1. Machine Learning and MLP

Donghwan Chi*

Artificial Intelligence in Korea University(AIKU)

Department of Computer Science and Engineering, Korea University

Ice Breaking

반갑습니다 🤗

D2D - Overview

- 수동적인 학습 ❌
- 동료와 협업, 토론 적극 권장 ⭕
  - 학점 안 줍니다❗ / 경쟁하는 수업 아닙니다‼️
- AIKU 공식 : @aiku._.official
- 지금 말하고 있는 사람 : @zheedong
- 도움이 될 만한 사이트
- ⚠️ 모르는거 있으면 꼭 물어봐 주세요!! ⚠️

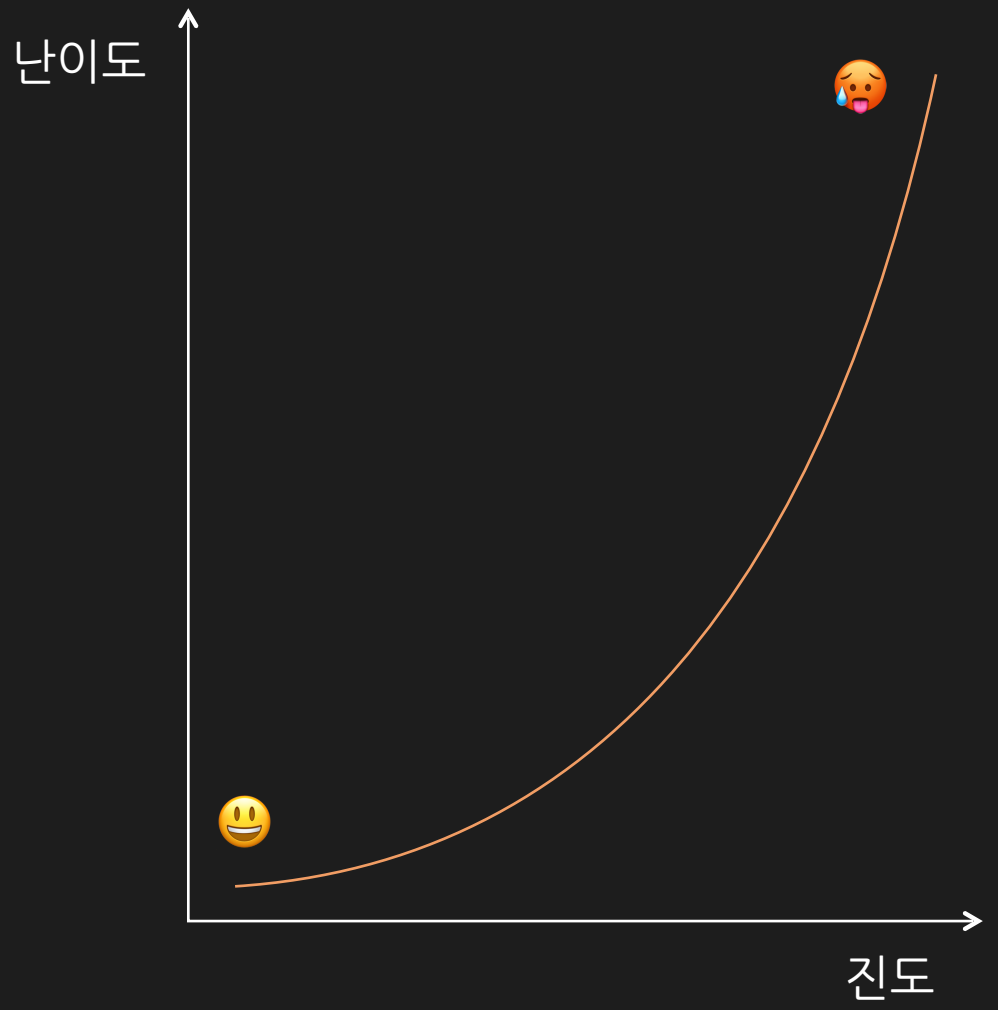
D2D - Details

- 과제
  - Discussion 과제
    - 매 수업마다 하나. Notion에 기입
  - Coding 과제
    - 매 주차에 하나. Colab Notebook 제출.
- 승급 기준
  - 출석과 Discussion 과제 각각 80% 이상
  - 프로그래밍 과제 중 과제 1, 2를 필수 과제로 하되, 프로그래밍 과제 5개 중
  - (1) 2개 미만 탈회
  - (2) 4개 미만 주니어
  - (3) 4개 이상 승급

Let’s start!

2023년 2학기
DeepIntoDeep 커리큘럼

회차	날짜	주제	강사	과제	후보
1	2023.07.11	Machine Learning and MLP	지동환	과제 1	지동환, 김민성, 김성찬, 배민성, etc.
2	2023.07.13	Convolutional Neural Networks	지동환		
3	2023.07.18	Recurrent Neural Networks	김민성	과제 2	
4	2023.07.20	Transformers	김민성		
5	2023.07.25		배민성		
6	2023.07.27	Techniques for Training Neural Networks	김성찬	없음	
7	2023.08.01	CV 집중 탐색 I (Object Detection & Segmentation)		과제 3	배민성, 김승현
8	2023.08.03				
9	2023.08.08	NLP 집중 탐색 I (QA, Summarization, Retrieval)	박수빈	과제 4	조혜진, 박수빈
10	2023.08.10		박수빈		
11	2023.08.15	CV 집중 탐색 II (Generative Models, 3D Vision)		과제 5 (자유도 높음)	배민성, 김승현
12	2023.08.17				
13	2023.08.22	NLP 집중 탐색 II (Generative Models in NLP & LLM)	조혜진		조혜진, 박수빈
14	2023.08.24		조혜진		

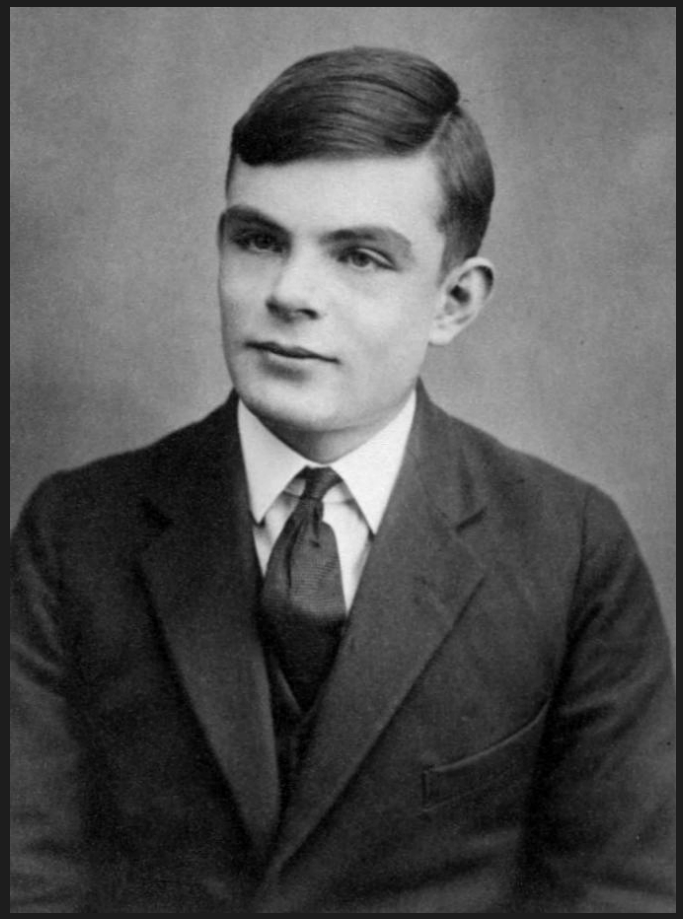


Contents

- Machine Learning
  - Key Components
  - “Good” Model
  - Kinds of Machine Learning
  - Linear Neural Networks for regression
- Multi Layer Perceptron
  - Why “Deep” Learning?

Machine Learning

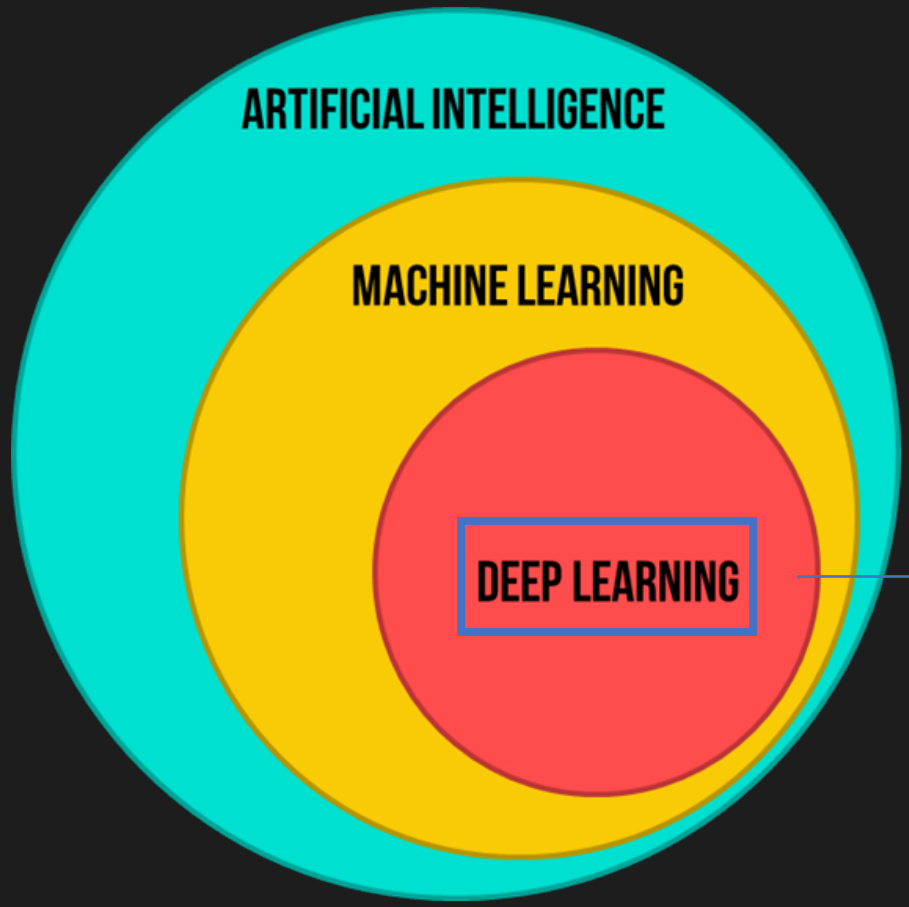
What is “Artificial Intelligence”?



출처 : Wikipedia

"인공지능의 발명이란 자동차에서 바퀴를 떼어낸 뒤 그 자리에 발을 달기 위해 고심하는 것이다."

AI, ML, DL



우리의 주요 관심사

출처 : https://infyskill.in/insights/

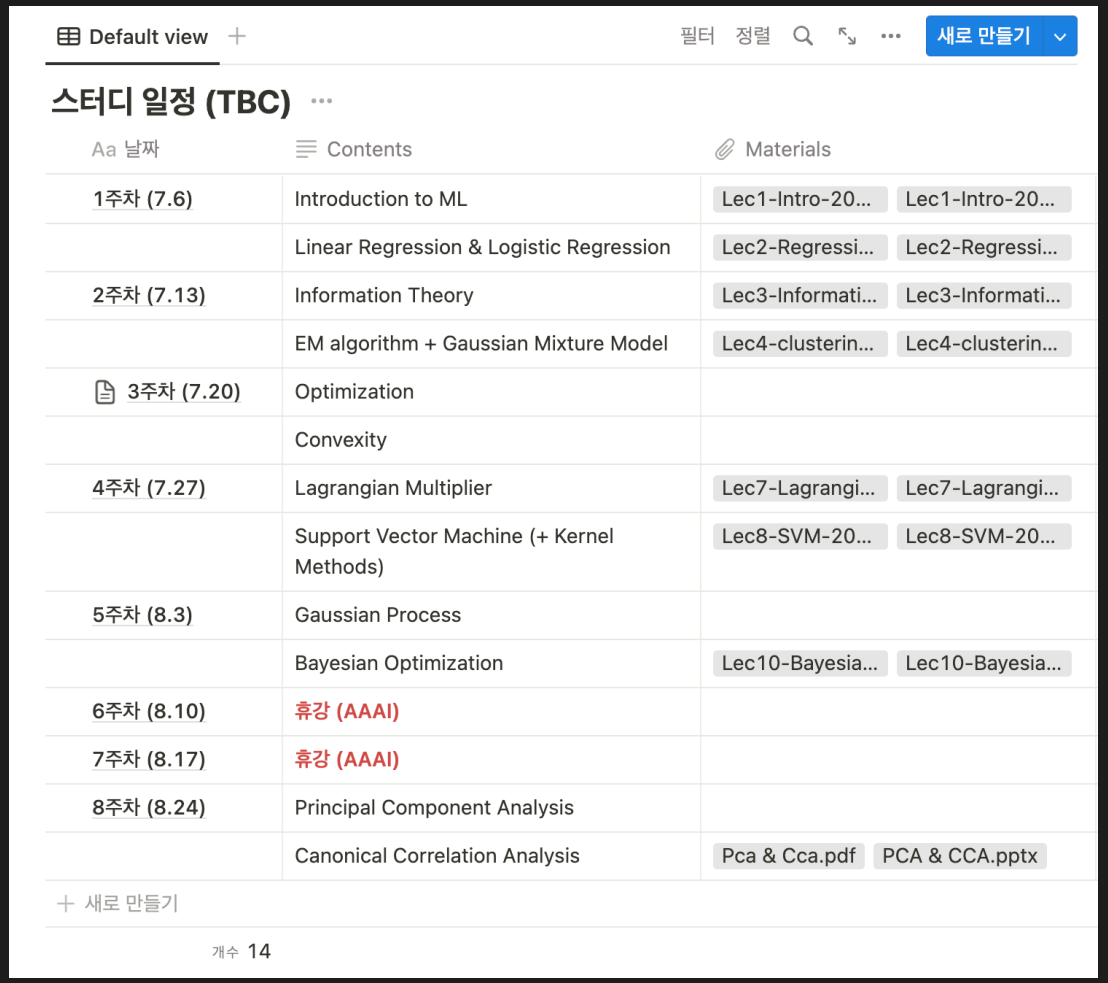
Do we need “Machine Learning”?

- 딥러닝 그거 딸깍 딸깍 하면 되는거 아님?

Certainly “YES”.

- 딥러닝 그거 딸깍 딸깍 하면 되는거 아님?
  - (쳐) 맞습니다.

Machine Learning



Default view + 필터 정렬 새로 만들기

스터디 일정 (TBC)

Aa 날짜	Contents	Materials	
1주차 (7.6)	Introduction to ML	Lec1-Intro-20...	Lec1-Intro-20...
	Linear Regression & Logistic Regression	Lec2-Regressi...	Lec2-Regressi...
2주차 (7.13)	Information Theory	Lec3-Informati...	Lec3-Informati...
	EM algorithm + Gaussian Mixture Model	Lec4-clusterin...	Lec4-clusterin...
3주차 (7.20)	Optimization		
	Convexity		
4주차 (7.27)	Lagrangian Multiplier	Lec7-Lagrangi...	Lec7-Lagrangi...
	Support Vector Machine (+ Kernel Methods)	Lec8-SVM-20...	Lec8-SVM-20...
5주차 (8.3)	Gaussian Process		
	Bayesian Optimization	Lec10-Bayesia...	Lec10-Bayesia...
6주차 (8.10)	휴강 (AAAI)		
7주차 (8.17)	휴강 (AAAI)		
8주차 (8.24)	Principal Component Analysis		
	Canonical Correlation Analysis	Pca & Cca.pdf	PCA & CCA.pptx

+ 새로 만들기

개수 14

출처 : MLV Notion

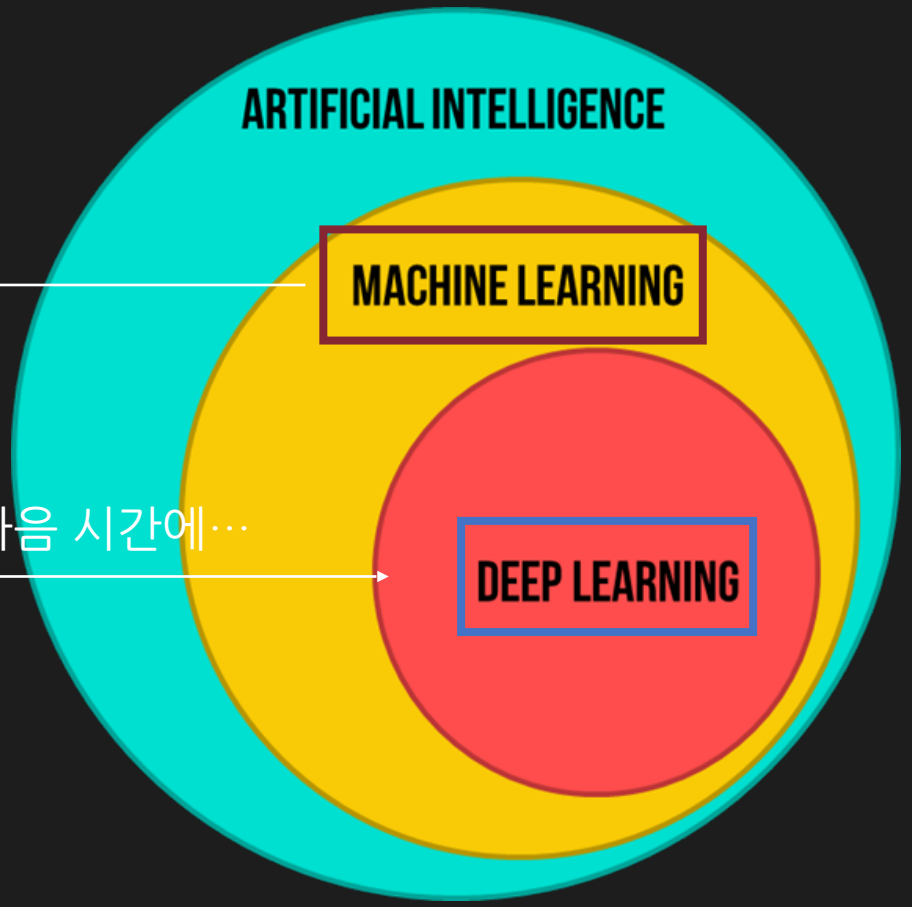
- 수학적인 근거
  - 선형대수 + 확률 + 최적화
- 딥러닝
  - 강력한 GPU + 새로운 방법론
- 머신 러닝 몰라도 되나요?
  - 매우 강력한 Abstraction
  - 기본적인 이해 돕기
  - 더 깊어지고 싶다면?

Overall Guide Line

오늘 이야기 해 볼 것

Limitation 파악

다음 시간에…



출처 : https://infyskill.in/insights/

Machine Learning - Key Components

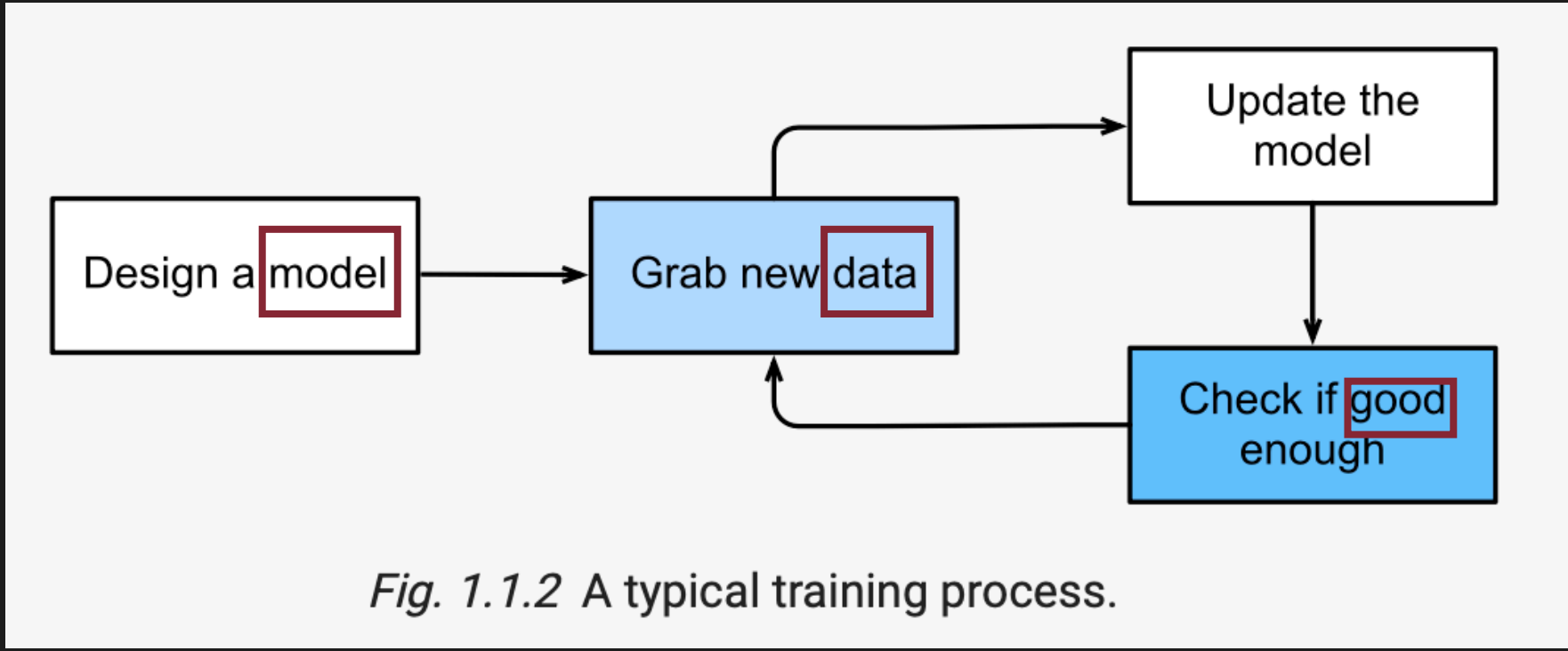


Fig. 1.1.2 A typical training process.

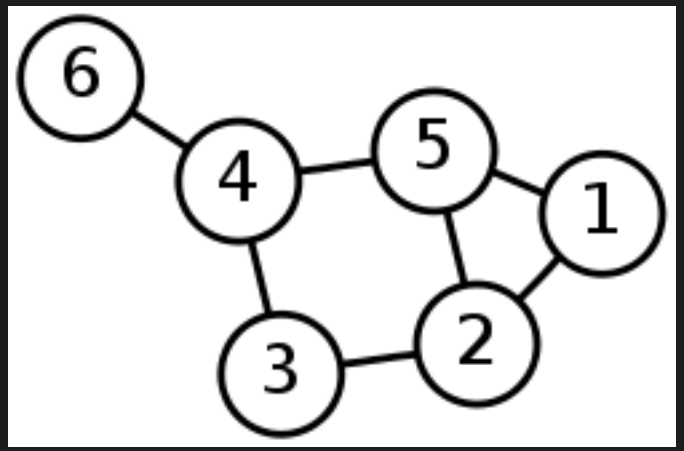
출처 : Dive Into Deep Learning

1. Randomly initialized 된 아무 것도 할 수 없는 model에서 시작
2. Data의 일부를 넣어 줌
3. Model을 조정해 주어진 예시에서 잘 하도록 바꿈
4. Model이 Awesome해 질 때까지 Step 2, 3을 반복

Key Components - Data

Image

Text

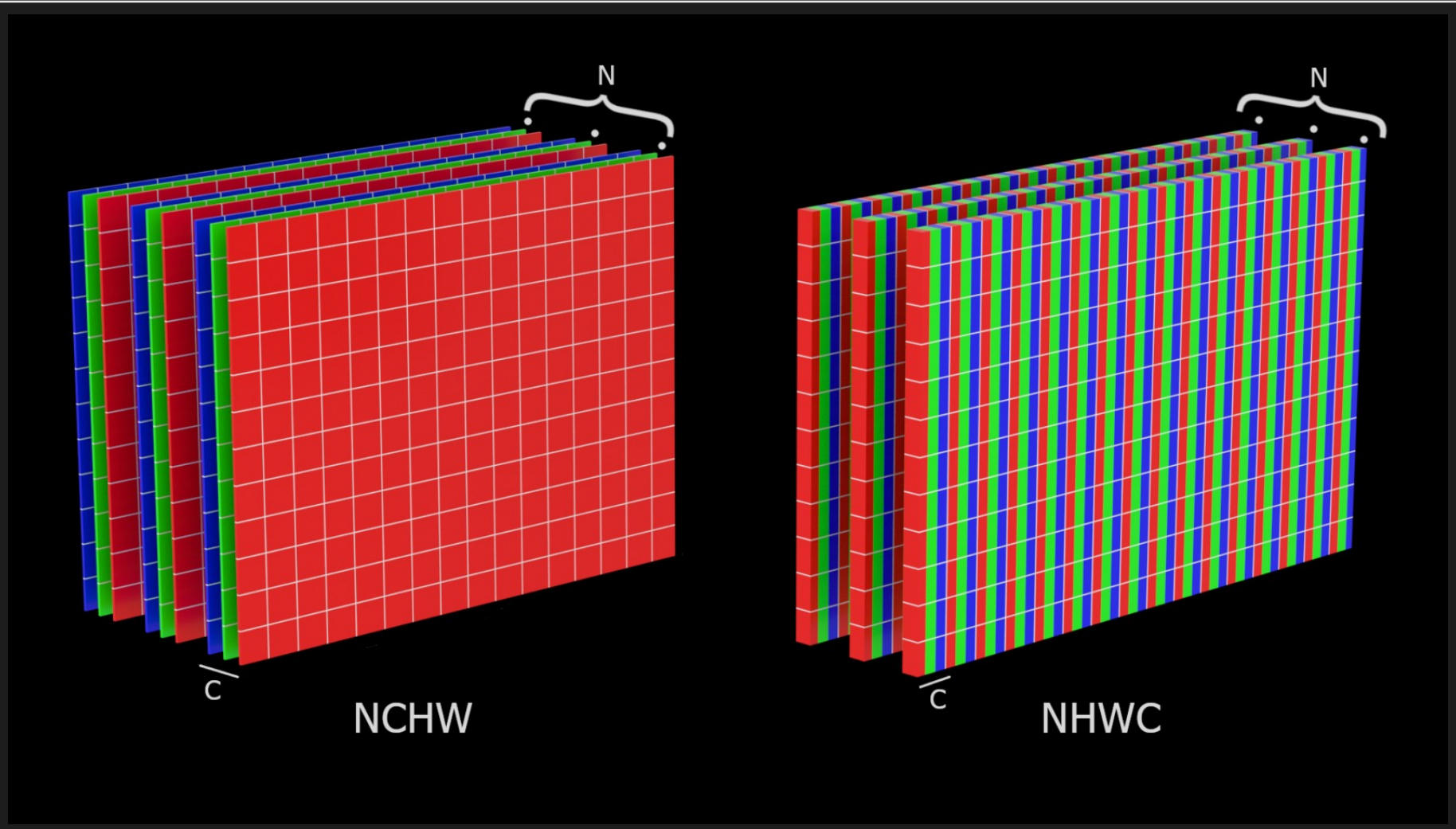


Graph

Sound

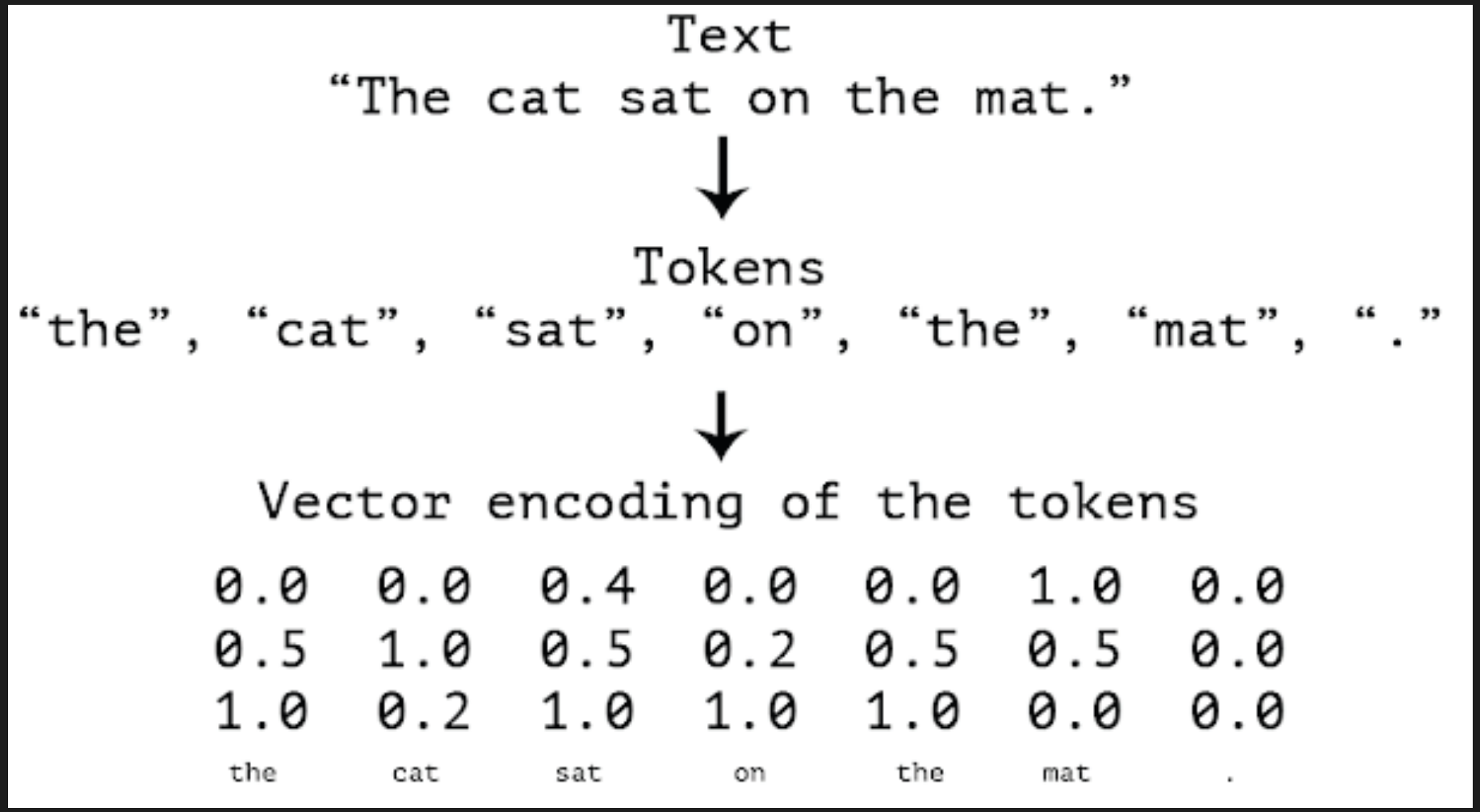
출처 : 구글 이미지

Key Components - Data (Image)



출처 : https://maxliani.wordpress.com/2023/03/24/dnnd-2-tensors-and-convolution/

Key Components - Data (Sentences)



출처 : https://freecontent.manning.com/deep-learning-for-text/

Key Components - Is data important?

출처 : https://www.linkedin.com/pulse/meta-heats-up-tech-giants-fight-launch-llama-ai-language-andre-havro/

Scaling Laws for Neural Language Models

Jared Kaplan *
Johns Hopkins University, OpenAI
jaredk@jhu.edu

Sam McCandlish*
OpenAI
sam@openai.com

Tom Henighan
OpenAI
henighan@openai.com

Tom B. Brown
OpenAI
tom@openai.com

Benjamin Chess
OpenAI
bchess@openai.com

Rewon Child
OpenAI
rewon@openai.com

Scott Gray
OpenAI
scott@openai.com

Alec Radford
OpenAI
alec@openai.com

Jeffrey Wu
OpenAI
jeffwu@openai.com

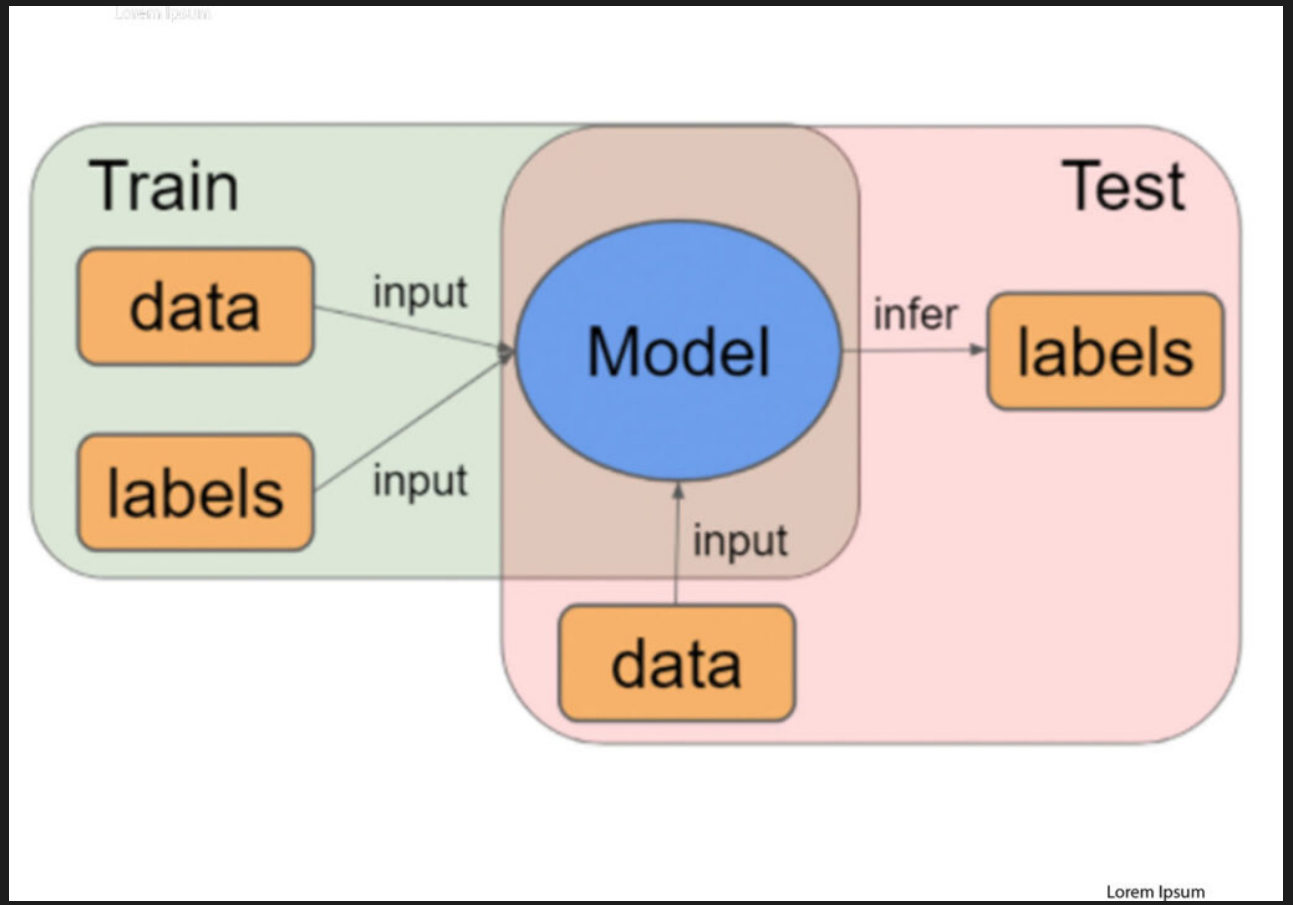
Dario Amodei
OpenAI
damodei@openai.com

Abstract

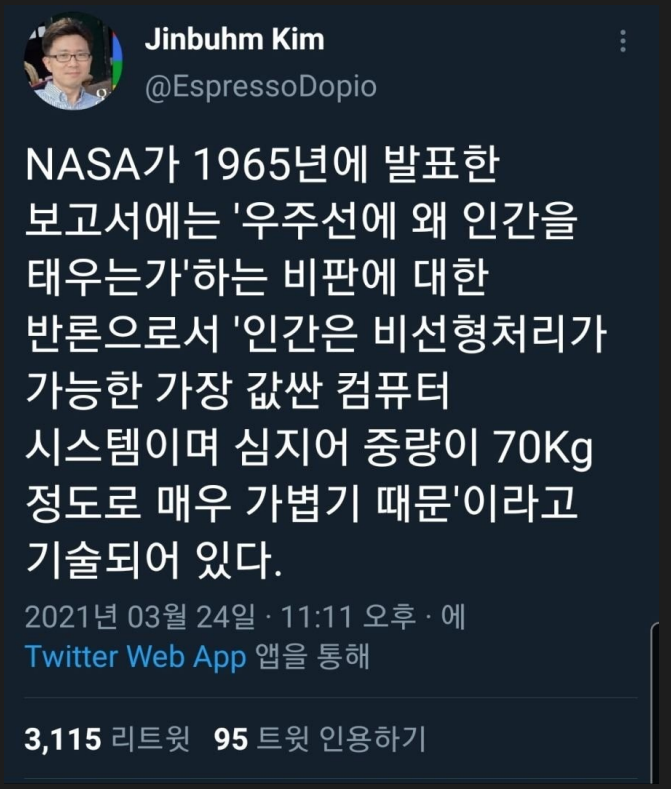
We study empirical scaling laws for language model performance on the cross-entropy loss. The loss scales as a power-law with model size, dataset size, and the amount of compute used for training, with some trends spanning more than seven orders of magnitude. Other architectural details such as network width or depth have minimal effects within a wide range. Simple equations govern the dependence of overfitting on model/dataset size and the dependence of training speed on model size. These relationships allow us to determine the optimal allocation of a fixed compute budget. Larger models are significantly more sample-efficient, such that optimally compute-efficient training involves training very large models on a relatively modest amount of data and stopping significantly before convergence.

Scaling Laws for Neural Language Models

Key Components - Models



출처 : https://www.mvision.ai/supervised-unsupervised-ml/



출처 : 고파스

Key Components - Objective Functions

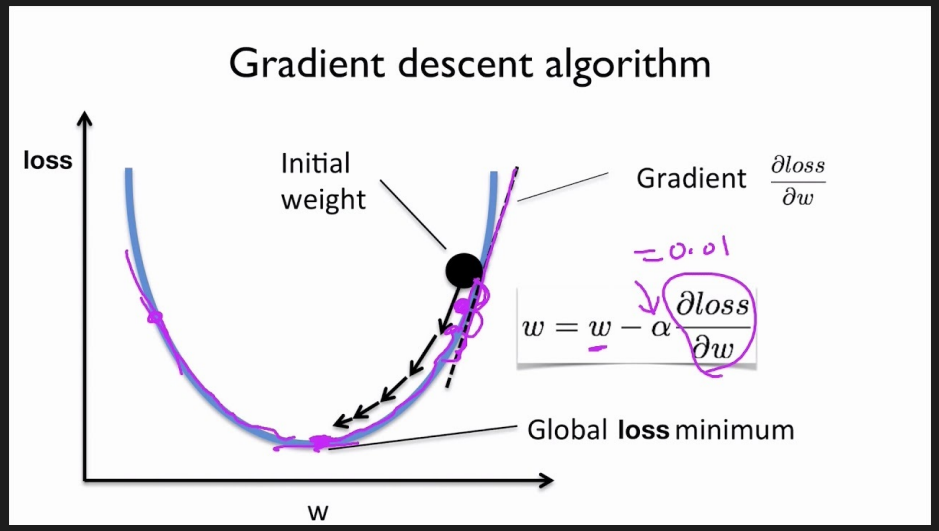
- 우리가 원하는 것 : Model의 예측 값이 실제 값과 비슷해 지는 것
  - 즉, 차이 (Loss)를 최소화 하고 싶다.
- Objective Function >= Cost Function >= Loss Function
  - Objective Function : 학습을 통해 최적화 시키고자 하는 목표
  - Cost Function : 모든 input data에 대한 오차
  - Loss Function : input의 예측 값과 실제 값 사이의 차이

Key Components - Loss Functions

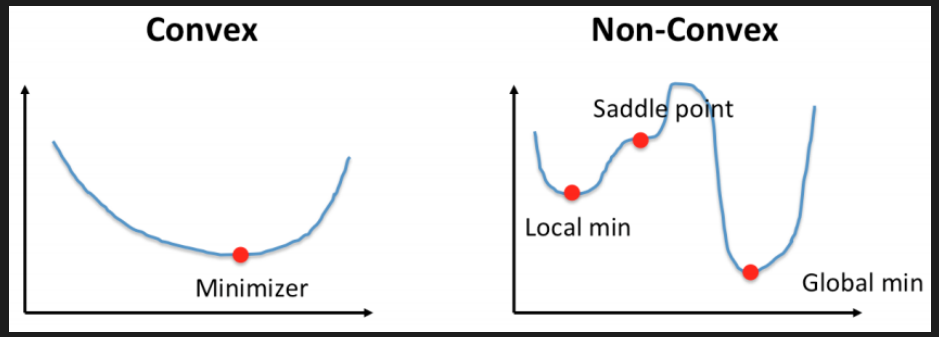
- Regression
  - MSE (Mean Squared Error)
  - MAE (Mean Absolute Error)
  - RMSE (Root Mean Squared Error)
- Classification
  - Cross-entropy loss
  - Binary Cross-entropy loss (Sigmoid + CE Loss)
  - Categorical Cross-entropy loss (Softmax + CE Loss)
- 기타 등등… 자습 추천!

Key Components - Optimization Algorithms

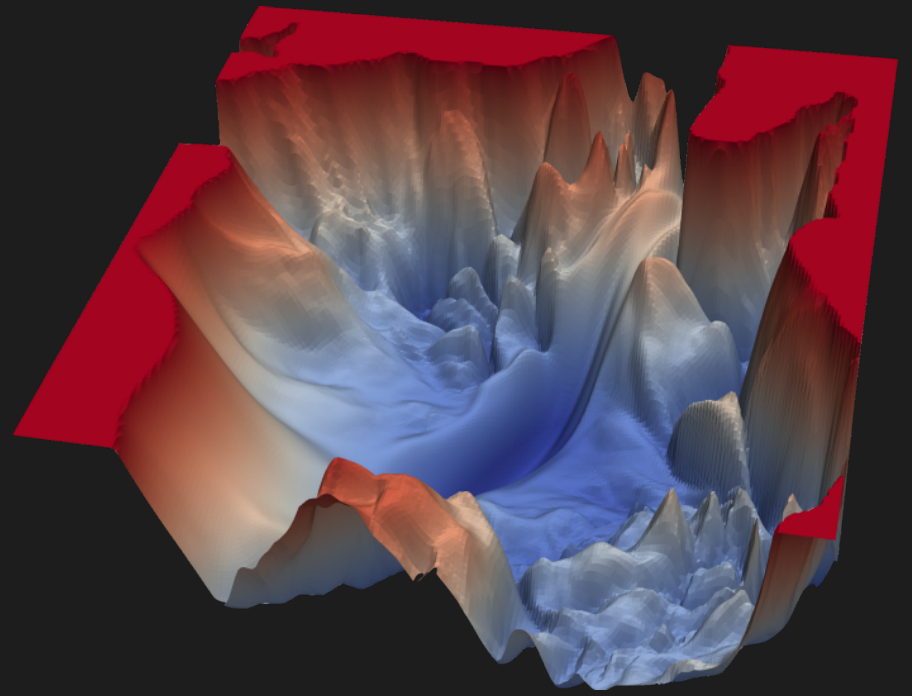
Gradient Descent



Non-Convexity

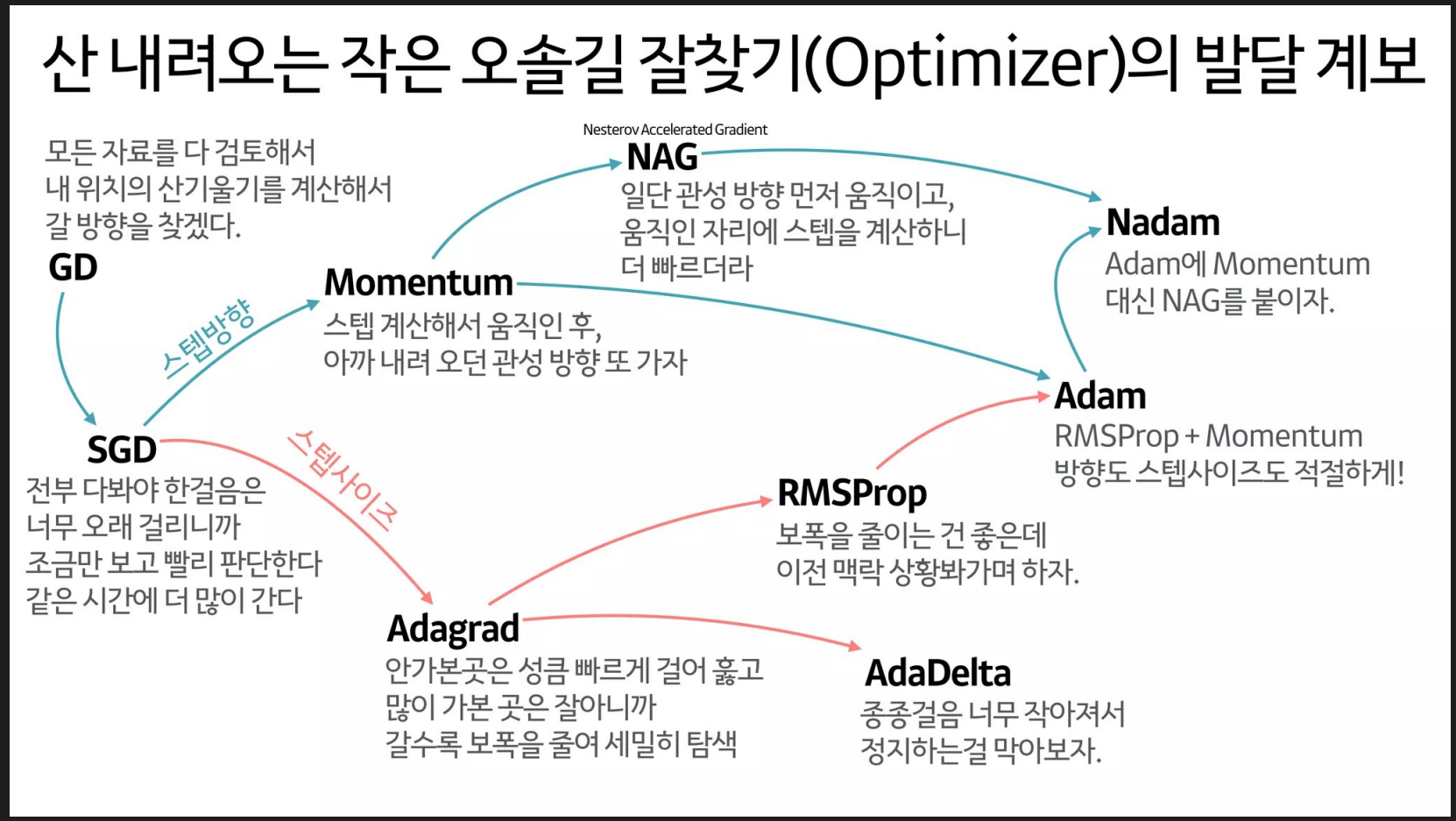


출처 : https://only-wanna.tistory.com/entry/목적-함수와-최적화



Visualizing the Loss Landscape of Neural Nets :
VGG56의 목적 함수

Key Components - Optimization Algorithms



출처 : https://www.slideshare.net/yongho/ss-79607172

Reading Material

- Loss
  - 손실함수 간략 정리
  - Loss Function
- Optimization
  - 최적화 알고리즘
  - https://hyunw.kim/blog/2017/11/01/Optimization.html

What is “Good” model?

- Machine Learning의 궁극적인 목표
  - Unseen data에 대해 좋은 ‘Predict’

출처 : https://datacadamia.com/data_mining/overfitting

Interviewer: What's your biggest strength?

Me: I'm an expert in machine learning.

Inteviewer: What's 6 + 10?

Me: Zero.

Interviewer: Nowhere near, it's 16.

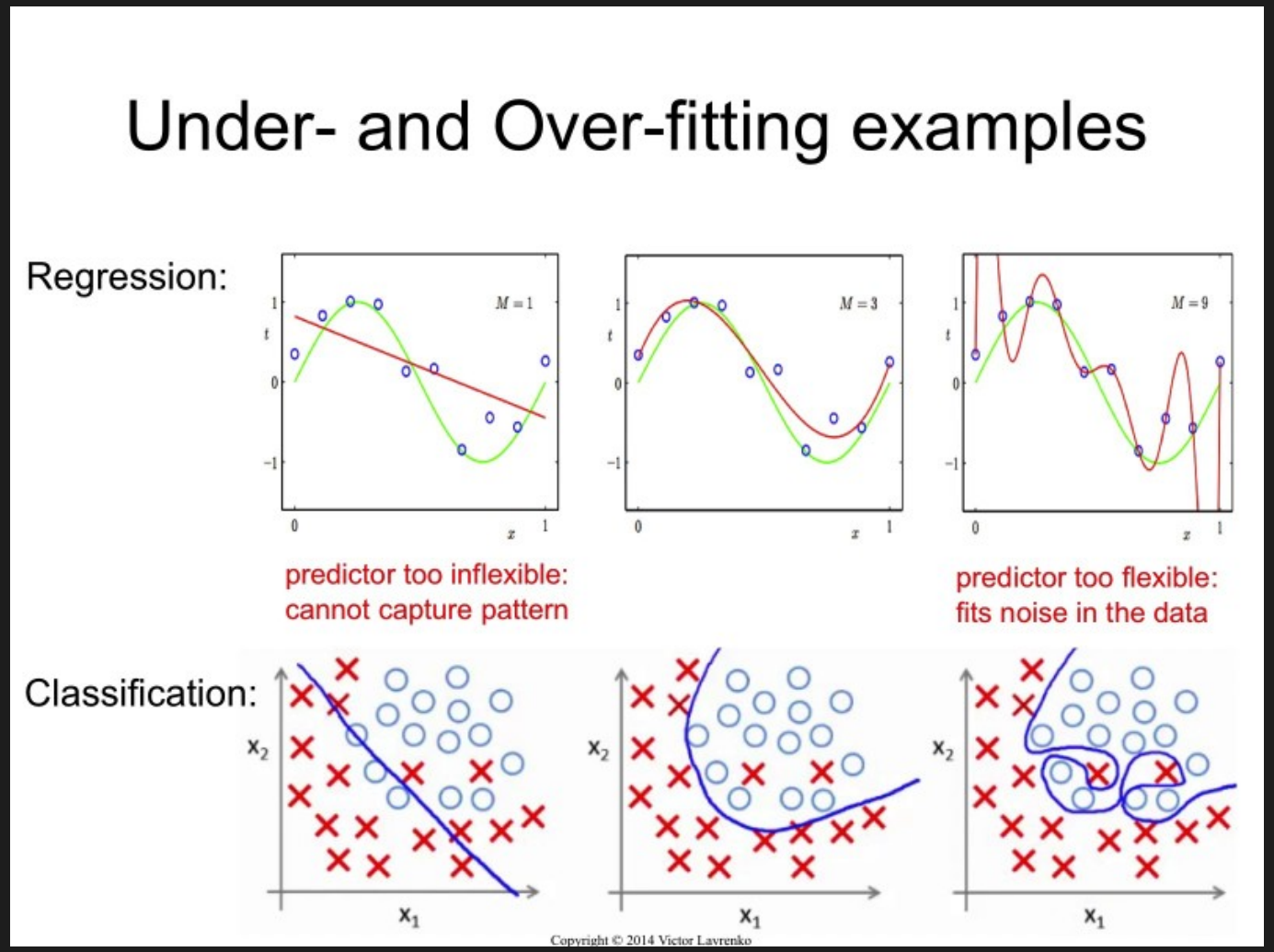
Me: It's 16.

Interviewer: Ok... What's 10 + 20?

Me: It's 16.

출처 : https://www.reddit.com/r/ProgrammerHumor/comments/8izngb/overfitting/

Overfitting, Underfitting



Data에는 Noise가 많이 포함 되어 있다!

출처 : https://m.blog.naver.com/qbxlvnf11/221324122821

Decomposition of MSE - bias and variance

Evaluating Estimator: Mean Squared Error

Definition (MSE). The mean square error (MSE) of an estimator $\hat{\theta}$ that estimate a parameter θ is

$$\mathrm{MSE}(\hat{\theta}) := \mathrm{E}\left((\hat{\theta} - \theta)^2\right)$$

Decomposition of MSE. The MSE of an estimator $\hat{\theta}$ for θ can be expressed as the sum of variance and bias;

MSE는 variance와 bias의 합이다!

$$\mathrm{MSE}(\hat{\theta}) = \underbrace{E\left((\hat{\theta} - E(\hat{\theta}))^2\right)}_{\text{variance}} + \underbrace{(E(\hat{\theta}) - \theta)^2}_{\text{bias}}$$

우리가 배워야 그치지 요구사항.

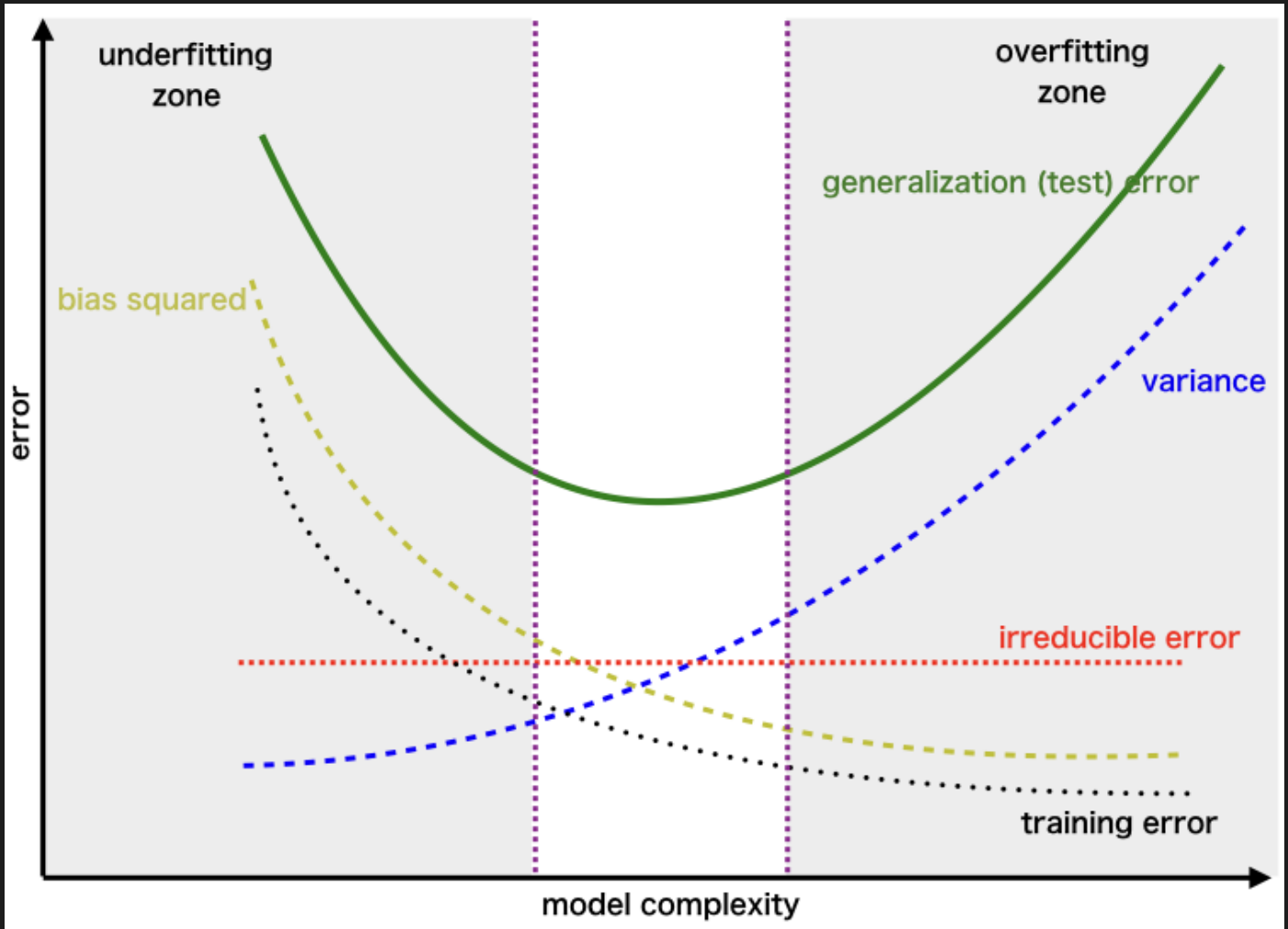
Proof. From the definition of MSE,

$$\begin{aligned}
\mathrm{MSE}(\hat{\theta}) &= E\left((\hat{\theta} - \theta)^2\right) = E((\hat{\theta} - E(\hat{\theta}) + E(\hat{\theta}) - \theta)^2) \\
&= E(\hat{\theta} - E(\hat{\theta})^2) + (E(\hat{\theta}) - \theta)^2 + 2E\left((\hat{\theta} - E(\hat{\theta}))(E(\hat{\theta}) - \theta)\right) \\
&= E(\hat{\theta} - E(\hat{\theta})^2) + (E(\hat{\theta}) - \theta)^2
\end{aligned}$$

Estimator Information Theory & Inference 13

출처 : '정보이론과추론학습' 강의자료

Bias-variance tradeoff



출처 : https://www.geeksforgeeks.org/ml-bias-variance-trade-off/

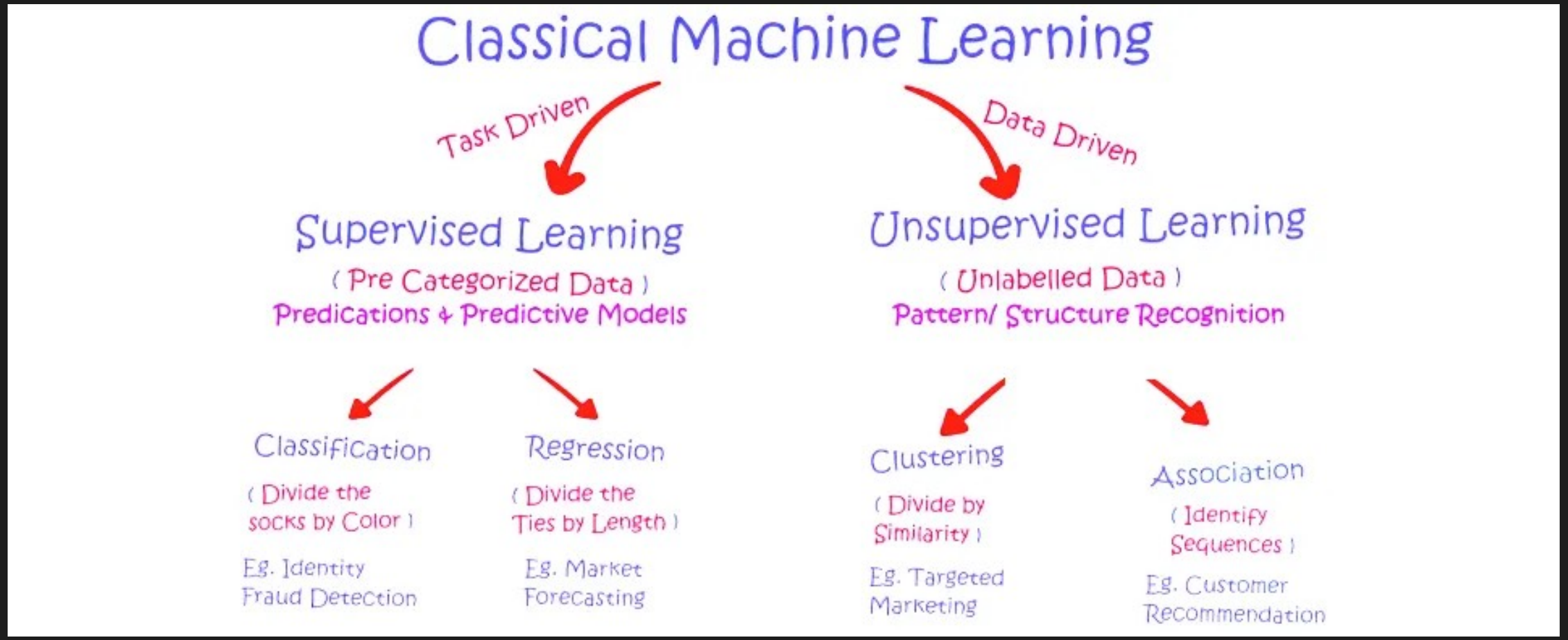
- Occam's Razor
- 'Sweet Spot'을 찾아라

Learning Techniques

- Regularization
  - L1 Regularization (Lasso regression)
  - L2 Regularization (Ridge regression)
- Weight Decay
- Early Stopping
- Dropout
- **Deep Learning스러운 해결책 : More Data!**

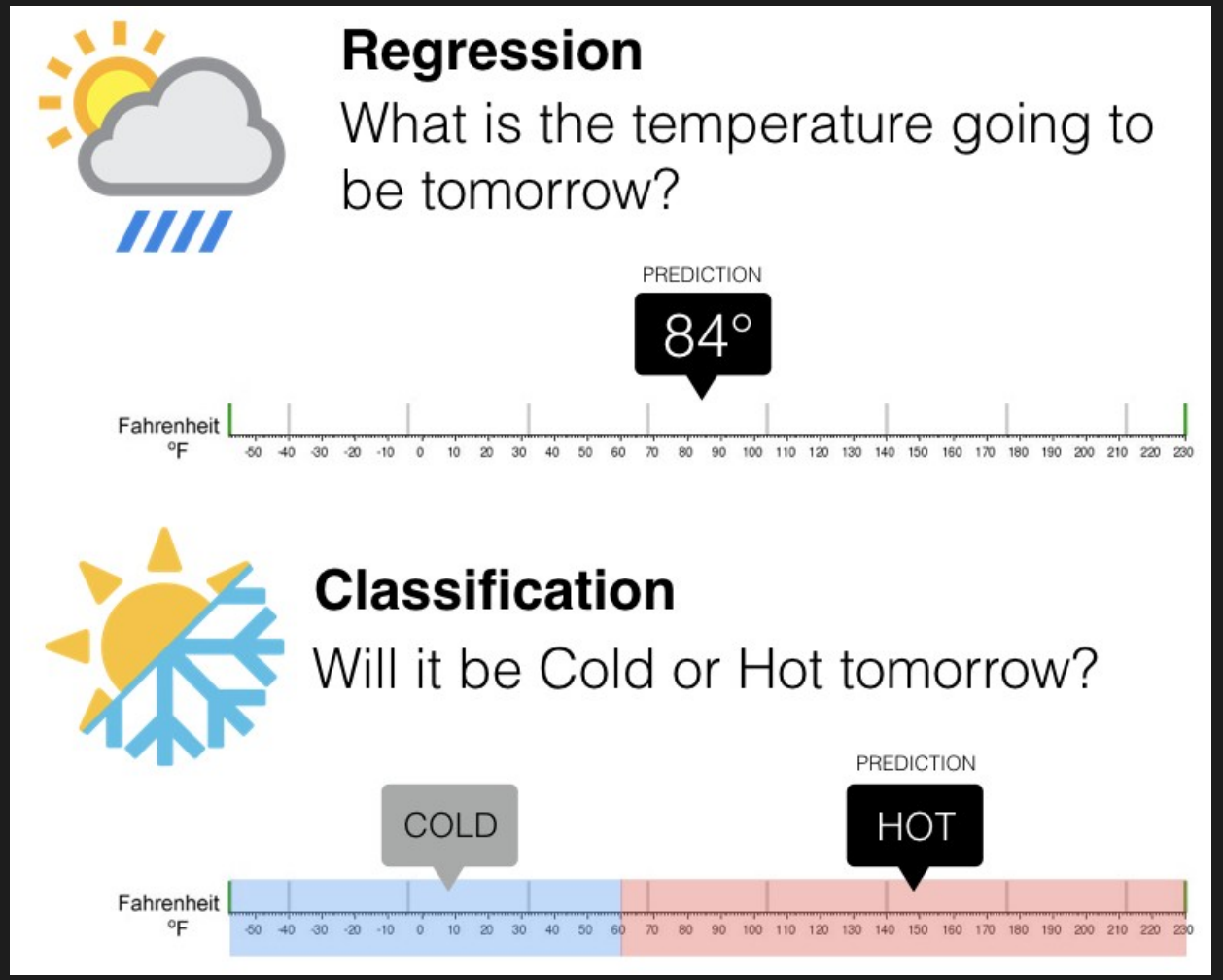
- 6강에서 더 자세히…

Kinds of Machine Learning Problems



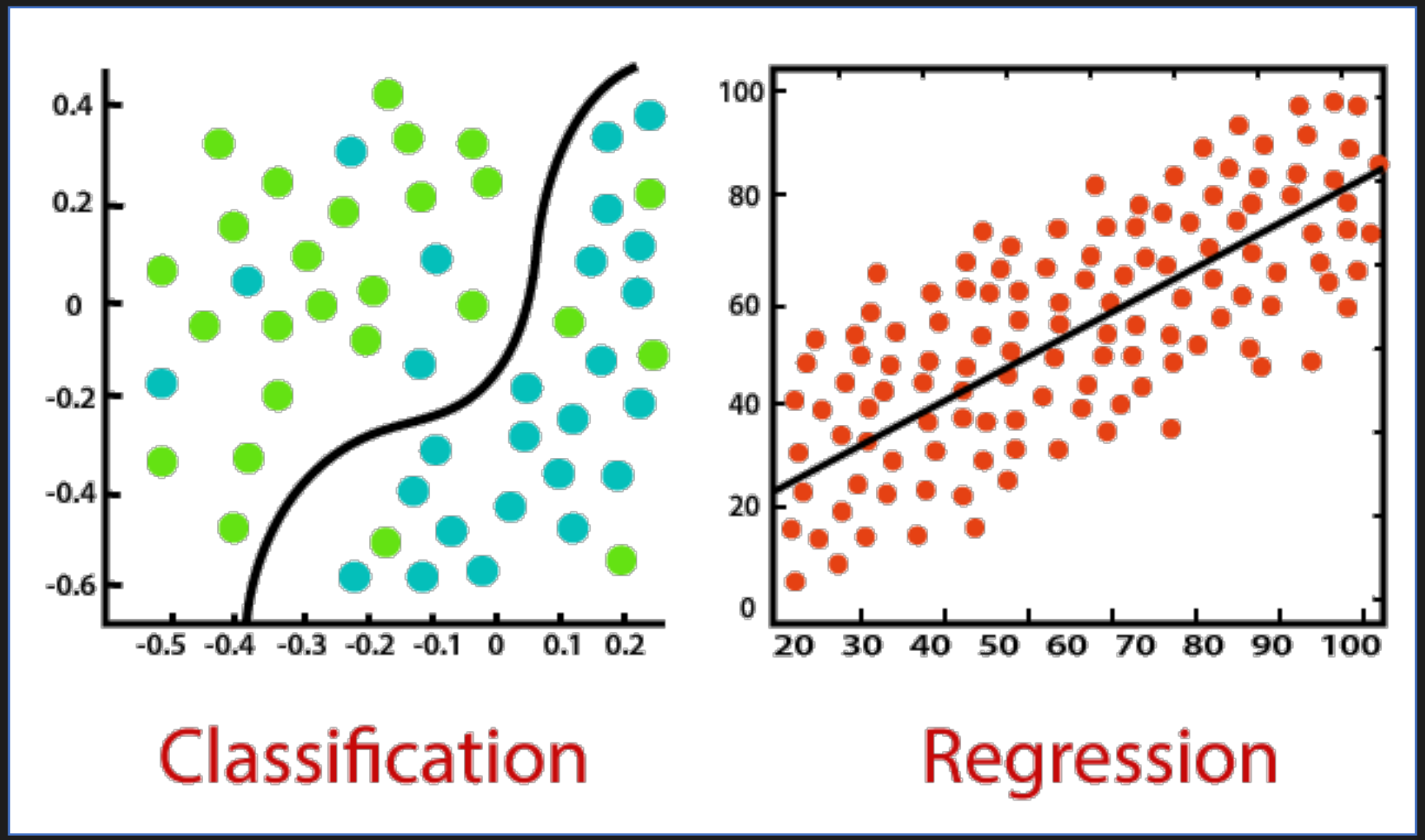
출처 : https://medium.datadriveninvestor.com/supervised-vs-unsupervised-machine-learning-732d49413986

Supervised Learning - Regression, Classification



출처 : https://www.springboard.com/blog/data-science/regression-vs-classification/

Supervised Learning - Regression, Classification



출처 : https://www.javatpoint.com/regression-vs-classification-in-machine-learning

Supervised Learning - Regression, Classification

Regression	Classification
input value (x)를 continuous output (y)로	input value (x)를 discrete output (y)로
best fit line을 찾기	decision boundary를 찾기
집 값 예측, 날씨 예측 등	스팸 분류, 암 진단 등

Quiz ! Regression vs Classification

- 뭐가 더 어려운 문제일까요?

learnersbucket.com

Knapsack Problem

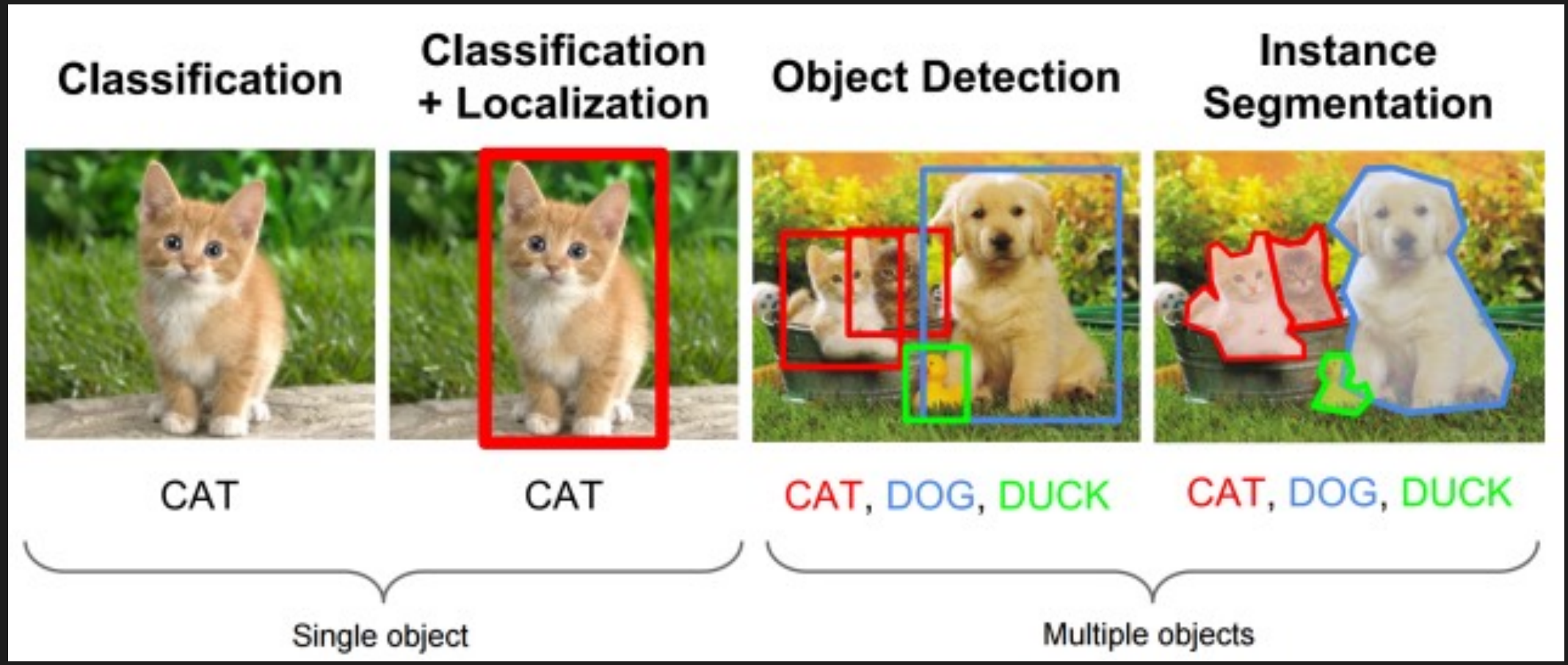
Given a set of items, each with a weight and a value, determine the number of each item to include in a collection so that the total weight is less than or equal to a given limit and the total value is as large as possible

Input:
values = [20, 5, 10, 40, 15, 25];
weights = [1, 2, 3, 8, 7, 4];
target = 10;

Output: 60 (1 (20) + 8(40) = 9 < 10)

출처 : https://learnersbucket.com/examples/algorithms/knapsack-problem-in-javascript/

Supervised Learning - Mixture



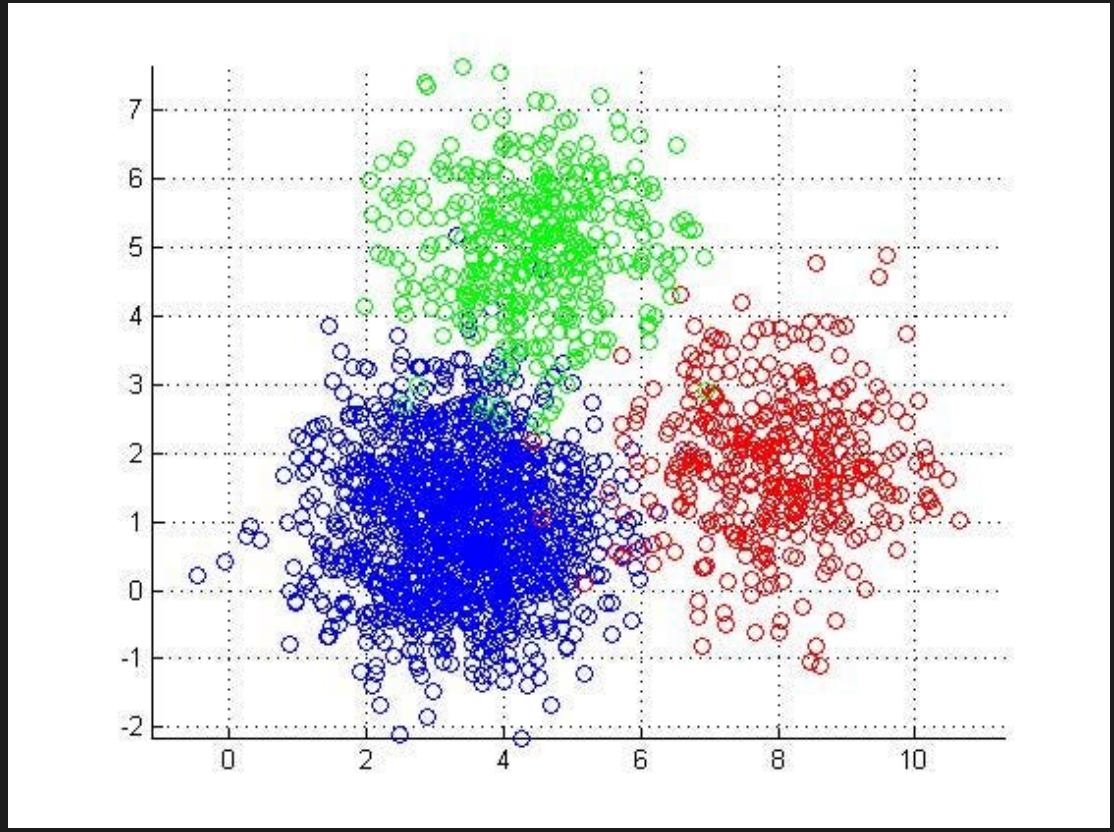
출처 : Standford University 2016 winter lectures CS231n Fei-Fei Li & Andrej Karpathy & Justin Johnson

Supervised Learning - etc

- Tagging
- Search
- Recommender Systems
- Sequence Learning
- 기타 등등…

Unsupervised Learning

- Label이 없는 Data라면?
  - K-means Clustering
- Self-Supervised Learning?
  - 일부만 Label이 존재한다면?
  - Trend in DL



출처 : https://medium.com/@yashlabhsetwar2001/k-means-is-one-of-the-most-popular-unsupervised-machine-learning-algorithms-used-for-solving-1b66f4f6abed

Linear Regression

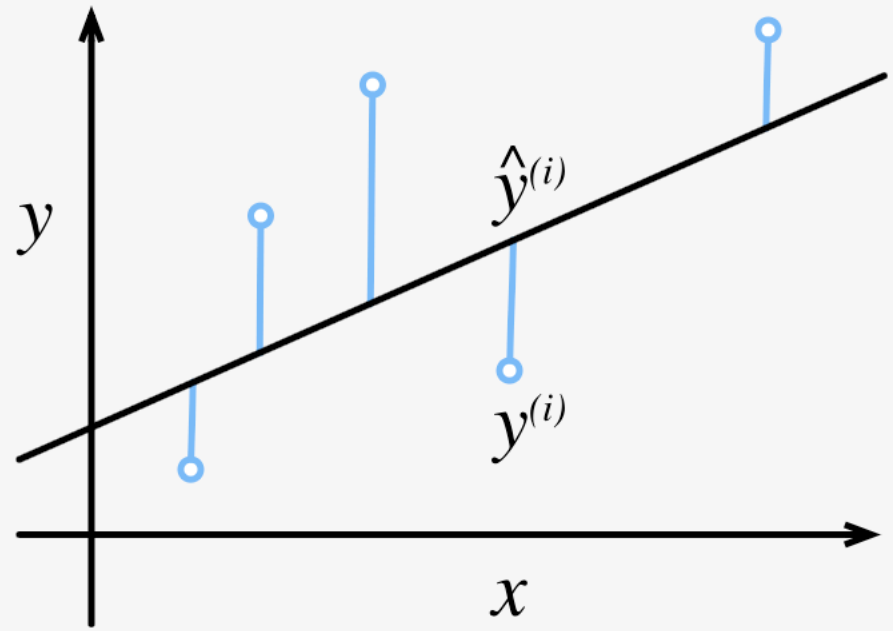


Fig. 3.1.1 Fitting a linear regression model to one-dimensional data.

출처 : Dive Into Deep Learning

$$\hat{y} = w_1 x_1 + \ldots + w_d x_d + b.$$

Prediction

$$\hat{y} = \mathbf{w}^\top \mathbf{x} + b.$$

Prediction in Matrix Form

$$l^{(i)}(\mathbf{w}, b) = \frac{1}{2}\left(\hat{y}^{(i)} - y^{(i)}\right)^2.$$

Loss function : squared error

Multi Layer Perceptron

Linear Neural Networks for Regression

- Paradigm shift to “Neural Networks”
- Linear regression은 single-layer fully connected neural network!

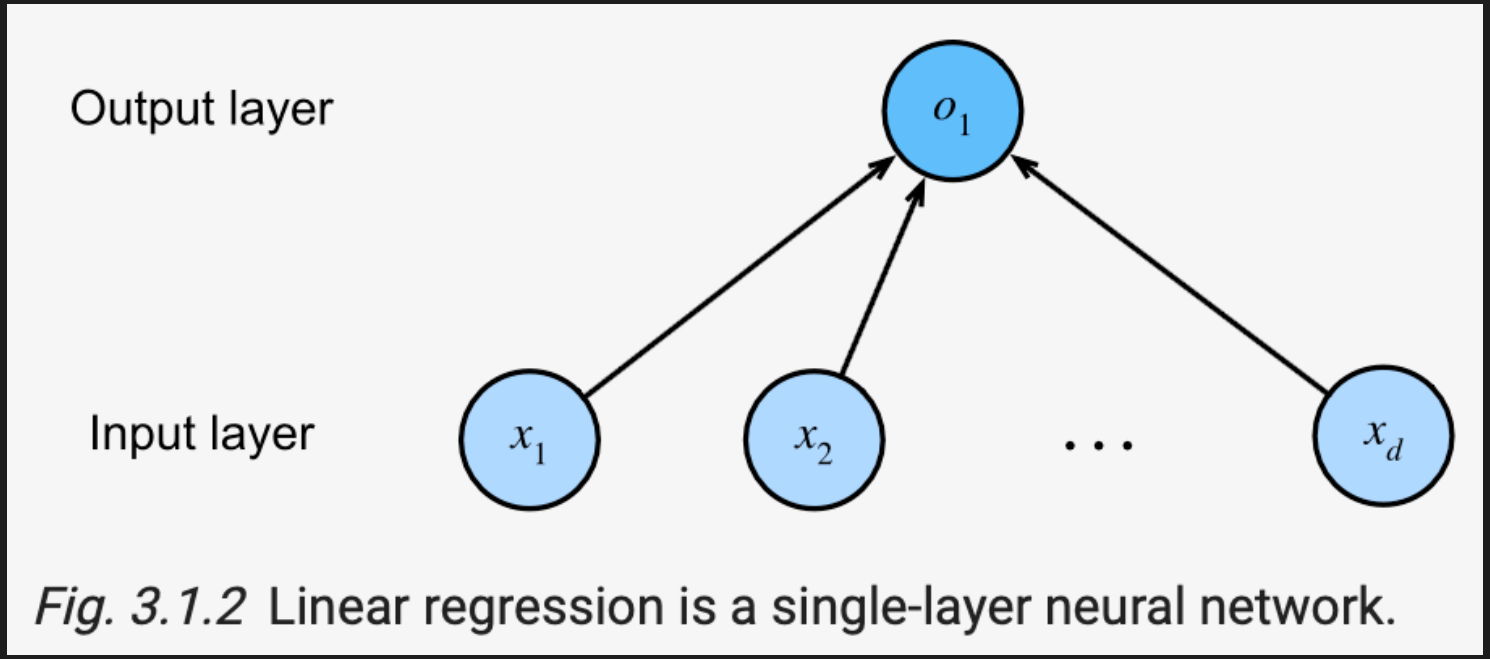
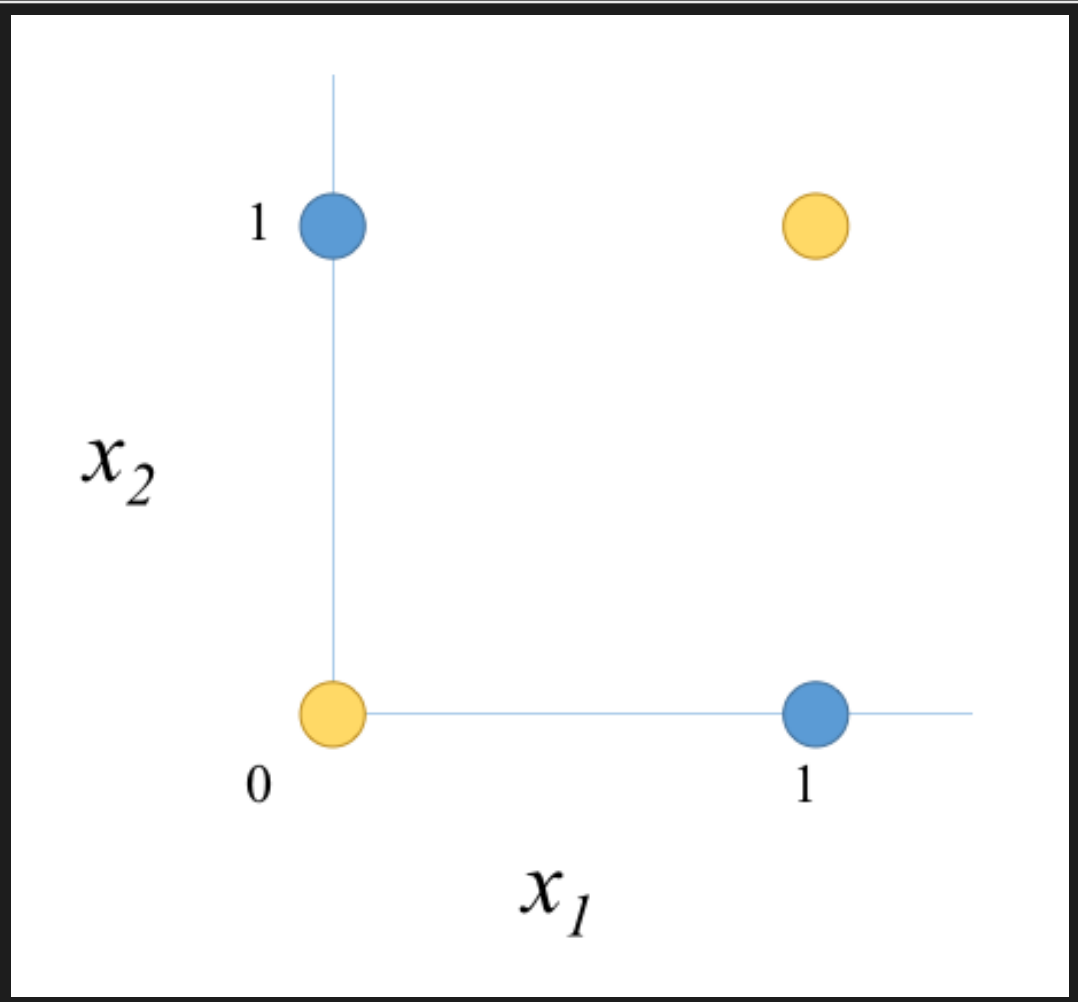


Fig. 3.1.2 Linear regression is a single-layer neural network.

출처 : Dive Into Deep Learning

“One Layer” is enough?

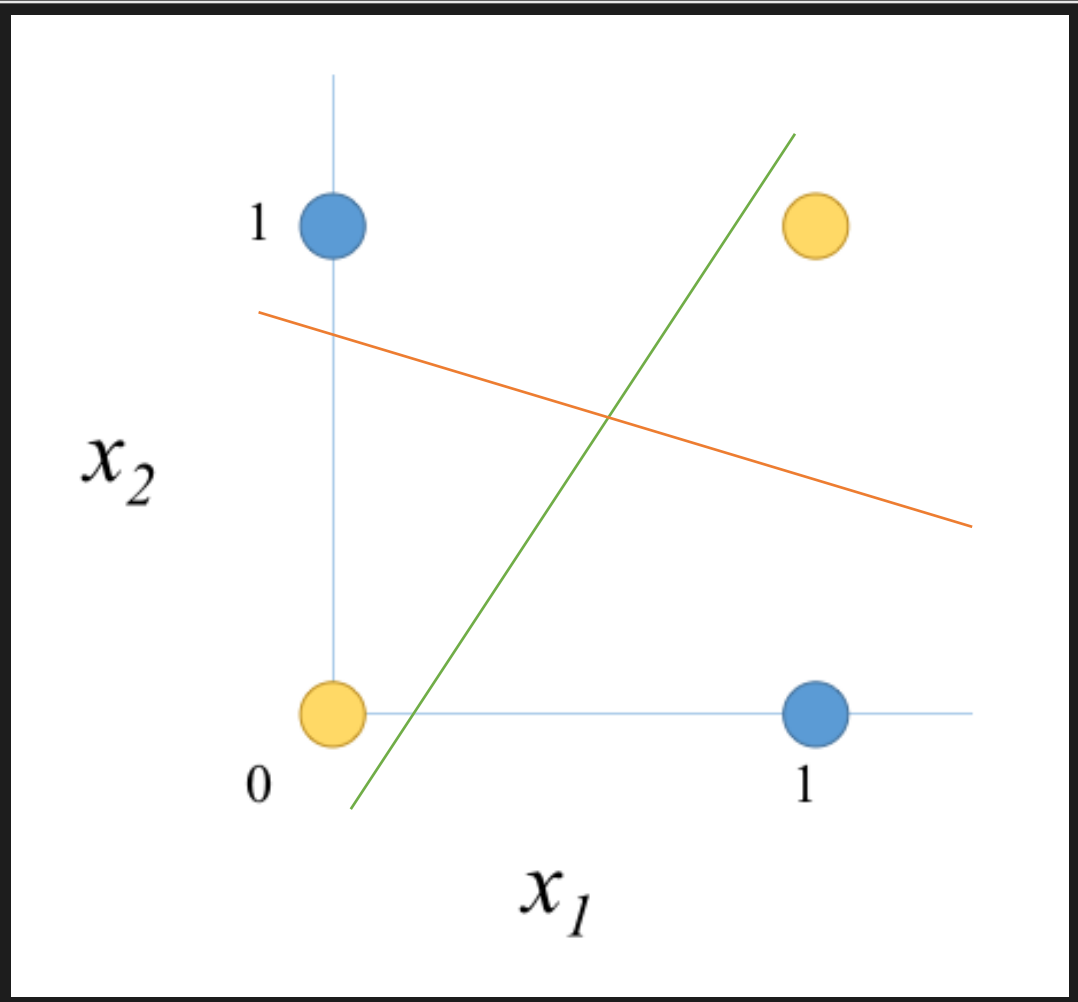
- XOR Problem
- 한 직선으로 점들을 나눌 수 있을까?



출처 : https://aimatters.wordpress.com/2016/01/11/solving-xor-with-a-neural-network-in-python/

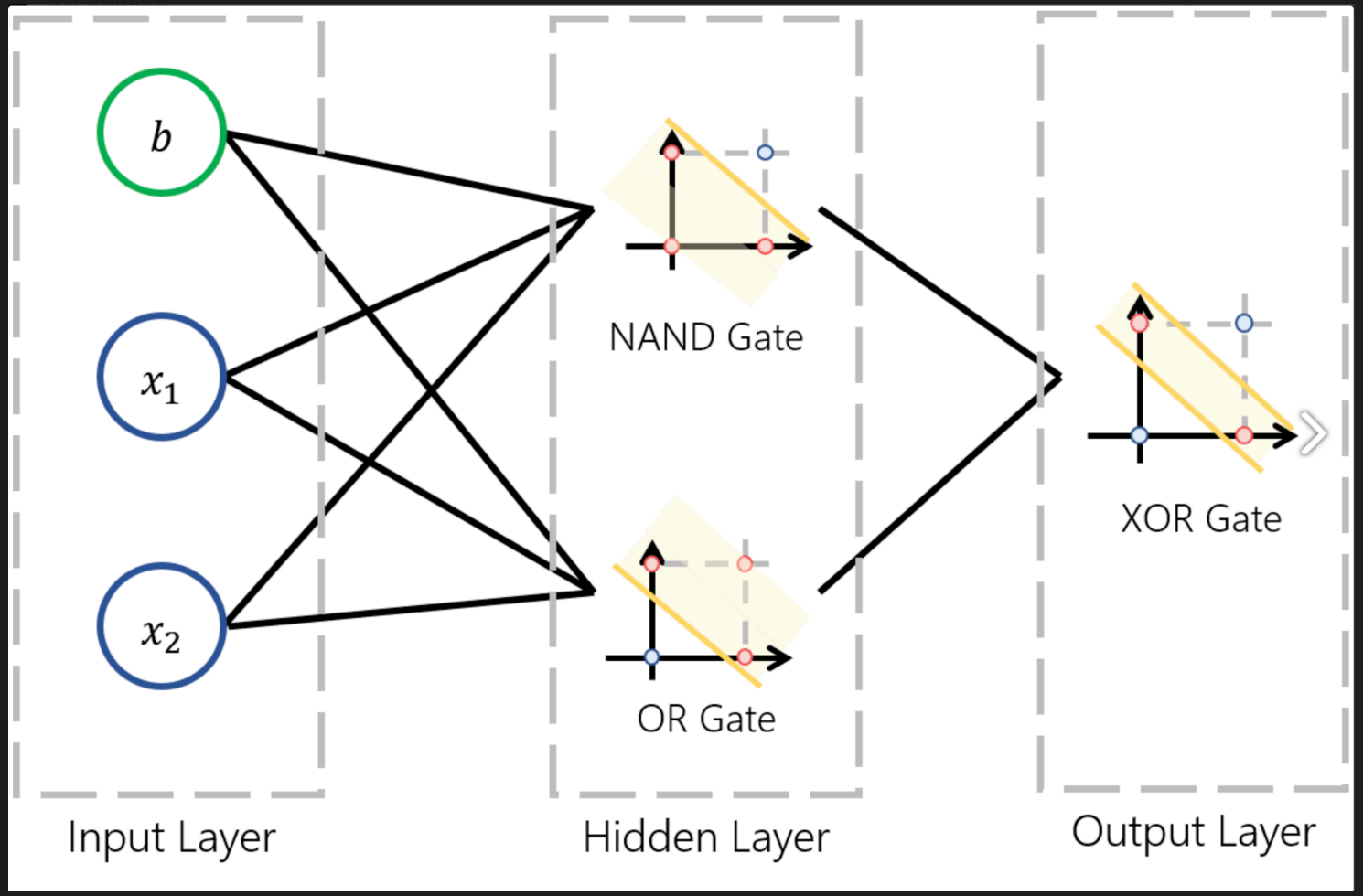
“One Layer” is enough?

- XOR Problem
- 한 직선으로 점들을 나눌 수 있을까?
- No… We need more layers!



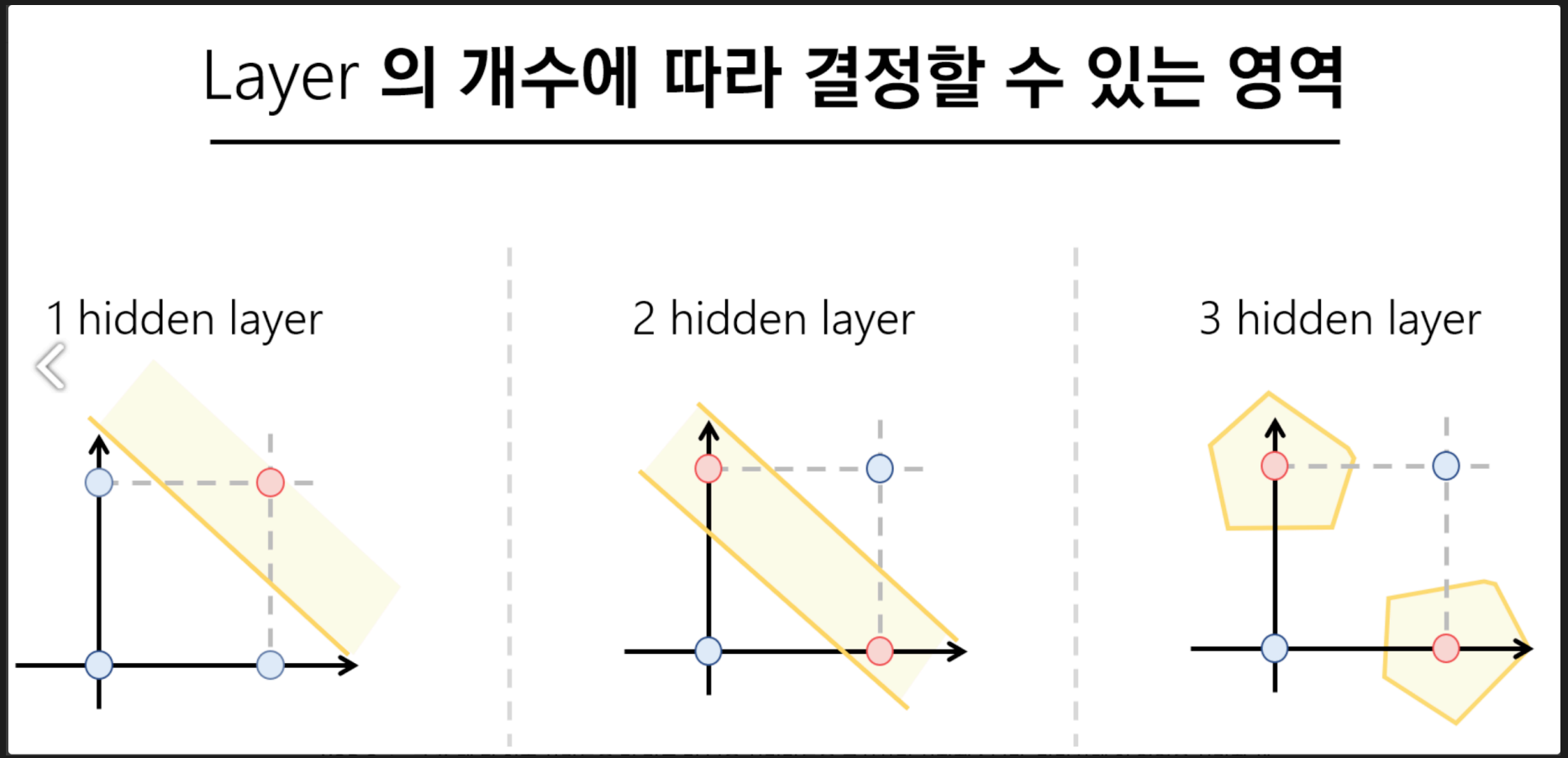
출처 : https://aimatters.wordpress.com/2016/01/11/solving-xor-with-a-neural-network-in-python/

Hidden Layers



출처 : https://gomguard.tistory.com/178

Hidden Layers



출처 : https://gomguard.tistory.com/178

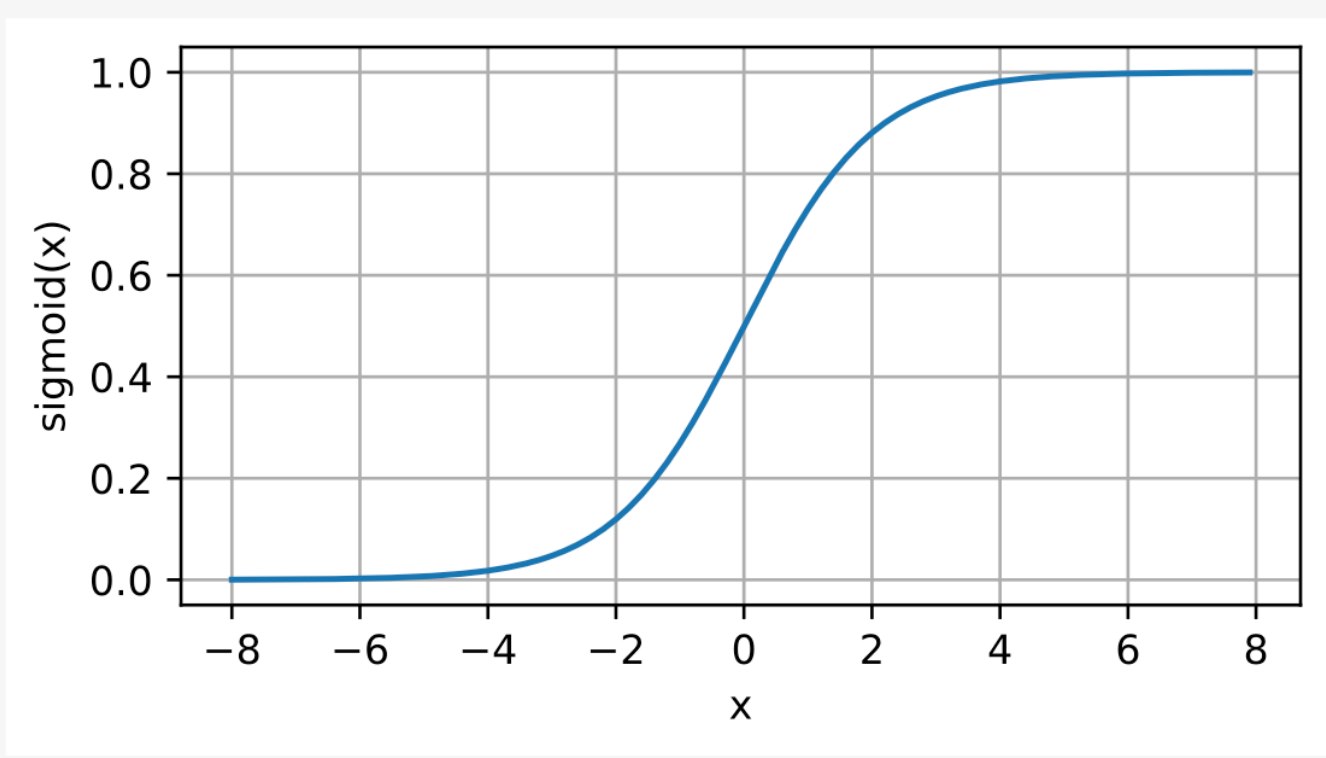
“Linear” is okay?

- 층을 쌓는다 == 합성 함수를 만든다.
- 선현성(linearity)에 대한 정의
  - 동차성(Homogeneity). $f(\alpha u) = \alpha f(u)$
  - 가산성(Additivity). $f(x + y) = f(x) + f(y)$
- 즉, Linear Layer 100만층을 쌓아도, 그냥 1층 Linear 함수이다!!

“Nonlinear” activation function

- Linear combination이 안 되게 하고 싶다!
  - Linear Layer 사이에 Nonlinear 함수를 끼워 넣자!

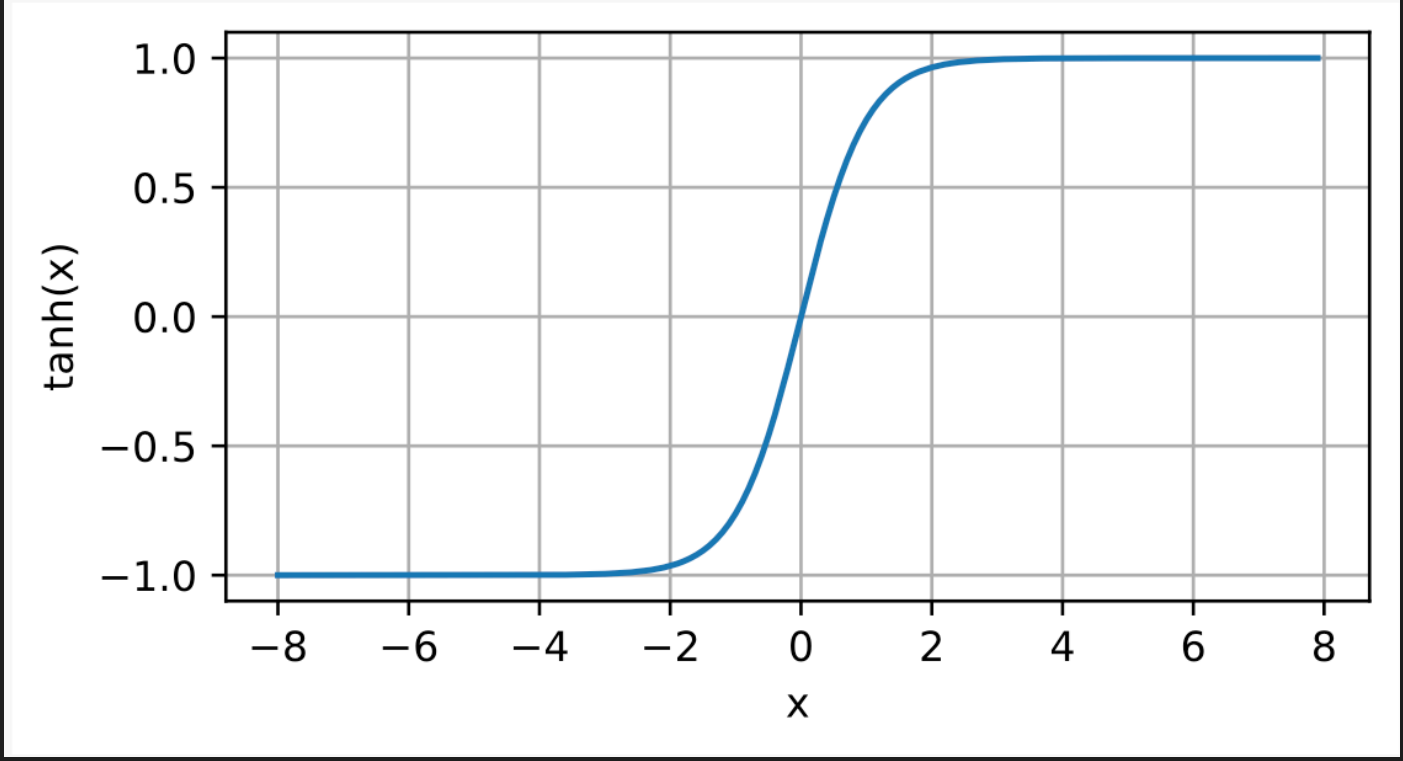
“Nonlinear” activation function



출처 : Dive Into Deep Learning

- Sigmoid Function
- $sigmoid(x) = \frac{1}{1 + \exp(-x)}$

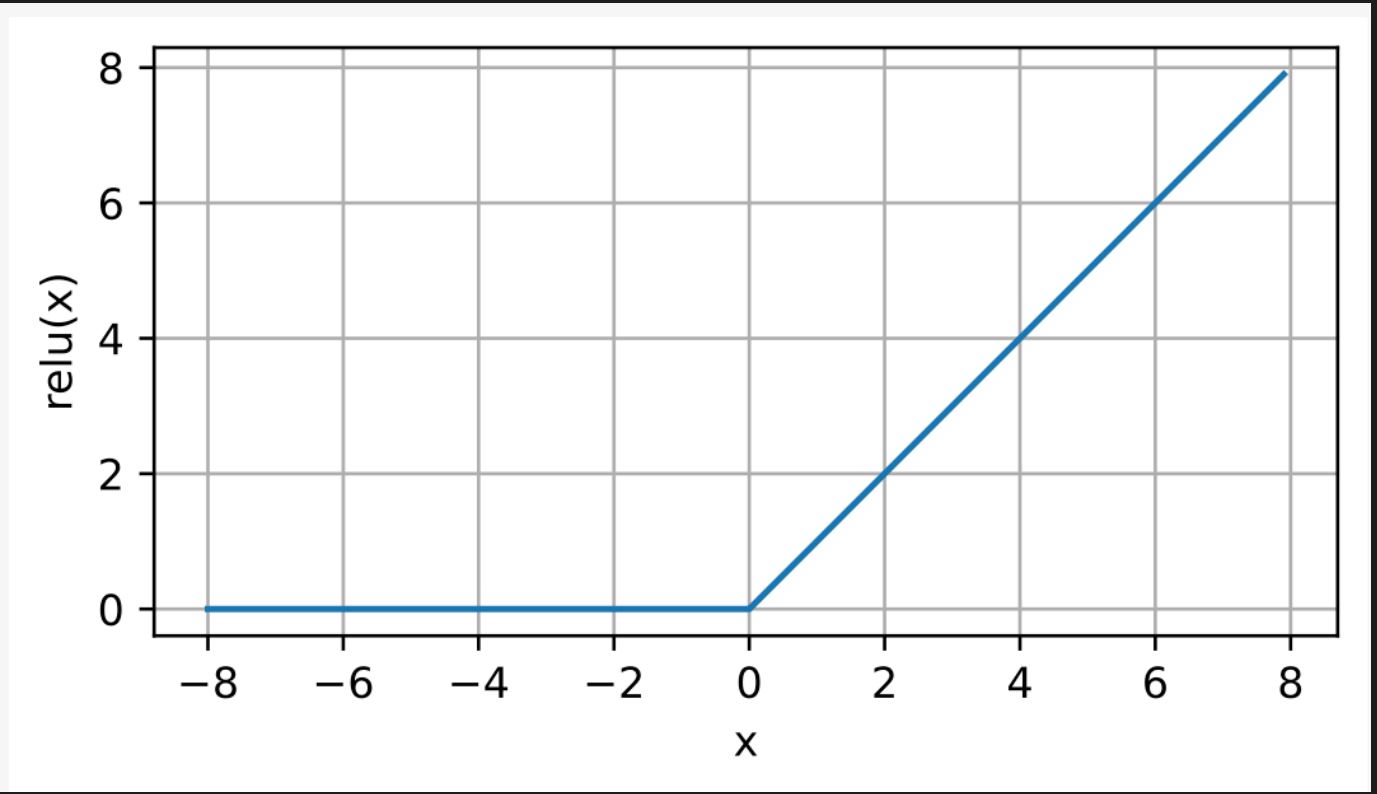
“Nonlinear” activation function



출처 : Dive Into Deep Learning

- Tanh Function
- $tanh(x) = \frac{1 - \exp(-2x)}{1 + \exp(-2x)}$

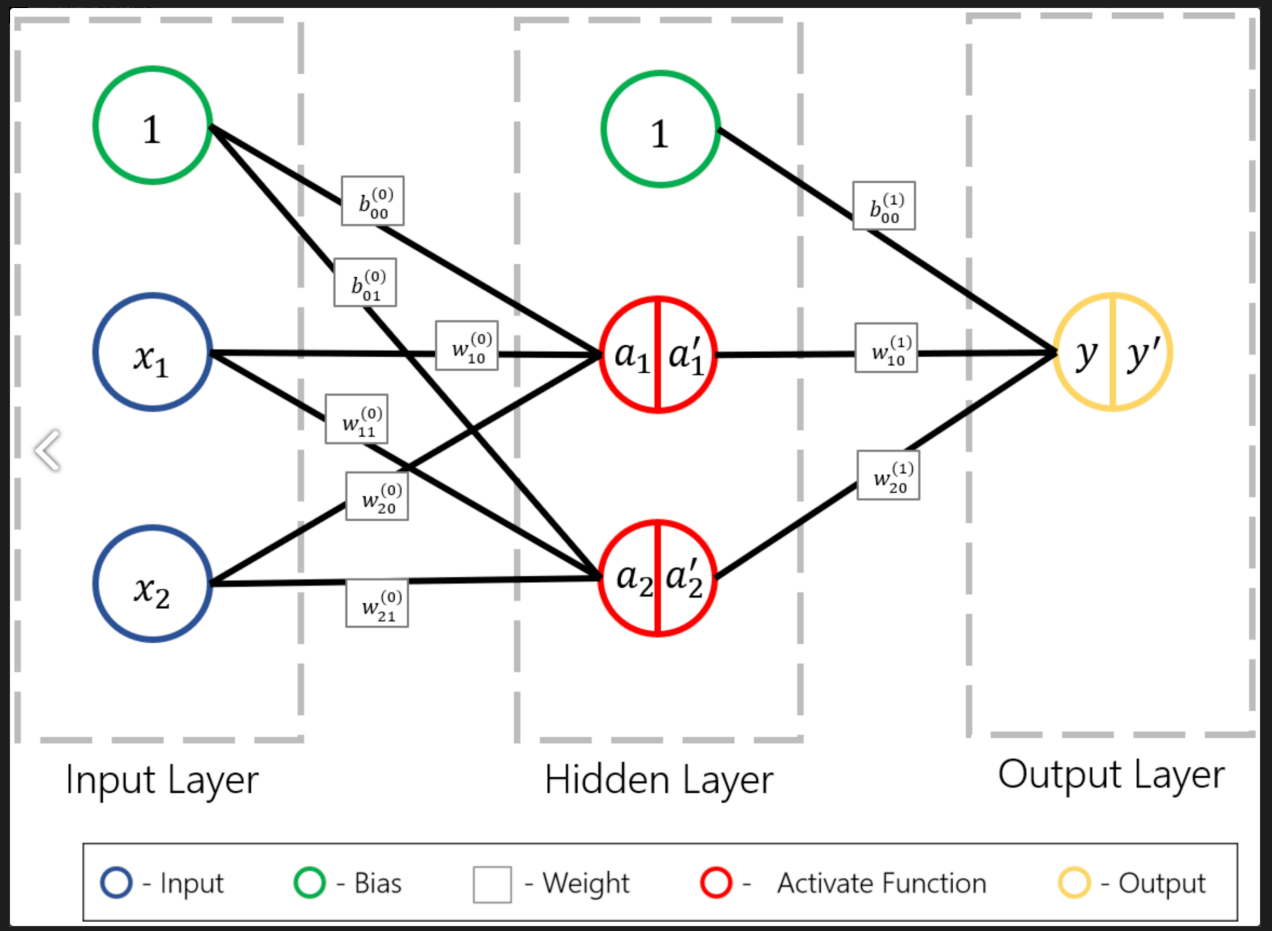
“Nonlinear” activation function



출처 : Dive Into Deep Learning

- ReLU Function
- $ReLU(x) = max(x, 0)$

Multi Layer Perceptrons



출처 : https://gomguard.tistory.com/178

Multi Layer Perceptrons

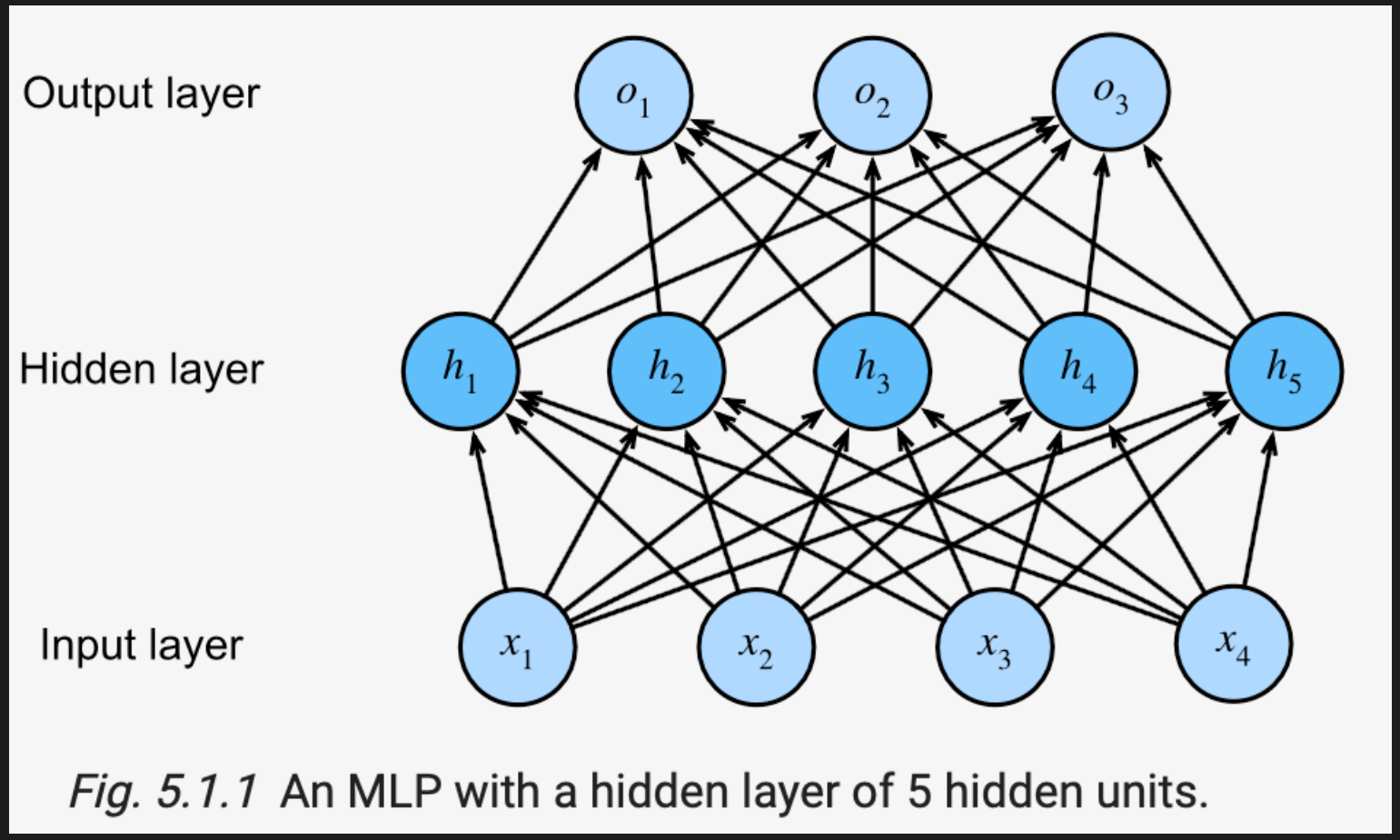


Fig. 5.1.1 An MLP with a hidden layer of 5 hidden units.

출처 : Dive Into Deep Learning

Universal Approximation Theorem

In the mathematical theory of artificial neural networks, **universal approximation theorems** are results[1][2] that establish the density of an algorithmically generated class of functions within a given function space of interest. Typically, these results concern the approximation capabilities of the feedforward architecture on the space of continuous functions between two Euclidean spaces, and the approximation is with respect to the compact convergence topology.

However, there are also a variety of results between non-Euclidean spaces[3] and other commonly used architectures and, more generally, algorithmically generated sets of functions, such as the convolutional neural network (CNN) architecture,[4][5] radial basis functionss,[6] or neural networks with specific properties.[7][8] Most universal approximation theorems can be parsed into two classes. The first quantifies the approximation capabilities of neural networks with an arbitrary number of artificial neurons ("*arbitrary width*" case) and the second focuses on the case with an arbitrary number of hidden layers, each containing a limited number of artificial neurons ("*arbitrary depth*" case). In addition to these two classes, there are also universal approximation theorems for neural networks with bounded number of hidden layers and a limited number of neurons in each layer ("*bounded depth and bounded width*" case).

Universal approximation theorems imply that neural networks can *represent* a wide variety of interesting functions when given appropriate weights. On the other hand, they typically do not provide a construction for the weights, but merely state that such a construction is possible.

출처 : https://en.wikipedia.org/wiki/Universal_approximation_theorem

- To be simple…임의의 개수의 Perceptron을 가진, Non linear activation function을 가진, 1 hidden layer의 Feed-Forward Neural Network는 적절한 Weight가 있다면, 어떤 연속함수든 근사할 수 있다.

Then… many layers are solution?

- 그럼 FCN (Fully-Connected Layer) 만능 아님?
  - 당연히 아닙니다. 그냥 '이론상 가능하다.'
- 왜 안 될까요?
  - Vanishing gradient Problem
  - Too many parameters
- It's deep learning time!

In our brains…

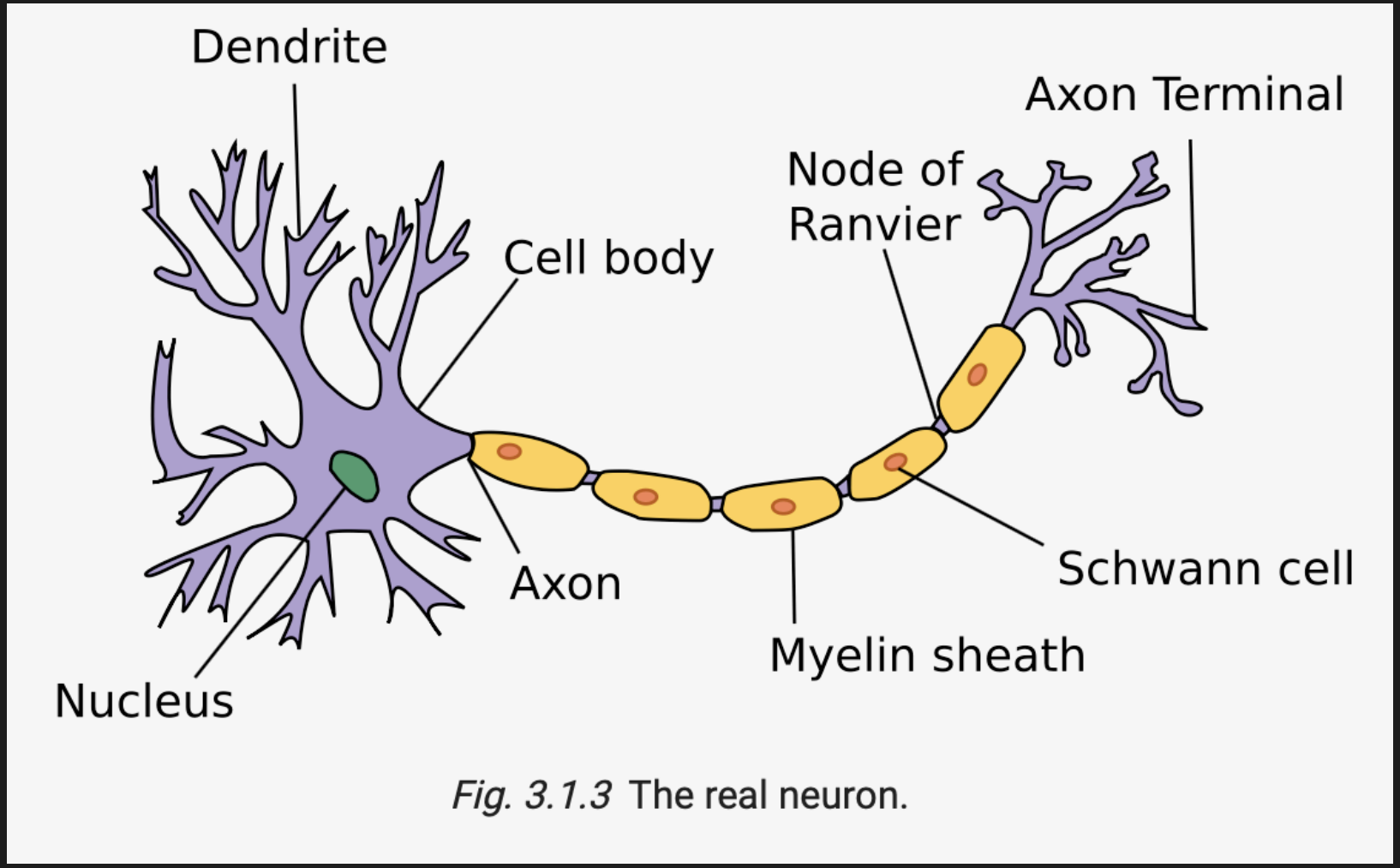
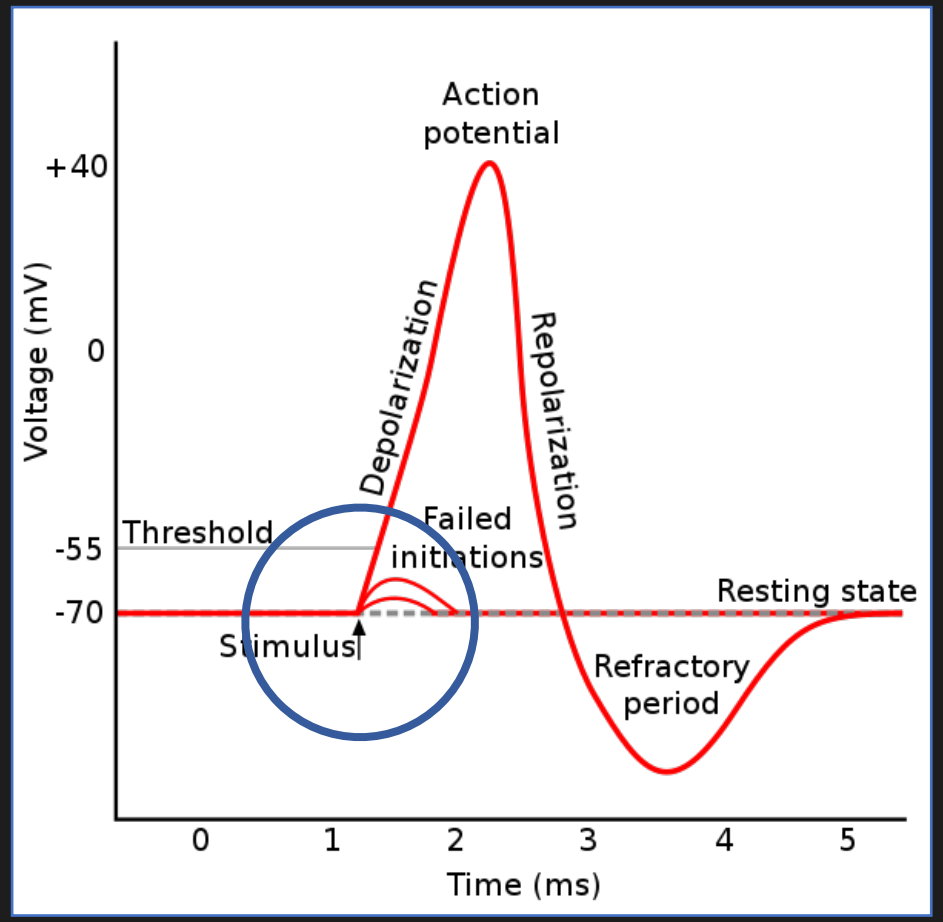


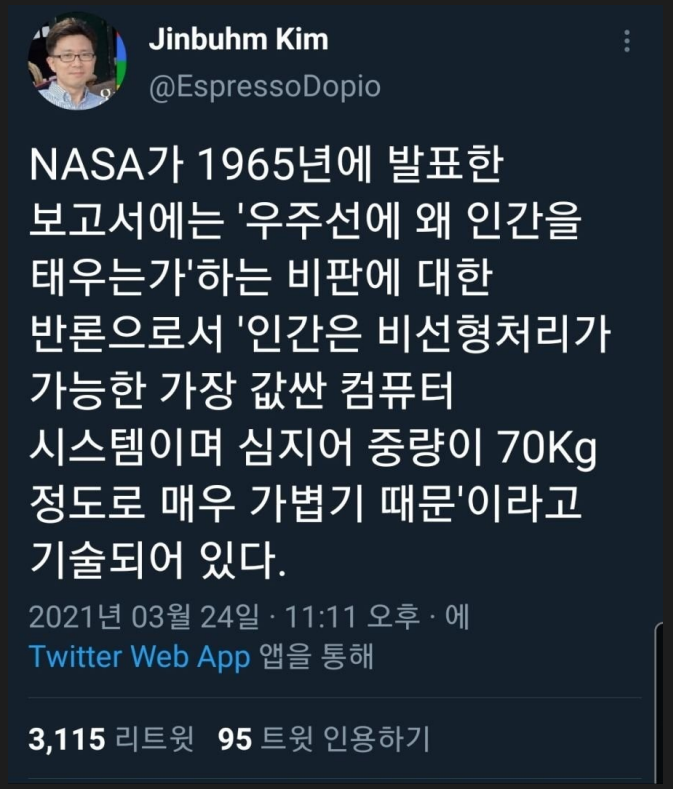
Fig. 3.1.3 The real neuron.

출처 : Dive Into Deep Learning

Magic of Non-linearity



출처 : https://ko.wikipedia.org/wiki/활동전위



이제 이해 되시나요?

⚠️ Caution ⚠️

출처 : https://www.discoverwildlife.com/animal-facts/birds/heaviest-flying-bird/

≠

출처 : https://www.businessinsider.com/vueling-plane-excess-fuel-passengers-fly-low-altitude-hours-report-2022-8

How can we train MLP?

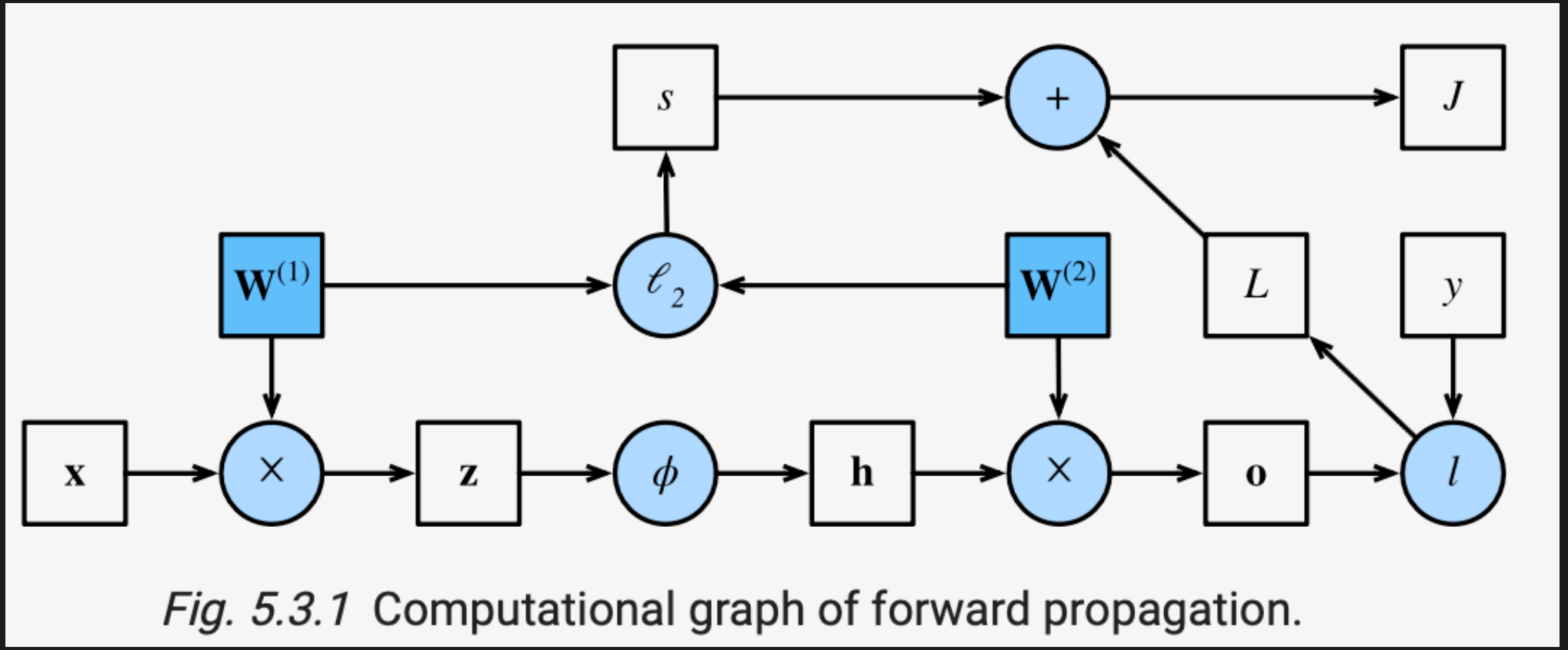
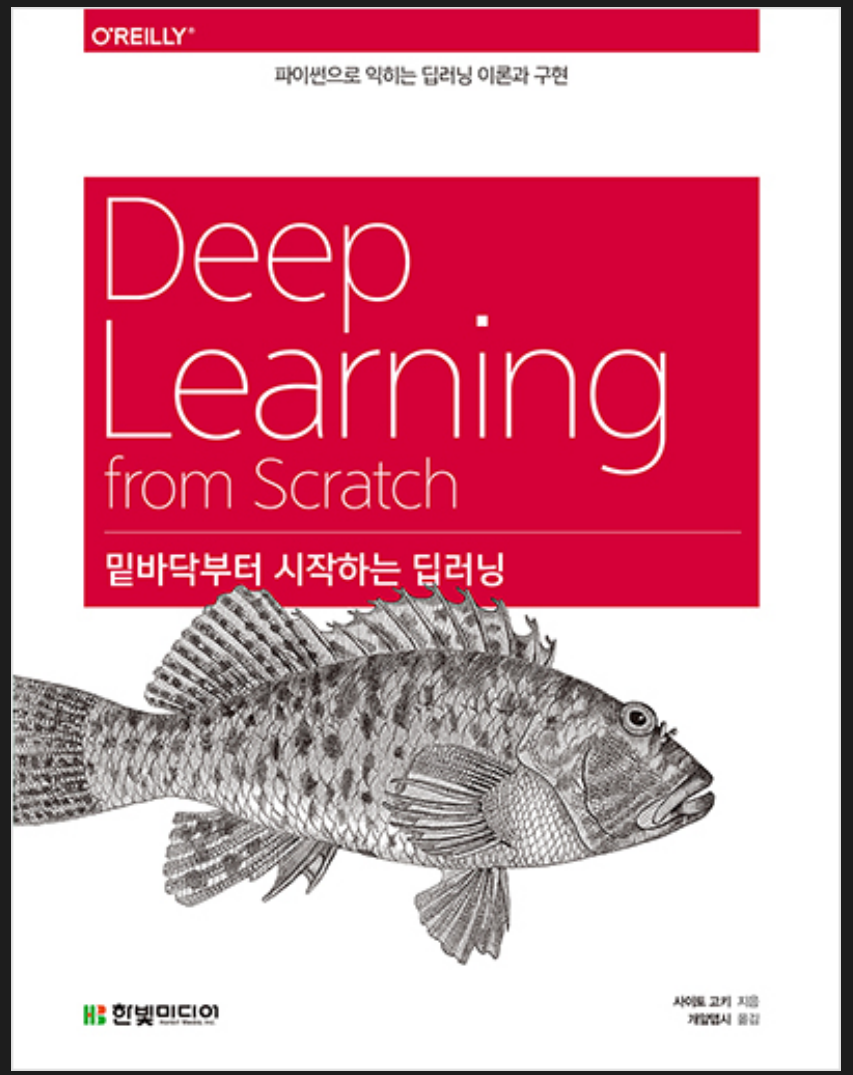


Fig. 5.3.1 Computational graph of forward propagation.

출처 : Dive Into Deep Learning

How can we train MLP?

- Backpropagation!
- Simple chain rule
- ‘밑바닥부터 시작하는 딥러닝’ 1권 강추
- torch.autograd



Discussion Assignment

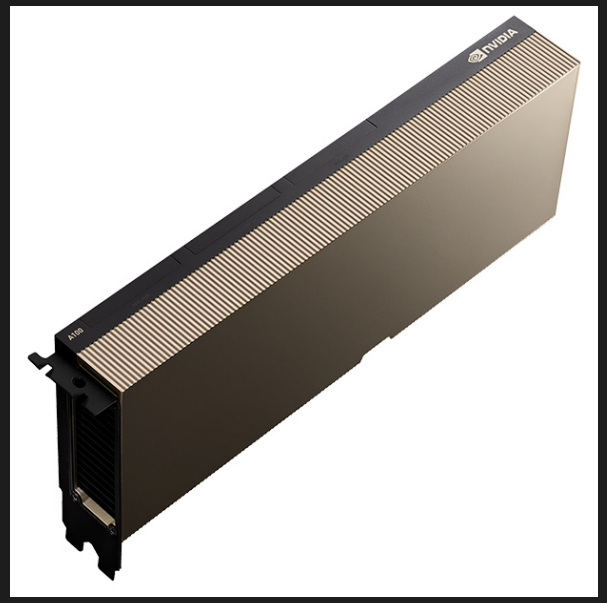
- Neural Network는 깊을 수록 좋은가?
- 울퉁불퉁한 landscape를 가진 loss function을 학습할 때, local minima를 벗어나려면 어떻게 하면 좋을까?
- Due day ~다음 수업 시작 전까지
  - AIKU Notion → 일정 → DeepIntoDeep 해당 회차
  - 본인 칸에 자신의 생각 적기!

The Road to Deep Learning

Table 1.5.1 Dataset vs. computer memory and computational power

Decade	Dataset	Memory	Floating point calculations per second
1970	100 (Iris)	1 KB	100 KF (Intel 8080)
1980	1 K (house prices in Boston)	100 KB	1 MF (Intel 80186)
1990	10 K (optical character recognition)	10 MB	10 MF (Intel 80486)
2000	10 M (web pages)	100 MB	1 GF (Intel Core)
2010	10 G (advertising)	1 GB	1 TF (Nvidia C2050)
2020	1 T (social network)	100 GB	1 PF (Nvidia DGX-2)

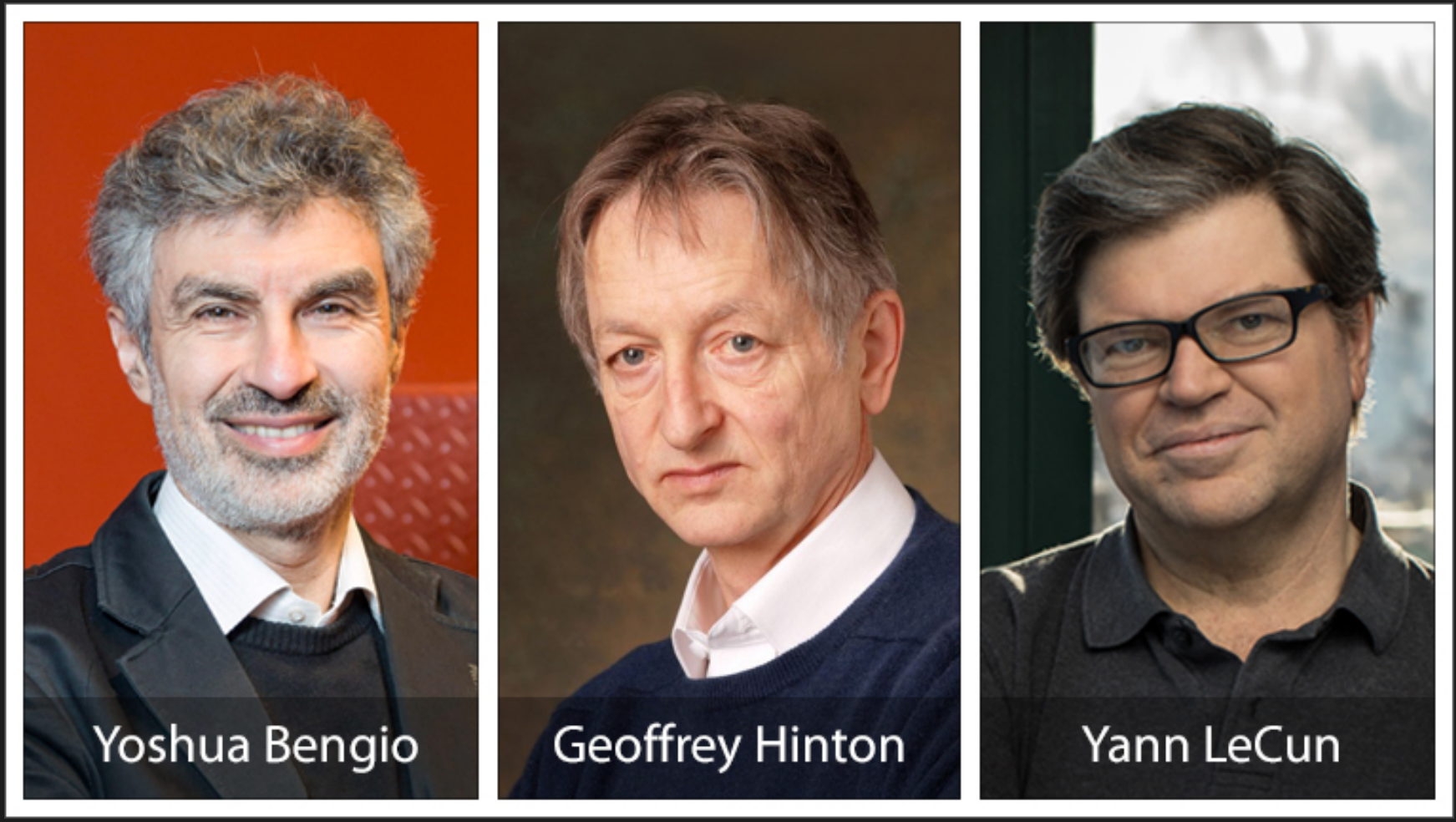
출처 : Dive Into Deep Learning



NVIDA A100

가…가즈아!!!

The Road to Deep Learning



출처 : https://awards.acm.org/about/2018-turing

감사합니다!