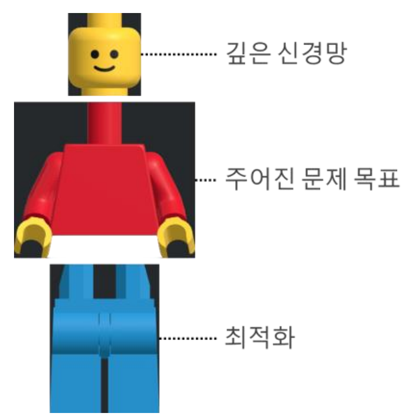
n	x_1	x_2	y
1	29	56k	approve
2	64	89k	approve
3	33	17k	deny
4	45	94k	approve
5	24	26k	deny
6	55	24k	deny
7	35	52k	approve
8	57	65k	approve
9	45	32k	deny
10	52	75k	approve
11	62	31k	deny



기계학습 기초 (학습 문제)

- 기계학습 개요



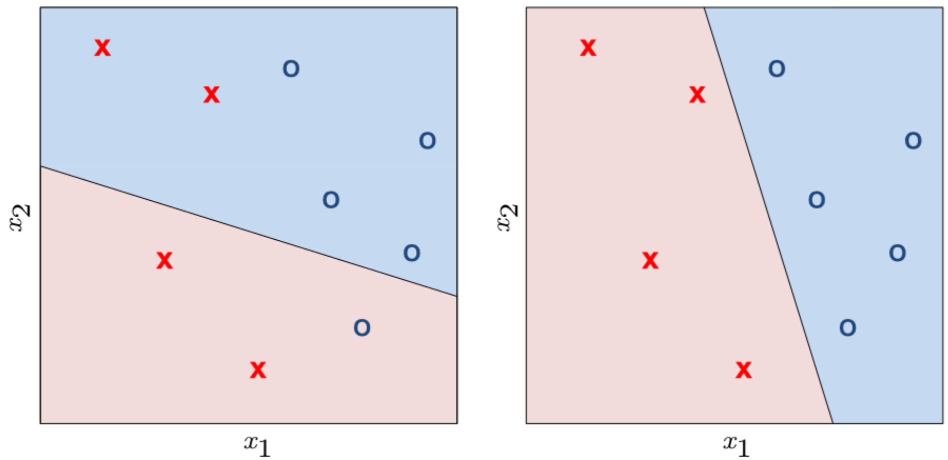


Hypothesis set

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

기계학습 기초 (학습 문제)

Two-dimensional case



(a) misclassified data (b) perfectly classified data

Figure : 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)

A simple hypothesis set - the 'perceptron'

For input $\mathbf{x} = (x_1, \cdots, x_d)$ 'attributes of a customer'

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

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

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

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

기계학습 기초 (학습 문제)

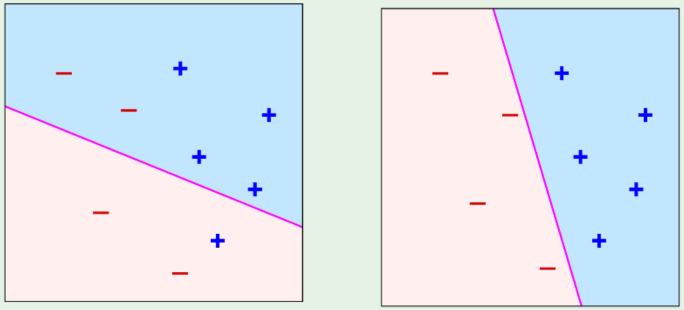
$$h(\mathbf{x}) = \text{sign}\left(\left(\sum_{i=1}^{d} w_i \; x_i\right) + \quad w_0 \quad\right)$$

Introduce an artificial coordinate $x_0 = 1$:

$$h(\mathbf{x}) = \text{sign}\left(\sum_{i=0}^{d} w_i \; x_i\right)$$

In vector form, the perceptron implements

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



'linearly separable' data

The roles of the learning alorithm

$$\mathcal{H} = \{h(\mathbf{x}) = \mathrm{sign}(\mathbf{w}^{\mathrm{T}}\mathbf{x})\} \quad \longleftarrow \text{uncountably infinite } \mathcal{H}$$

- 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 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 separable
  - there is a vector $\mathbf{w}$ that makes $h(\mathbf{x})$ achieve the correct decision $h(\mathbf{x}_n) = y_n$ on all training examples (Figure)
- perceptron learning algorithm (PLA)
  - an incremental algorithm
  - guaranteed to converge for linearly separable data

기계학습 기초 (학습 문제)

A simple learning algorithm - PLA

The perceptron implements

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

Given the training set:

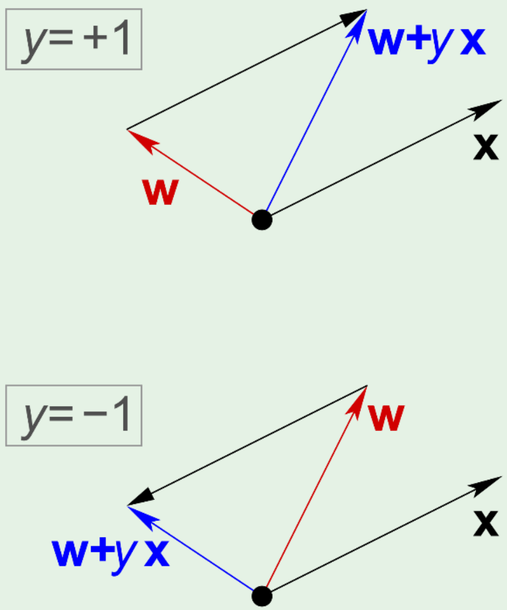
$$(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), \cdots, (\mathbf{x}_N, y_N)$$

pick a misclassified point:

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

and update the weight vector:

$$\boxed{\mathbf{w} \leftarrow \mathbf{w} + y_n \mathbf{x}_n}$$



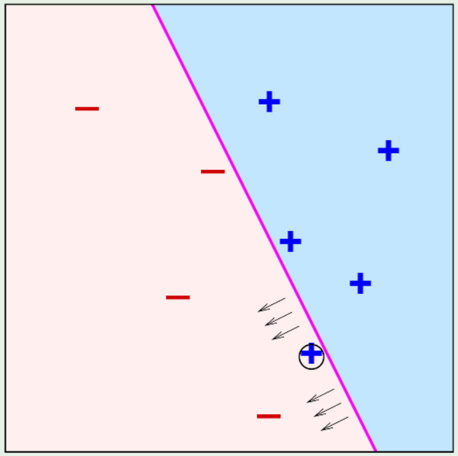
기계학습 기초 (학습 문제)

Iterations of PLA

- One iteration of the PLA:

$$\mathbf{w} \leftarrow \mathbf{w} + y\mathbf{x}$$

where $(\mathbf{x}, y)$ is a misclassified training point.



- At iteration $t = 1, 2, 3, \cdots$, pick a misclassified point from

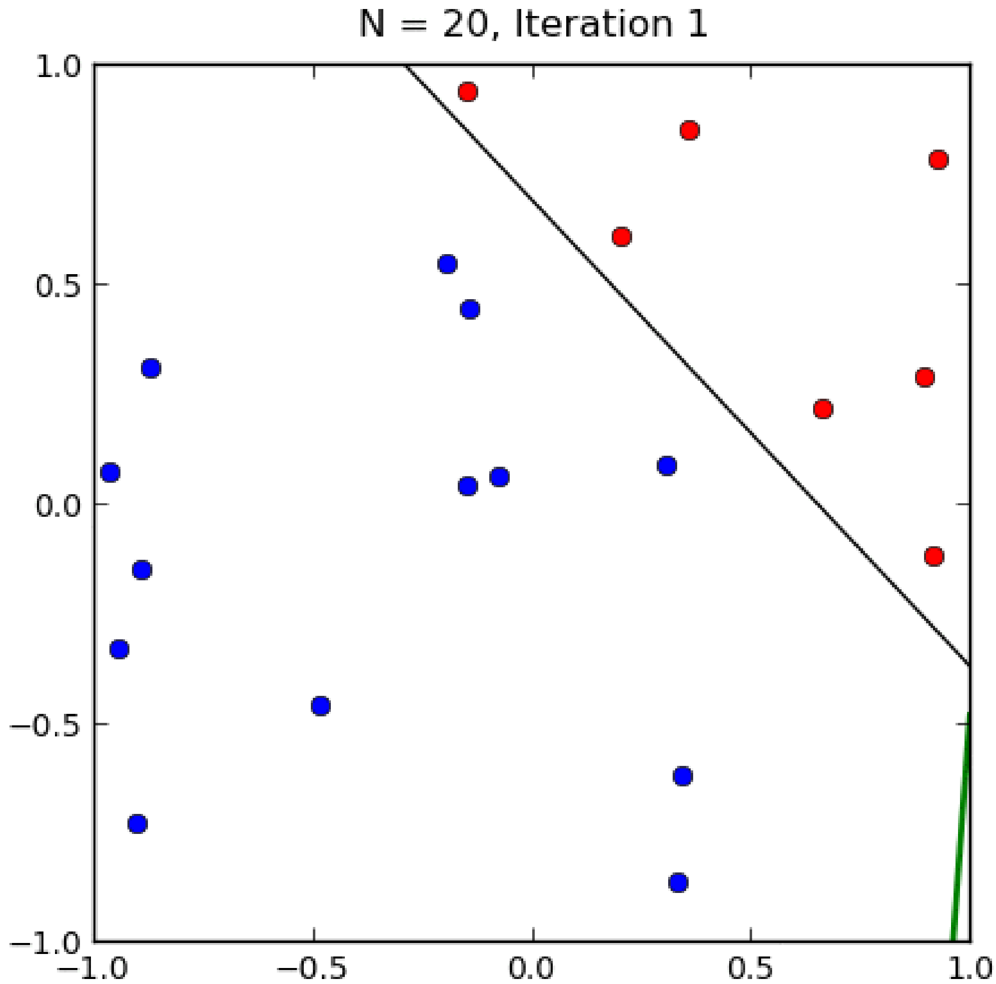
$$(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), \cdots, (\mathbf{x}_N, y_N)$$

and run a PLA iteration on it.

- That's it!

기계학습 기초 (학습 문제)

- Example of PLA



기계학습 기초 (학습 문제)

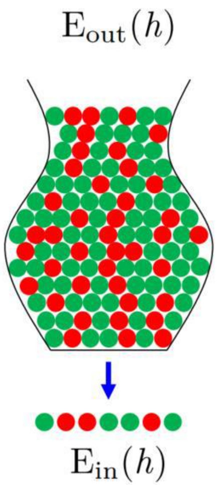
Error and noise

- Error (=cost, objective, risk): quantifies how far we are from the target
  - What does "$h \approx f$" mean?
  - choice of an error measure affects outcome of learning
  - measure $E(h, f)$ from pointwise to overall
    - usually defined error on individual input points (pointwise definition): $e(h(x), f(x))$
    - examples
      squared error: $e\,(h(\mathbf{x}), f(\mathbf{x})) = (\, h(\mathbf{x}) - f(\mathbf{x})\,)^2$
      binary error: $e\,(h(\mathbf{x}), f(\mathbf{x})) = [\, h(\mathbf{x}) \neq f(\mathbf{x})\,]$
    - (overall error) then average over pointwise errors $e(h(x), f(x))$

- Noise: about the nature of the target function
  - the part of y we cannot model
  - makes output of f target distribution determined by the input

기계학습 기초 (학습 문제)

- Feasibility of learning
  - Target f
    - Unknown
      - we cannot learn f deterministically
      - but we cam learn f in a probabilistic sense $E_{out} \approx E_{in}$ (generalization capability)
    - Probably, approximately correct learning



- Ultimate goal of learning
  - Learn g such that $g \approx f$
  - which means making $E_{out} \approx 0$ (good learning)

기계학습 기초 (학습 문제)

- Learning for $E_{out} \approx 0$ split into two questions

  Q1. can we make sure that $E_{out} \approx E_{in}$?

  Q2. can we make $E_{in} \approx 0$?

  - answering yes to both $\Rightarrow E_{out} \approx E_{in} \approx 0$

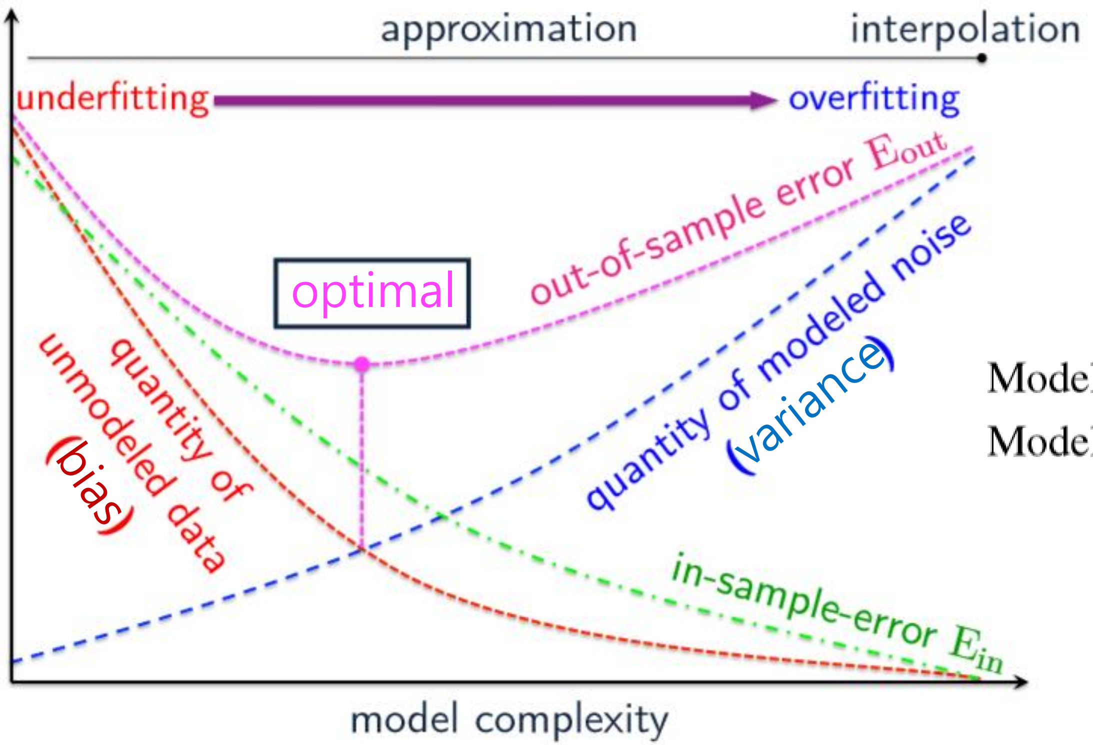
  - Answer to Q1
    - theoretical
    - better for simpler models
    - if fails ⇒ overfitting ⇒ need regularization

  - Answer to Q2
    - more practical: run A on training data D
    - better for more complex models
    - if fails ⇒ underfitting ⇒ need better optimizer

기계학습 기초 (학습 문제)

- Trade-off
  - true for any machine learning system
  - approximation-generalization trade-off

$\underbrace{\hspace{4em}}_{E_{in} \approx 0,}$ bias $\quad \underbrace{\hspace{4em}}_{E_{out} \approx E_{in},}$ variance



Model complexity ↑ $\quad E_{in}$ ↓

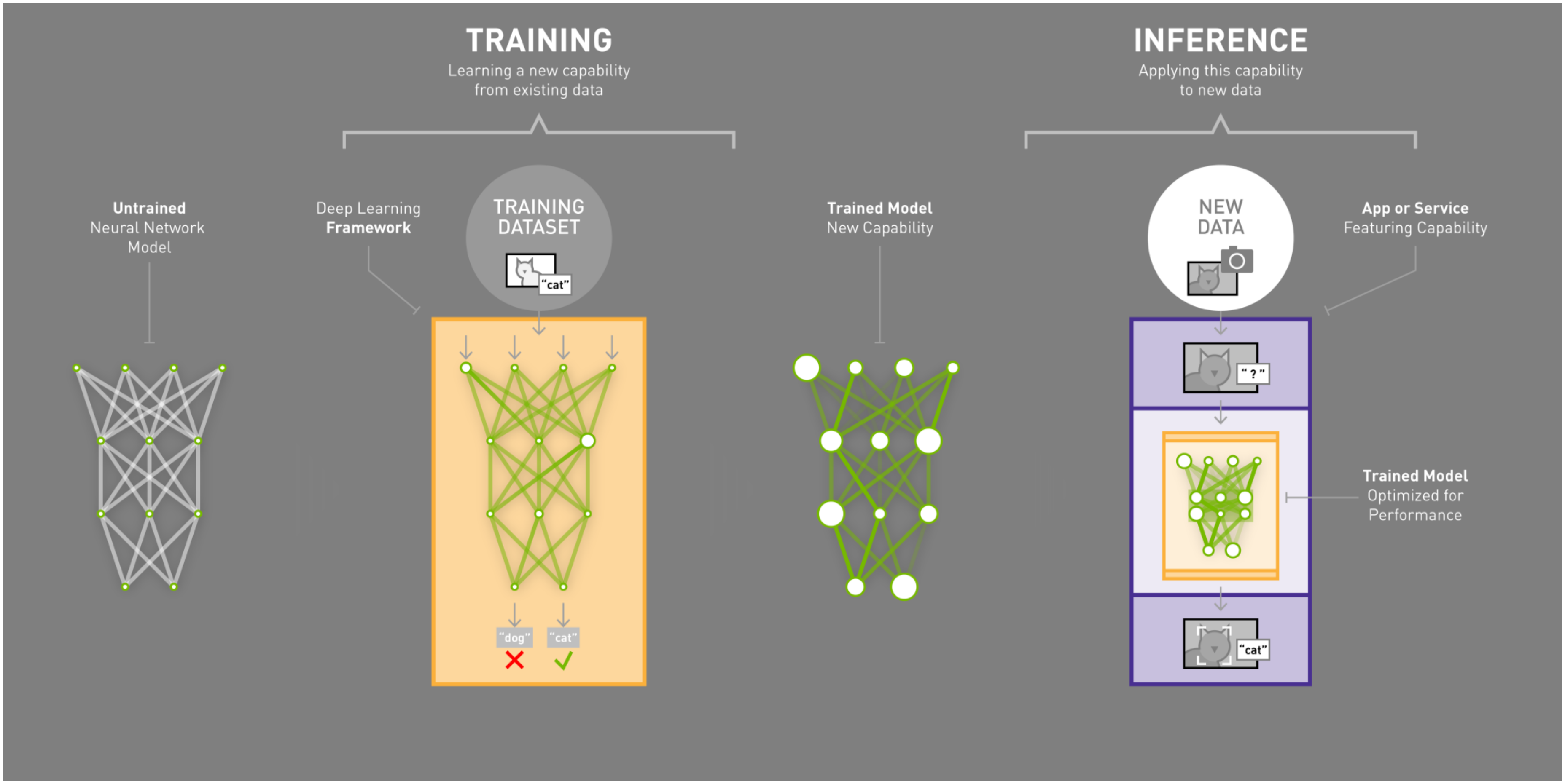
Model complexity ↑ $\quad E_{in} - E_{out}$ ↑

기계학습 기초 (학습 문제)

- Strategies in modern machine learning
  - selecting large capacity models (e.g., deep neural networks)
  - then, applying regularizations (e.g., weight decay, …)

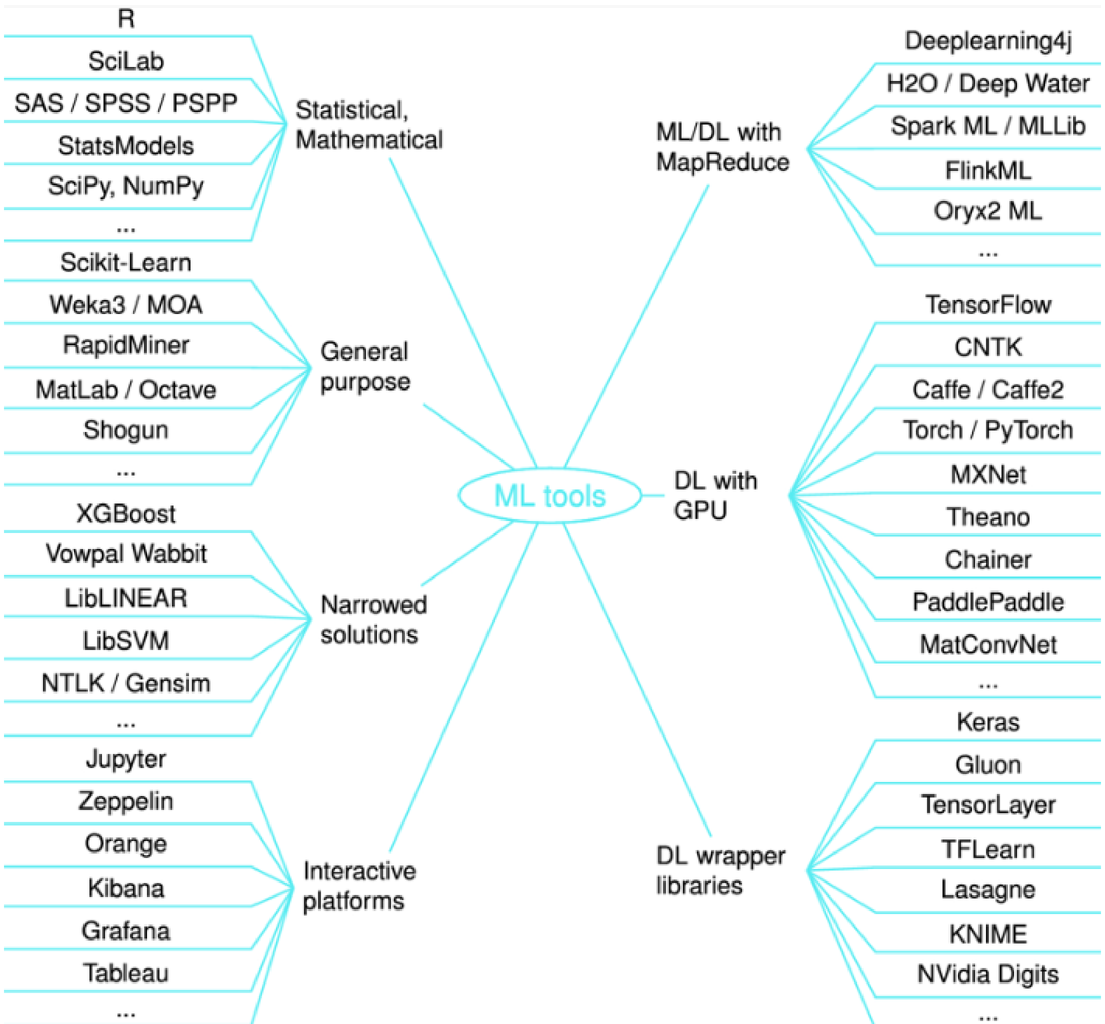
기계학습 기초

- 기계학습 절차: 훈련training 과 추론inference



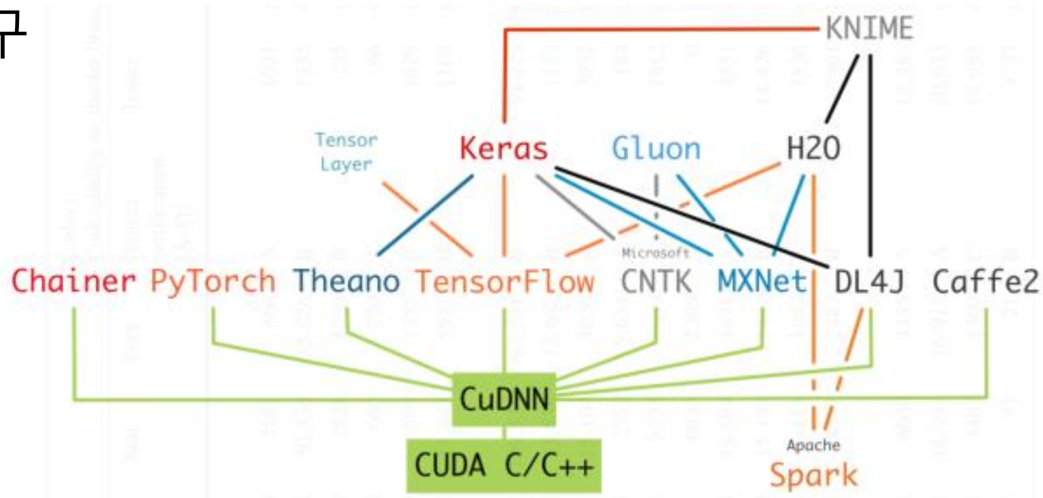
기계학습 기초

■ 기계학습 도구



기계학습 기초

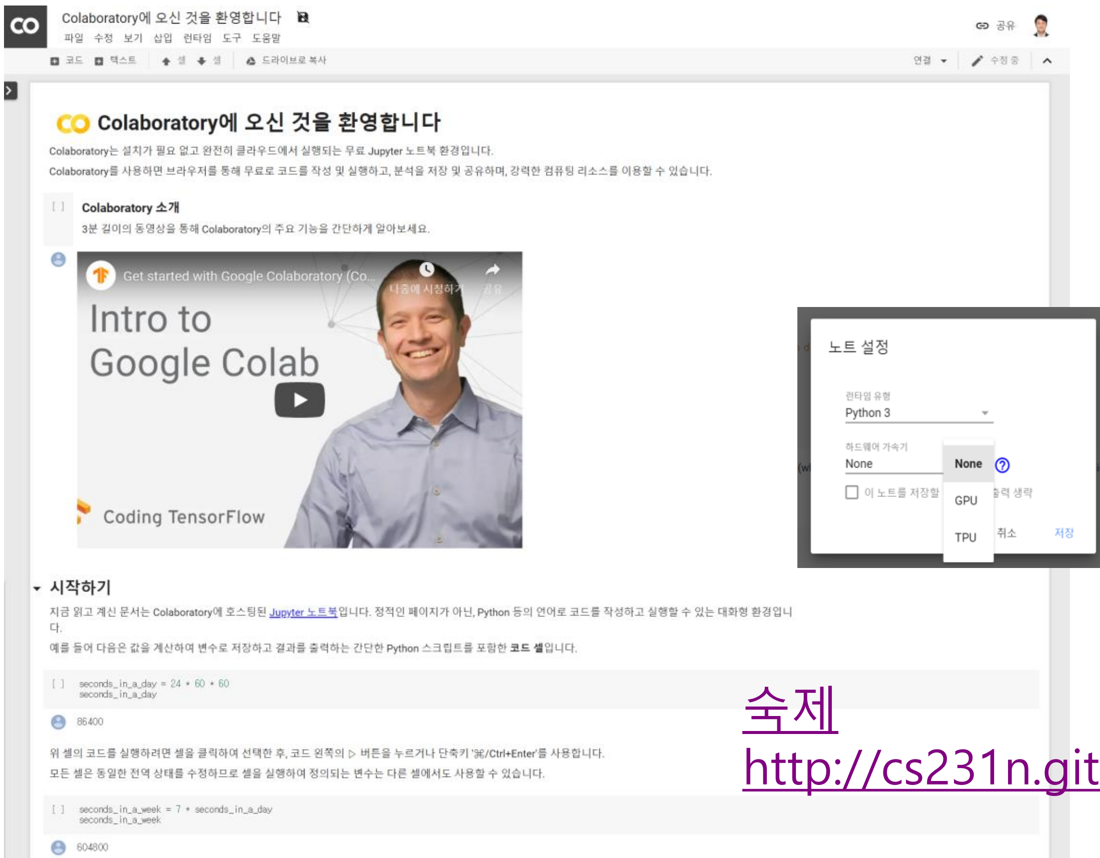
- 심층학습 도구



Tool	Licence	Written in	Computation graph	Interface	Popularity	Usage	Creator (notes)
TensorFlow (Numerical framework)	Open source, Apache 2.0	C++, Python	Static with small support for dynamic graph	Python, C++[a], Java[a], Go[a]	Very High Growing very fast	Academic Industrial	– **Google**
Keras (Library)	Open source, MIT	Python	Static	Python Wrapper for TensorFlow, CNTK, DL4J, MXNet, Theano	High Growing very fast	Academic Industrial	F. Chollet
CNTK (Framework)	Open source, Microsoft permissive license	C++	Static	Python, C++, BrainScript, ONNX	Medium Growing fast	Academic Industrial Limited mobile solution	– **Microsoft**
Caffe (Framework)	Open source, BSD 2-clause	C++	Static	C++, Python, MatLab	High Growing fast	Academic Industrial	Y. Jia **BAIR**
Caffe2 (Framework)	Open source, Apache 2.0	C++	Static	C++, Python, ONNX	Medium-low Growing fast	Academic Industrial Mobile solution	Y. Jia **Facebook**
Torch (Framework)	Open source, BSD	C++, Lua	Static	C, C++, LuaJIT, Lua, OpenCL	Medium–low Growing low	Academic Industrial	R. Collobert, K. Kavukcuoglu, C. Farabet
PyTorch (Library)	Open source, BSD	Python, C	Dynamic	Python, ONNX	Medium Growing very fast	Academic Industrial	A. Paszke, S. Gross, S. Chintala, G. Chanan
MXNet (Framework)	Open source, Apache 2.0	C++	Dynamic dependency scheduler	C++, Python, Julia, MatLab, Go, R, Scala, Perl, ONNX	Medium Growing fast	Academic Industrial	– **Apache**
Chainer (Framework)	Open source, Owners permissive license	Python	Dynamic	Python	Low Growing low	Academic Industrial	– **Preferred Networks**
Theano (Numerical framework)	Open source, BSD	Python	Static	Python	Medium-low Growing low	Academic Industrial	Y. Bengio **University of Montreal**

참고

- Google Colab
  - 심층학습 또는 기계학습 모델을 실행할 수 있는 Google의 무료 클라우드 서비스 (12시간)
  - https://colab.research.google.com/




```
import tensorflow as tf
device_name = tf.test.gpu_device_name()
if device_name != '/device:GPU:0':
  raise SystemError('GPU device not found')
print('Found GPU at: {}'.format(device_name))
```

```
Found GPU at: /device:GPU:0
```

숙제
http://cs231n.github.io/python-numpy-tutorial/

Statistics

- 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

기계학습과 통계

comparison:

- statistics
  - focuses on *idealized models* 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 *data analysis* than on prediction
  - DBs are usually huge $\Rightarrow$ computational issues critical in DM

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)